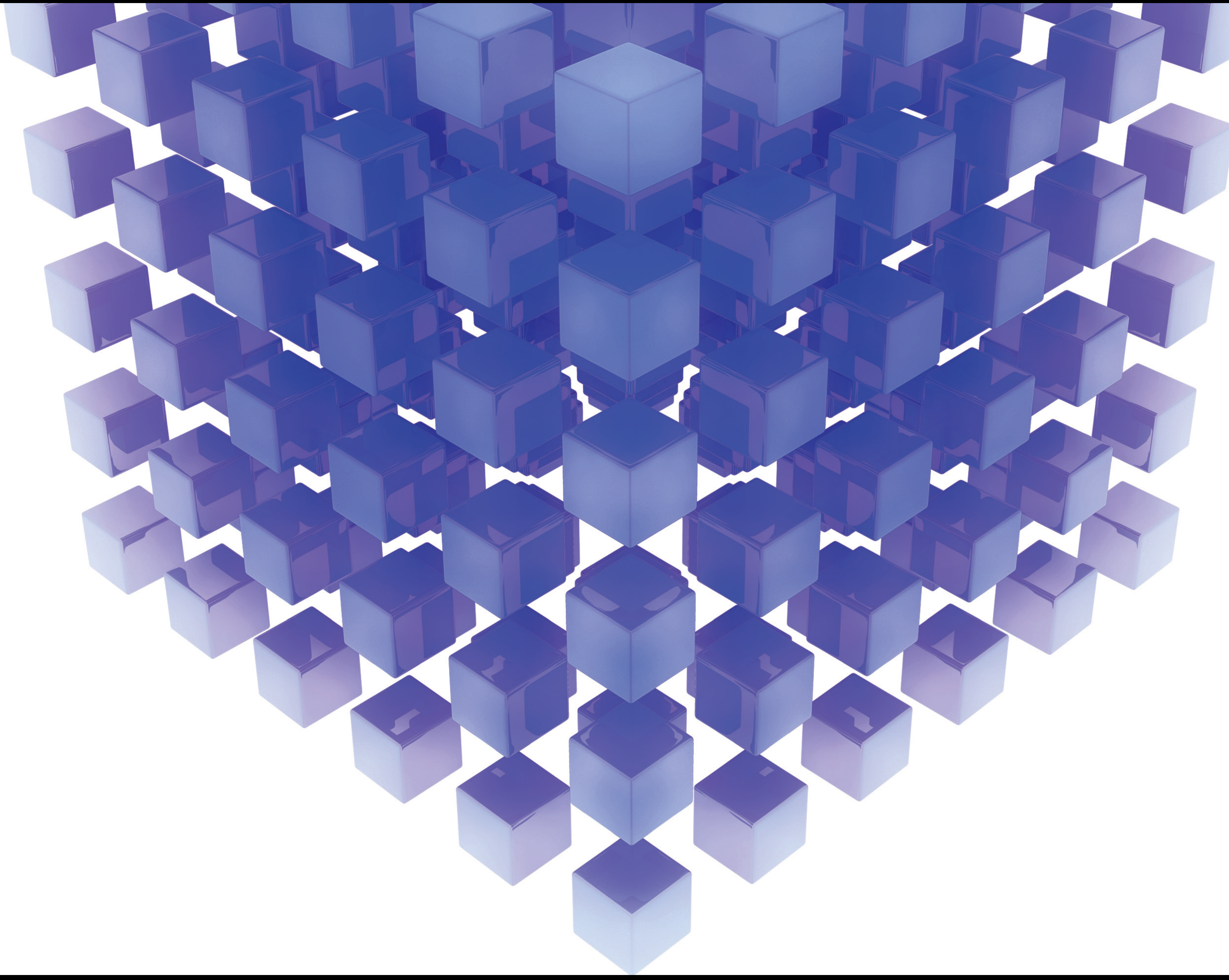
Mathematical Problems in Engineering

Meta-Heuristic Techniques for Solving Computational Engineering Problems

Lead Guest Editor: Dilbag Singh
Guest Editors: Kehui Sun and Manjit Kaur



Meta-Heuristic Techniques for Solving Computational Engineering Problems

Mathematical Problems in Engineering

Meta-Heuristic Techniques for Solving Computational Engineering Problems

Lead Guest Editor: Dilbag Singh
Guest Editors: Kehui Sun and Manjit Kaur

Chief Editor

Academic Editors

Paolo Boscariol, Italy
Daniela Boso, Italy
Guillermo Botella-Juan, Spain
Abdesselem Boulkroune, Algeria
Boulaïd Boulkroune, Belgium
Fabio Bovenga, Italy
Francesco Braghin, Italy
Ricardo Branco, Portugal
Julien Bruchon, France
Matteo Bruggi, Italy
Michele Brun, Italy
Maria Elena Bruni, Italy
Maria Angela Butturi, Italy
Bartłomiej Błachowski, Poland
Dhanamjayulu C, India
Raquel Caballero-Águila, Spain
Filippo Cacace, Italy
Salvatore Caddemi, Italy
Zuowei Cai, China
Roberto Caldelli, Italy
Francesco Cannizzaro, Italy
Maosen Cao, China
Ana Carpio, Spain
Rodrigo Carvajal, Chile
Caterina Casavola, Italy
Sara Casciati, Italy
Federica Caselli, Italy
Carmen Castillo, Spain
Inmaculada T. Castro, Spain
Miguel Castro, Portugal
Giuseppe Catalanotti, United Kingdom
Alberto Cavallo, Italy
Gabriele Cazzulani, Italy
Fatih Vehbi Celebi, Turkey
Miguel Cerrolaza, Venezuela
Gregory Chagnon, France
Ching-Ter Chang, Taiwan
Kuei-Lun Chang, Taiwan
Qing Chang, USA
Xiaoheng Chang, China
Prasenjit Chatterjee, Lithuania
Kacem Chehdi, France
Peter N. Cheimets, USA
Chih-Chiang Chen, Taiwan
He Chen, China
Kebing Chen, China
Mengxin Chen, China
Shyi-Ming Chen, Taiwan
Xizhong Chen, Ireland
Xue-Bo Chen, China
Zhiwen Chen, China
Qiang Cheng, USA
Zeyang Cheng, China
Luca Chiapponi, Italy
Francisco Chicano, Spain
Tirivanhu Chinyoka, South Africa
Adrian Chmielewski, Poland
Seongim Choi, USA
Gautam Choubey, India
Hung-Yuan Chung, Taiwan
Yusheng Ci, China
Simone Cinquemani, Italy
Roberto G. Citarella, Italy
Joaquim Ciurana, Spain
John D. Clayton, USA
Piero Colajanni, Italy
Giuseppina Colicchio, Italy
Vassilios Constantoudis, Greece
Enrico Conte, Italy
Alessandro Contento, USA
Mario Cools, Belgium
Gino Cortellessa, Italy
Carlo Cosentino, Italy
Paolo Crippa, Italy
Erik Cuevas, Mexico
Guozeng Cui, China
Mehmet Cunkas, Turkey
Giuseppe D'Aniello, Italy
Peter Dabnichki, Australia
Weizhong Dai, USA
Zhifeng Dai, China
Purushothaman Damodaran, USA
Sergey Dashkovskiy, Germany
Adiel T. De Almeida-Filho, Brazil
Fabio De Angelis, Italy
Samuele De Bartolo, Italy
Stefano De Miranda, Italy
Filippo De Monte, Italy

Antoine Grall, France
George A. Gravvanis, Greece
Fabrizio Greco, Italy
David Greiner, Spain
Jason Gu, Canada
Federico Guarracino, Italy
Michele Guida, Italy
Muhammet Gul, Turkey
Dong-Sheng Guo, China
Hu Guo, China
Zhaoxia Guo, China
Yusuf Gurefe, Turkey
Salim HEDDAM, Algeria
ABID HUSSANAN, China
Quang Phuc Ha, Australia
Li Haitao, China
Petr Hájek, Czech Republic
Mohamed Hamdy, Egypt
Muhammad Hamid, United Kingdom
Renke Han, United Kingdom
Weimin Han, USA
Xingsi Han, China
Zhen-Lai Han, China
Thomas Hanne, Switzerland
Xinan Hao, China
Mohammad A. Hariri-Ardebili, USA
Khalid Hattaf, Morocco
Defeng He, China
Xiao-Qiao He, China
Yanchao He, China
Yu-Ling He, China
Ramdane Hedjar, Saudi Arabia
Jude Hemanth, India
Reza Hemmati, Iran
Nicolae Herisanu, Romania
Alfredo G. Hernández-Diaz, Spain
M.I. Herreros, Spain
Eckhard Hitzer, Japan
Paul Honeine, France
Jaromir Horacek, Czech Republic
Lei Hou, China
Yingkun Hou, China
Yu-Chen Hu, Taiwan
Yunfeng Hu, China
Can Huang, China
Gordon Huang, Canada
Linsheng Huo, China
Sajid Hussain, Canada
Asier Ibeas, Spain
Orest V. Iftime, The Netherlands
Przemyslaw Ignaciuk, Poland
Giacomo Innocenti, Italy
Emilio Insfran Pelozo, Spain
Azeem Irshad, Pakistan
Alessio Ishizaka, France
Benjamin Ivorra, Spain
Breno Jacob, Brazil
Reema Jain, India
Tushar Jain, India
Amin Jajarmi, Iran
Chiranjibe Jana, India
Łukasz Jankowski, Poland
Samuel N. Jator, USA
Juan Carlos Jáuregui-Correa, Mexico
Kandasamy Jayakrishna, India
Reza Jazar, Australia
Khalide Jbilou, France
Isabel S. Jesus, Portugal
Chao Ji, China
Qing-Chao Jiang, China
Peng-fei Jiao, China
Ricardo Fabricio Escobar Jiménez, Mexico
Emilio Jiménez Macías, Spain
Maolin Jin, Republic of Korea
Zhuo Jin, Australia
Ramash Kumar K, India
BHABEN KALITA, USA
MOHAMMAD REZA KHEDMATI, Iran
Viacheslav Kalashnikov, Mexico
Mathiyalagan Kalidass, India
Tamas Kalmar-Nagy, Hungary
Rajesh Kaluri, India
Jyotheeswara Reddy Kalvakurthi, India
Zhao Kang, China
Ramani Kannan, Malaysia
Tomasz Kapitaniak, Poland
Julius Kaplunov, United Kingdom
Konstantinos Karamanos, Belgium
Michal Kawulok, Poland

Francisco J. Martos, Spain
Elio Masciari, Italy
Paolo Massioni, France
Alessandro Mauro, Italy
Jonathan Mayo-Maldonado, Mexico
Pier Luigi Mazzeo, Italy
Laura Mazzola, Italy
Driss Mehdi, France
Zahid Mehmood, Pakistan
Roderick Melnik, Canada
Xiangyu Meng, USA
Jose Merodio, Spain
Alessio Merola, Italy
Mahmoud Mesbah, Iran
Luciano Mescia, Italy
Laurent Mevel, France
Constantine Michailides, Cyprus
Mariusz Michta, Poland
Prankul Middha, Norway
Aki Mikkola, Finland
Giovanni Minafò, Italy
Edmondo Minisci, United Kingdom
Hiroyuki Mino, Japan
Dimitrios Mitsotakis, New Zealand
Ardashir Mohammadzadeh, Iran
Francisco J. Montáns, Spain
Francesco Montefusco, Italy
Gisele Mophou, France
Rafael Morales, Spain
Marco Morandini, Italy
Javier Moreno-Valenzuela, Mexico
Simone Morganti, Italy
Caroline Mota, Brazil
Aziz Moukrim, France
Shen Mouquan, China
Dimitris Mourtzis, Greece
Emiliano Mucchi, Italy
Taseer Muhammad, Saudi Arabia
Ghulam Muhiuddin, Saudi Arabia
Amitava Mukherjee, India
Josefa Mula, Spain
Jose J. Muñoz, Spain
Giuseppe Muscolino, Italy
Marco Mussetta, Italy
Hariharan Muthusamy, India
Alessandro Naddeo, Italy
Raj Nandkeolyar, India
Keivan Navaie, United Kingdom
Soumya Nayak, India
Adrian Neagu, USA
Erivelton Geraldo Nepomuceno, Brazil
AMA Neves, Portugal
Ha Quang Thinh Ngo, Vietnam
Nhon Nguyen-Thanh, Singapore
Papakostas Nikolaos, Ireland
Jelena Nikolic, Serbia
Tatsushi Nishi, Japan
Shanzhou Niu, China
Ben T. Nohara, Japan
Mohammed Nouari, France
Mustapha Nourelfath, Canada
Kazem Nouri, Iran
Ciro Núñez-Gutiérrez, Mexico
Wlodzimierz Ogryczak, Poland
Roger Ohayon, France
Krzysztof Okarma, Poland
Mitsuhiro Okayasu, Japan
Murat Olgun, Turkey
Diego Oliva, Mexico
Alberto Olivares, Spain
Enrique Onieva, Spain
Calogero Orlando, Italy
Susana Ortega-Cisneros, Mexico
Sergio Ortobelli, Italy
Naohisa Otsuka, Japan
Sid Ahmed Ould Ahmed Mahmoud, Saudi Arabia
Taoreed Owolabi, Nigeria
EUGENIA PETROPOULOU, Greece
Arturo Pagano, Italy
Madhumangal Pal, India
Pasquale Palumbo, Italy
Dragan Pamučar, Serbia
Weifeng Pan, China
Chandan Pandey, India
Rui Pang, United Kingdom
Jürgen Pannek, Germany
Elena Panteley, France
Achille Paolone, Italy

George A. Papakostas, Greece
Xosé M. Pardo, Spain
You-Jin Park, Taiwan
Manuel Pastor, Spain
Pubudu N. Pathirana, Australia
Surajit Kumar Paul, India
Luis Payá, Spain
Igor Pažanin, Croatia
Libor Pekař, Czech Republic
Francesco Pellicano, Italy
Marcello Pellicciari, Italy
Jian Peng, China
Mingshu Peng, China
Xiang Peng, China
Xindong Peng, China
Yuexing Peng, China
Marzio Pennisi, Italy
Maria Patrizia Pera, Italy
Matjaz Perc, Slovenia
A. M. Bastos Pereira, Portugal
Wesley Peres, Brazil
F. Javier Pérez-Pinal, Mexico
Michele Perrella, Italy
Francesco Pesavento, Italy
Francesco Petrini, Italy
Hoang Vu Phan, Republic of Korea
Lukasz Pieczonka, Poland
Dario Piga, Switzerland
Marco Pizzarelli, Italy
Javier Plaza, Spain
Goutam Pohit, India
Dragan Poljak, Croatia
Jorge Pomares, Spain
Hiram Ponce, Mexico
Sébastien Poncet, Canada
Volodymyr Ponomaryov, Mexico
Jean-Christophe Ponsart, France
Mauro Pontani, Italy
Sivakumar Poruran, India
Francesc Pozo, Spain
Aditya Rio Prabowo, Indonesia
Anchasa Pramuanjaroenkij, Thailand
Leonardo Primavera, Italy
B Rajanarayan Prusty, India
Krzysztof Puszynski, Poland
Chuan Qin, China
Dongdong Qin, China
Jianlong Qiu, China
Giuseppe Quaranta, Italy
DR. RITU RAJ, India
Vitomir Racic, Italy
Carlo Rainieri, Italy
Kumbakonam Ramamani Rajagopal, USA
Ali Ramazani, USA
Angel Manuel Ramos, Spain
Higinio Ramos, Spain
Muhammad Afzal Rana, Pakistan
Muhammad Rashid, Saudi Arabia
Manoj Rastogi, India
Alessandro Rasulo, Italy
S.S. Ravindran, USA
Abdolrahman Razani, Iran
Alessandro Reali, Italy
Jose A. Reinoso, Spain
Oscar Reinoso, Spain
Haijun Ren, China
Carlo Renno, Italy
Fabrizio Renno, Italy
Shahram Rezapour, Iran
Ricardo Riaza, Spain
Francesco Riganti-Fulginei, Italy
Gerasimos Rigatos, Greece
Francesco Ripamonti, Italy
Jorge Rivera, Mexico
Eugenio Roanes-Lozano, Spain
Ana Maria A. C. Rocha, Portugal
Luigi Rodino, Italy
Francisco Rodríguez, Spain
Rosana Rodríguez López, Spain
Francisco Rossomando, Argentina
Jose de Jesus Rubio, Mexico
Weiguo Rui, China
Rubén Ruiz, Spain
Ivan D. Rukhlenko, Australia
Dr. Eswaramoorthi S., India
Weichao SHI, United Kingdom
Chaman Lal Sabharwal, USA
Andrés Sáez, Spain

Bekir Sahin, Turkey
Laxminarayan Sahoo, India
John S. Sakellariou, Greece
Michael Sakellariou, Greece
Salvatore Salamone, USA
Jose Vicente Salcedo, Spain
Alejandro Salcido, Mexico
Alejandro Salcido, Mexico
Nunzio Salerno, Italy
Rohit Salgotra, India
Miguel A. Salido, Spain
Sinan Salih, Iraq
Alessandro Salvini, Italy
Abdus Samad, India
Sovan Samanta, India
Nikolaos Samaras, Greece
Ramon Sancibrian, Spain
Giuseppe Sanfilippo, Italy
Omar-Jacobo Santos, Mexico
J Santos-Reyes, Mexico
José A. Sanz-Herrera, Spain
Musavarah Sarwar, Pakistan
Shahzad Sarwar, Saudi Arabia
Marcelo A. Savi, Brazil
Andrey V. Savkin, Australia
Tadeusz Sawik, Poland
Roberta Sburlati, Italy
Gustavo Scaglia, Argentina
Thomas Schuster, Germany
Hamid M. Sedighi, Iran
Mijanur Rahaman Seikh, India
Tapan Senapati, China
Lotfi Senhadji, France
Junwon Seo, USA
Michele Serpilli, Italy
Silvestar Šesnić, Croatia
Gerardo Severino, Italy
Ruben Sevilla, United Kingdom
Stefano Sfarra, Italy
Dr. Ismail Shah, Pakistan
Leonid Shaikhet, Israel
Vimal Shanmuganathan, India
Prayas Sharma, India
Bo Shen, Germany
Hang Shen, China

Xin Pu Shen, China
Dimitri O. Shepelsky, Ukraine
Jian Shi, China
Amin Shokrollahi, Australia
Suzanne M. Shontz, USA
Babak Shotorban, USA
Zhan Shu, Canada
Angelo Sifaleras, Greece
Nuno Simões, Portugal
Mehakpreet Singh, Ireland
Piyush Pratap Singh, India
Rajiv Singh, India
Seralathan Sivamani, India
S. Sivasankaran, Malaysia
Christos H. Skiadas, Greece
Konstantina Skouri, Greece
Neale R. Smith, Mexico
Bogdan Smolka, Poland
Delfim Soares Jr., Brazil
Alba Sofi, Italy
Francesco Soldovieri, Italy
Raffaele Solimene, Italy
Yang Song, Norway
Jussi Sopanen, Finland
Marco Spadini, Italy
Paolo Spagnolo, Italy
Ruben Specogna, Italy
Vasilios Spitas, Greece
Ivanka Stamova, USA
Rafał Stanisławski, Poland
Miladin Stefanović, Serbia
Salvatore Strano, Italy
Yakov Strelniker, Israel
Kangkang Sun, China
Qiuqin Sun, China
Shuaishuai Sun, Australia
Yanchao Sun, China
Zong-Yao Sun, China
Kumarasamy Suresh, India
Sergey A. Suslov, Australia
D.L. Suthar, Ethiopia
D.L. Suthar, Ethiopia
Andrzej Swierniak, Poland
Andras Szekrenyes, Hungary
Kumar K. Tamma, USA

Yong (Aaron) Tan, United Kingdom
Marco Antonio Taneco-Hernández, Mexico
Lu Tang, China
Tianyou Tao, China
Hafez Tari, USA
Alessandro Tasora, Italy
Sergio Teggi, Italy
Adriana del Carmen Téllez-Anguiano, Mexico
Ana C. Teodoro, Portugal
Efstathios E. Theotokoglou, Greece
Jing-Feng Tian, China
Alexander Timokha, Norway
Stefania Tomasiello, Italy
Gisella Tomasini, Italy
Isabella Torcicollo, Italy
Francesco Tornabene, Italy
Mariano Torrisi, Italy
Thang nguyen Trung, Vietnam
George Tsiatas, Greece
Le Anh Tuan, Vietnam
Nerio Tullini, Italy
Emilio Turco, Italy
Ilhan Tuzcu, USA
Efstratios Tzirtzilakis, Greece
FRANCISCO UREÑA, Spain
Filippo Ubertini, Italy
Mohammad Uddin, Australia
Mohammad Safi Ullah, Bangladesh
Serdar Ulubeyli, Turkey
Mati Ur Rahman, Pakistan
Panayiotis Vafeas, Greece
Giuseppe Vairo, Italy
Jesus Valdez-Resendiz, Mexico
Eusebio Valero, Spain
Stefano Valvano, Italy
Carlos-Renato Vázquez, Mexico
Martin Velasco Villa, Mexico
Franck J. Vernerey, USA
Georgios Veronis, USA
Vincenzo Vespri, Italy
Renato Vidoni, Italy
Venkatesh Vijayaraghavan, Australia
Anna Vila, Spain
Francisco R. Villatoro, Spain
Francesca Vipiana, Italy
Stanislav Vítek, Czech Republic
Jan Vorel, Czech Republic
Michael Vynnycky, Sweden
Mohammad W. Alomari, Jordan
Roman Wan-Wendner, Austria
Bingchang Wang, China
C. H. Wang, Taiwan
Dagang Wang, China
Guoqiang Wang, China
Huaiyu Wang, China
Hui Wang, China
J.G. Wang, China
Ji Wang, China
Kang-Jia Wang, China
Lei Wang, China
Qiang Wang, China
Qingling Wang, China
Weiwei Wang, China
Xinyu Wang, China
Yong Wang, China
Yung-Chung Wang, Taiwan
Zhenbo Wang, USA
Zhibo Wang, China
Waldemar T. Wójcik, Poland
Chi Wu, Australia
Qiuhong Wu, China
Yuqiang Wu, China
Zhibin Wu, China
Zhizheng Wu, China
Michalis Xenos, Greece
Hao Xiao, China
Xiao Ping Xie, China
Qingzheng Xu, China
Binghan Xue, China
Yi Xue, China
Joseph J. Yame, France
Chuanliang Yan, China
Xinggang Yan, United Kingdom
Hongtai Yang, China
Jixiang Yang, China
Mijia Yang, USA
Ray-Yeng Yang, Taiwan

Contents

Hindawi
Mathematical Problems in Engineering
Volume 2021, Article ID 5012496, 17 pages
https://doi.org/10.1155/2021/5012496

Review Article

Computational Image Encryption Techniques: A Comprehensive Review

Mandeep Kaur,[1] Surender Singh,[1] and Manjit Kaur[2]

[1] *Computer Science and Engineering Department, Chandigarh University, Mohali, India*
[2] *Computer Science Engineering, School of Engineering and Applied Sciences, Bennett University, Greater Noida, India*

Correspondence should be addressed to Manjit Kaur; manjit.kr@yahoo.com

Received 28 April 2020; Revised 27 June 2020; Accepted 30 June 2020; Published 19 July 2021

Academic Editor: Erivelton Geraldo Nepomuceno

Images contain very sensitive and confidential information. Because images play a significant role in many applications such as military communication, remote-sensing, and medical-imaging, therefore, it is necessary to protect sensitive and confidential information from unauthorized use and modification. To achieve this objective, encryption is one of the best methods among the information hiding methods. In recent years, many image encryption approaches are designed by the researchers. They use different concepts for image encryption to increase security. The main aim of this paper is to present a comprehensive review of the existing image encryption approaches. These approaches are categorized based on different concepts such as chaotic maps, DNA, compressive sensing, and optical image encryption. Comparisons are made among the existing approaches to access the various security parameters. Key performance metrics are also presented. The future scope of image encryption is also presented to encourage the research community.

1. Introduction

Due to advancements in technology, digital images are utilized in many applications such as medical imaging, remote sensing, and private conferencing. These images may contain confidential and sensitive information [1]. The transmission of these images over public networks is prone to several issues such as modification and unauthorized access. The leakage of sensitive information may raise military, national security, and discretionary issues. Moreover, when individuals wish to exchange images through a public network, it is necessary to assure their privacy. Therefore, images require security against different security attacks [2].

From the literature, it has been found that image encryption approaches can be utilized to provide security to these images. Image encryption is a procedure which converts plain image to an encrypted image by employing a secret key. The decryption process decrypts the cipher image into the original image by employing the secret key [3, 4]. Mainly, decryption operation is like encryption operation but applies in reverse order. The secret keys play a critical role in encryption. Because the security of the encryption approach is mainly dependent on it, two types of keys are utilized, namely, private key and public key [5, 6]. In the private key, the encryption and decryption processes use the same key to encrypt and decrypt the images. In the case of a public key, two keys are utilized, one key for encryption and one for decryption. In this, the encryption key is made public, but the decryption key is always kept private [7]. Figure 1 represents the block diagram of image encryption.

1.1. Basic Terms Utilized in Encryption.

(i) Plain image: it is the image that needs security while there is transmission over the public network. It is also known as the original or input image.

(ii) Cipher image or encrypted image: the plain image converted into a nonreadable form after encryption is called a cipher image.

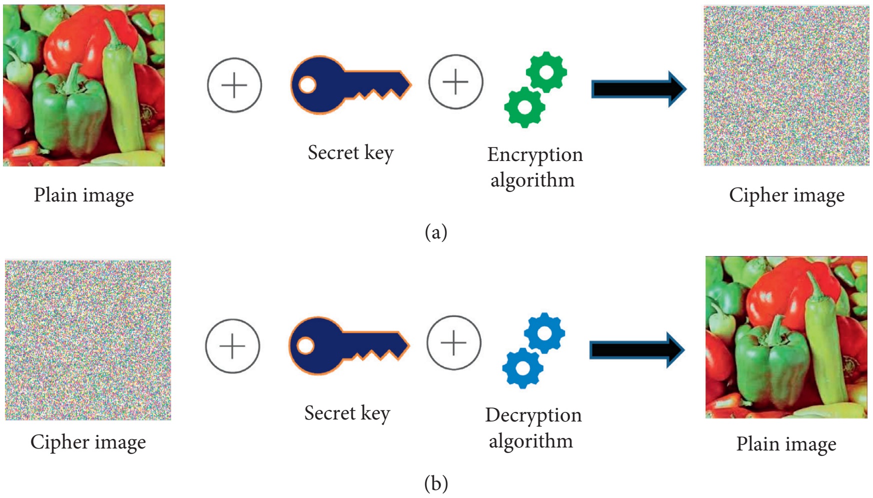


Figure 1: General framework of image encryption. (a) Encryption process at sender side. (b) Decryption process at receiver side.

(iii) Encryption: it is the process of converting a plain image into a cipher image utilizing an encryption approach and a secret key.

(iv) Decryption: at the receiver side, the cipher image is converted into a plain image utilizing a decryption approach and a secret key. This process is known as decryption.

(v) Key: the security of the encryption approach is mainly depending on the key. It can be numeric or alphanumeric. Both encryption and decryption need the key to performing their respective operations. Strong keys are always needed for better security of information.

1.2. Materials and Methods. Various image encryption approaches are designed so far. With time, researchers have also applied different types of concepts to increase the security of images. The traditional approaches such as DES, AES, and IDEA, have been obsolete in the case of images. Because the images have different properties as compared to text, many image encryption approaches are utilized in the last few decades; but in this study, we have considered only the last eight-year approaches (2013–2020), because we have found the application of diverse concepts in the area of information security.

Preferred reporting items for systematic reviews and meta-analyses (PRISMA) method is used in this study for getting the accurate results in order to summarize the existing work in image encryption field. The method consists of four phases: (i) identification, (ii) screening, (iii) eligibility, and (iv) inclusion that provide the accurate report for the analysis. By using the PRISMA method, the concluding outcome will be free of biases of the review studies; however, most of the reviews may be suffered from the selective outcome reports. In addition to this, number of sources can be utilized by giving the relevant Boolean queries for eliminating the articles that are not relevant to the study. The model begins from the step of identifying the sources of article, after that screening is performed by eliminating the replica and also the irrelevant articles by going through the titles and abstracts. Afterward, the articles left will be further screened by going through the full paper and all the articles that are not relevant to the study are excluded from the review studies.

In this study, five well-known databases have been selected for getting the relevant articles for performing the review including Wiley library, IEEE, Springer, Science-Direct, and Google scholar. The Boolean query that has been run on these databases are: Query: TITLE-ABS-KEY (“image ∗ ” AND “encrypt ∗ ” AND PUBYEAR 2015–2020). Based on the abovementioned query entered in five different databases, a total of 10446 articles were found for related articles. Using PRISMA method, 9523 articles were excluded based on titles and abstracts. Now the remaining 923 were again the screened and 397 articles were again removed from the study as they were either the conference articles or the duplicate ones. Now, the number comes to 526 out of which 348 articles are not relevant to the study as these are not having all the evaluation metrics that were the actual evaluation parameters of the study. Thereafter, 19 articles were again excluded as they were not written in English language. Then, the final number that comes after passing through various parameters was 159. The detailed analysis report of all these 159 articles with their outcomes is summarized. Figure 2 shows the flow chart for the database search of publications for systematic reviews.

1.3. Contribution. To the best of our knowledge, this is the first systematic literature review paper which has discussed the metaheuristics-based image encryption techniques. Beside this, we have also compared optical image encryption techniques which were ignored in the most of the existing review papers. Also, by reviewing the latest papers, we have evaluated various shortcomings of the recently published image encryption techniques. The main contributions of this paper are as follows:

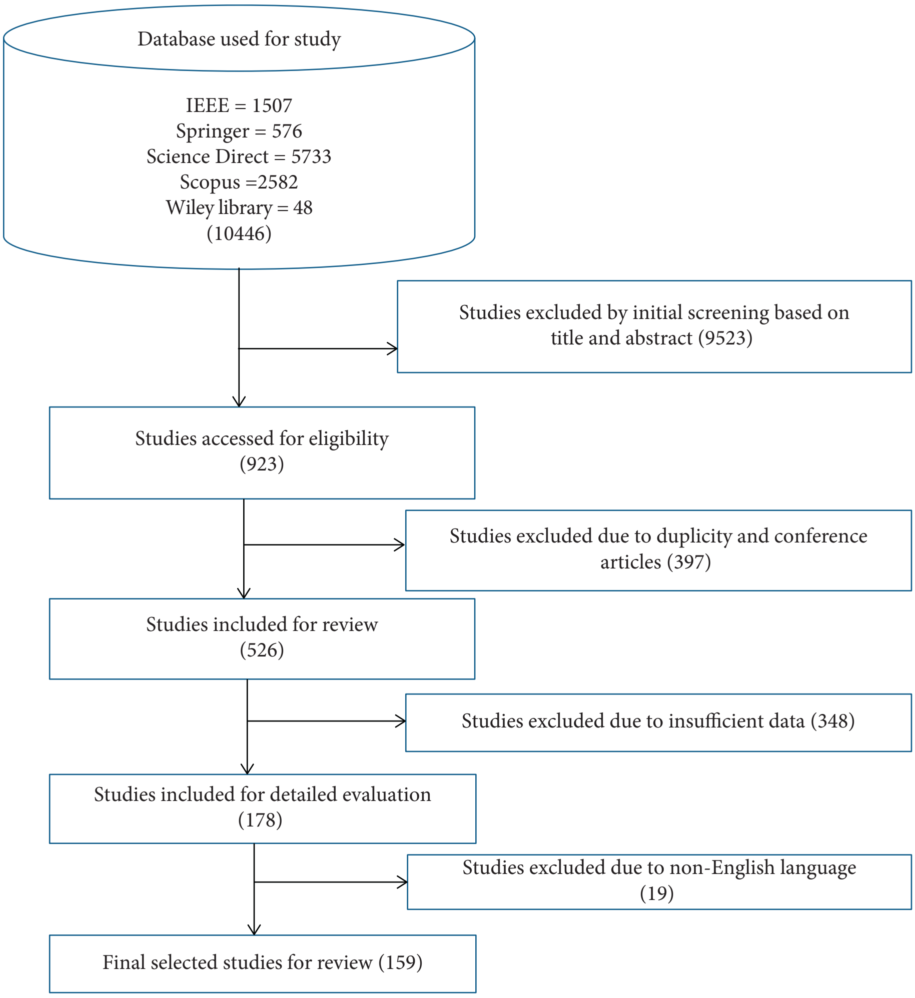

Database used for study
IEEE = 1507
Springer = 576
Science Direct = 5733
Scopus =2582
Wiley library = 48
(10446)
Studies excluded by initial screening based on title and abstract (9523)
Studies accessed for eligibility (923)
Studies excluded due to duplicity and conference articles (397)
Studies included for review (526)
Studies excluded due to insufficient data (348)
Studies included for detailed evaluation (178)
Studies excluded due to non-English language (19)
Final selected studies for review (159)

Figure 2: The flow chart for the database search of publications for systematic reviews.

(i) Initially, the existing image encryption approaches are categorized based upon various concepts such as chaos, DNA, compressive sensing, and optical.

(ii) Various metrics utilized to compute the performance of image encryption approaches are also discussed.

(iii) Comparisons are made among the existing approaches to access the various security parameters.

(iv) Finally, the future scope of image encryption is also presented to encourage the research community.

The remaining paper is organized as follows: the performance metrics are discussed in Section 2. In Section 3, various categories of the existing image encryption approaches are discussed. Section 4 presents the comparative analyses among the existing image encryption approaches. Future directions are discussed in Section 5. Section 6 concludes the paper.

2. Evaluation Parameters

Evaluation parameters are utilized to assess the performance of image encryption. There are many security attacks performed by the attackers to break the encryption approach as well as to find the key. Attackers mainly utilize the cryptanalysis to study the encryption approaches [8, 9]. Therefore, it is necessary to hide the statistics of plaintext and the secret key. The strength of image encryption can be evaluated utilizing security and quality analyses. The quality analysis assesses the image quality of decrypted image utilizing peak signal-to-noise ratio, mean square error, etc. The security analyses include statistical analysis, differential analysis, and key analysis.

Statistical properties of the generated cipher image can be tested utilizing entropy, correlation coefficient, and histogram analysis. It is required that the encryption approaches do not provide statistical details of the plain image. Sometimes, we assume that an attacker obtains the details of the encryption approach without knowing the key. In other words, the key is considered to be embedded in the encryption approach. Then, the attacker supplies an image to the encryption approach and gets a corresponding cipher image. Thereafter, he made small changes in the same image and got another cipher image. Then, he tries to find the similarity between two ciphered images to break the encryption approach. It means that the encryption approach is required to be sensitive to small changes towards the plain image. It is assessed utilizing differential analysis. In this, unified average changing intensity and number of pixel change rate metrics can be utilized for the same.

As we know that the performance of the image encryption approach is mainly dependent on the key, therefore, it should be large enough, so that it cannot be guessed easily. Secondly, it should be sensitive to small changes. The encryption approach should generate a totally different cipher image, even if the only one-bit difference is present in two keys. While there is transmission over a noisy channel, the cipher image may get affected. Therefore, the encryption approach should be robust against noise attacks. The receiver should be able to recover the original image. In real-time applications, the speed of encryption approaches matters a lot. It is always desirable that the encryption approach should be fast. Table 1 defines the various parameters utilized for image encryption performance evaluation. It also presents the desirable expectation of every parameter.

3. Image Encryption Approaches

Different types of image encryption approaches are designed so far. By reviewing the literature, we have divided it into different types such as spatial, transform, optical, and compressive sensing based image encryption approaches. Figure 3 demonstrates the categories of image encryption approaches. In the preceding subsection, these approaches are discussed and analyzed utilizing evaluation metrics. These parameters are KA, NPCR, HA, UACI, IE, CC, and NA. In comparisons, ✓ and ✗ symbols are utilized to represent whether the given approach has considered the respective metric and not, respectively.

3.1. Image Encryption in Spatial Domain. The approaches that are directly manipulating the pixels of the image are considered as spatial domain approaches. The various spatial domain-based image encryption are present in the literature. But we have considered the most famous approaches such as chaotic-based, elliptic curve-based, fuzzy-based, DNA, and Metaheuristics-based approaches.

3.1.1. Chaos-Based Image Encryption Approaches. Chaotic maps have great significance in the field of encryption. These maps generate random numbers that are utilized as secret keys in encryption [17]. The reason is its properties such as dynamic and deterministic nature, sensitive to initial conditions, and ergodicity. Different types of chaotic maps are utilized so far. But these are mainly divided as one-dimensional and higher-dimensional chaotic maps. Chaotic maps help in performing the confusion and diffusion operations in the encryption process [18]. Figure 4 demonstrates the diagrammatic flow of the chaotic maps in the image encryption approach.

Chen et al. [19] developed an image encryption approach utilizing a 2D sine map and Chebyshev map. It designed an antidegradation universal approach for chaotic maps, which improves the performance even on low-accuracy devices. Xuejing and Zihui [20] proposed image encryption based on spatiotemporal chaotic map and DNA encoding. Firstly, a plain image is changed into three DNA matrices dependent on a random encoding rule; afterward, DNA resultant is joined into a modern matrix. Then, it is permutated by the ascent matrix to generate the ciphered image. Wang et al. [21] utilized coupled map lattices (CML) and the DNA approach to encrypt the images. Ismail et al. [22] examined a new lossless image encryption system that was based on fractional-order and double-humped logistic maps.

Wu et al. [23] designed an encryption approach utilizing a 2D discrete wavelet transform and hyperchaotic system for color images. Chai et al. [24] designed an encryption approach for color images utilizing a 4D memristive hyperchaotic map with genetic recombination. Luo et al. [25] developed an image encryption approach based on quantum coding and hyperchaos system. Kumar Patro and Acharya [26] proposed an image encryption approach utilizing a piece-wise linear chaotic map (PWLCM). In this, a rotating permutation is applied row-wise and column-wise. At last, it applies a diffusion operation on the row, column, and block to generate the ciphered image.

Feng et al. [27] utilized a discrete logarithm and a memristive chaotic system to encrypt the images. Wang and Gao [28] developed an image encryption strategy based on matrix semitensor. Hyperchaotic Lorenz map is also utilized to generate random numbers. Hua and Zhou [29] designed an approach for encrypting the images that provide excellent effects against differential and statistical attacks. Image filtering idea is utilized in image encryption to enhance the security of encryption. Gan et al. [30] implemented an image encryption approach based on 3D bit-plane confusion. Lu et al. [31] proposed an image encryption approach based on chaotic map and S-box. The discrete compound chaotic map was designed in this approach. S-box is also constructed utilizing logistic-sine system.

Deng and Zong [32] presented a binary image encryption approach based on chaotic mapping. The authors hypothetically examined the approach and figured out that the approach did not need to have the earlier information on the orbital distribution and one can pick out any chaotic model. Patro et al. [33] developed a color image encryption approach that overcomes the drawbacks of execution block-level dispersion processes in arbitrary sized images. Wang et al. [34] implemented an image encryption approach utilizing logistic-dynamic mixed linear-nonlinear coupled map lattices.

Ye et al. [35] utilized a memristive chaotic map to generate the secret keys to perform image encryption. Liu et al. [36] designed a fast image encryption approach derived from a sine map and the iterative chaotic map with infinite collapse based on a closed-loop modulation coupling model. Cao et al. [37] designed an image encryption approach for medical images by utilizing edge maps. It consists of three parts: bit-plane decomposition, random number generator, and permutation. Chai [38] designed a bit-level Brownian motion and 1D chaotic framework for encrypting the digital images. Table 2 demonstrates the comparison among the exiting chaos-based image encryption approaches. It can be

Table 1: Image encryption evaluation parameters.

Evaluation parameter	Abbr.	Expectation
Key space analysis [3]	KA	Large key size (more than 2^{100})
Histogram analysis [1]	HA	Pixels should uniformly distribute
Information entropy [10]	IE	Equal to 8, for a 256-gray level image
Noise attack [11]	NA	Ought to be safe to noise attacks
Correlation coefficient [12]	CC	Near to 0
Mean squared error [13]	MSE	Minimum
Peak signal-to-noise ratio [2, 14]	PSNR	Low between encrypted and actual images; high between the ciphered and decrypted images
Execution time [2]	ET	Minimum for practical implementation
Number of pixel change rate [15]	NPCR	Maximum value (according to critical analysis, it should be near to 99.609% or more)
Unified average changing intensity [16]	UACI	Minimum value (according to critical analysis, it should be near to 33.464% or more)

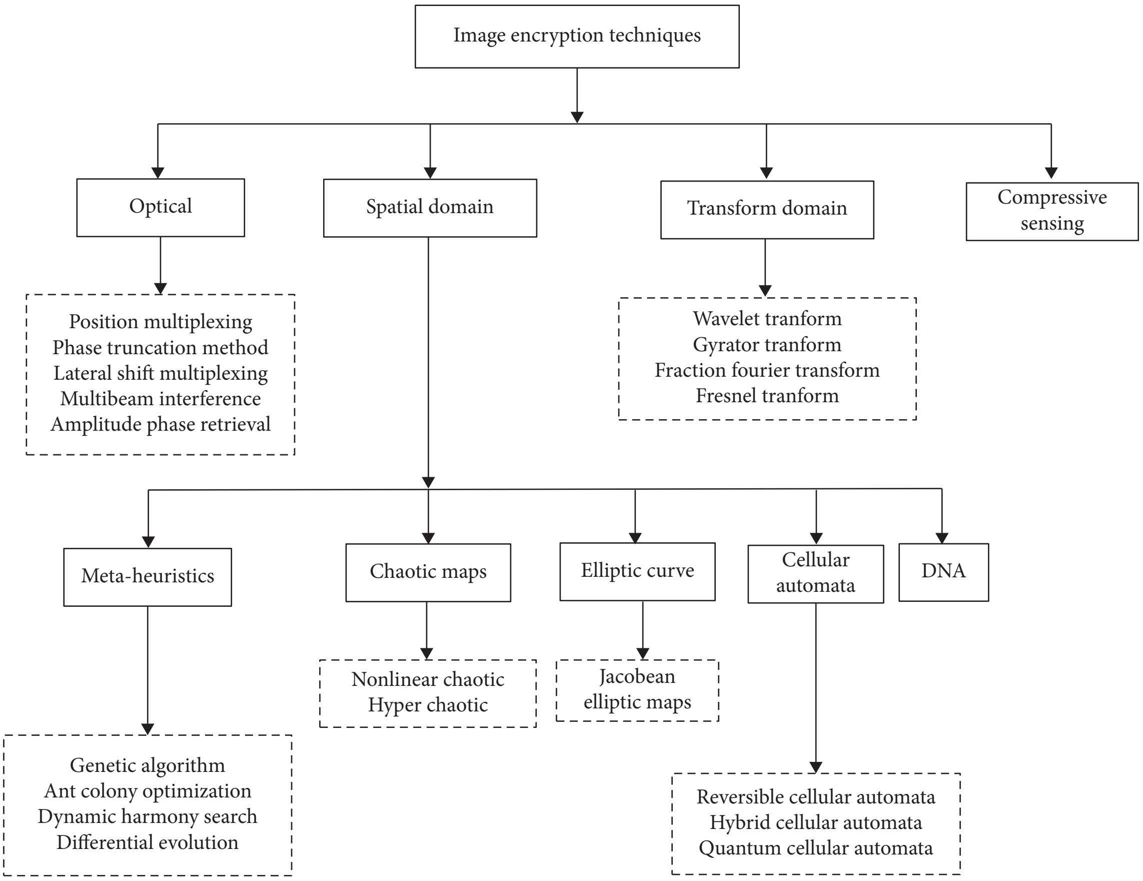


Figure 3: Categorization of image encryption approaches.

seen that most of the approaches do not satisfy all the security parameters. Therefore, it is still an open area for research.

3.1.2. Elliptic Curve-Based Image Encryption. Elliptic curve cryptography (ECC) works on the least amount of memory with the small key size [39]. Figure 5 demonstrates the use of elliptic curve in image encryption. The color image is initially compressed and changed into gray scale. Then, encryption is done by utilizing elliptic curve, 3D Lorenz chaotic map, and 4D Arnold cat map [40].Hayat and Azam [41] developed an approach based on pseudorandom numbers and substitution boxes for encrypting a digital image by utilizing an elliptic curve. Luo et al. [42] presented the asymmetric image encryption approach which depends on chaotic theory and the elliptic curve ElGamal (EC-ElGamal) cryptography. Banik et al. [43] discovered a medical image encryption approach based on Mersenne Twister pseudorandom number generator and elliptic curve analog ElGamal cryptosystem. The proposed approach enlivens the encryption time just as take care of the issue of information

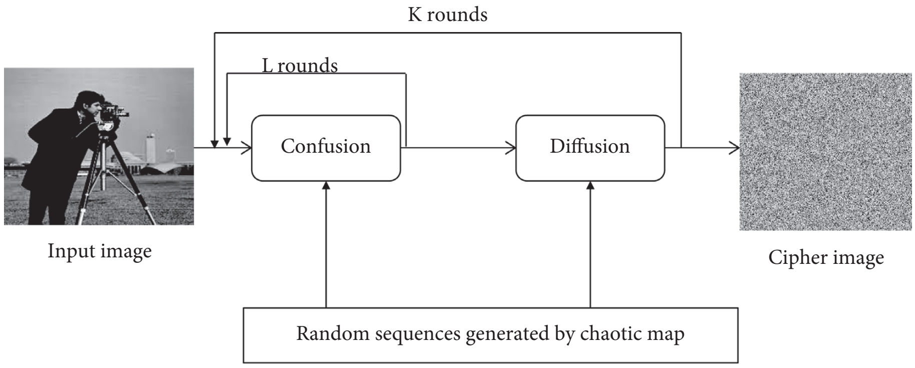


Figure 4: Working of chaotic map-based image encryption.

Table 2: Comparison of various chaotic-based image encryption approaches.

Ref.	Technique	HA	KA	IE	NA	NPCR	UACI	CC
[19]	Logistic map	✓	✓	✓	✓	✓	✓	✗
[20]	DNA and chaotic system	✓	✓	✓	✓	✓	✓	✗
[21]	Chaotic-based DNA	✓	✓	✓	✓	✓	✓	✗
[22]	Edge detection and chaotic map	✗	✗	✓	✓	✓	✓	✗
[23]	6D hyperchaotic	✓	✓	✓	✓	✓	✗	✓
[24]	Hyperchaotic system	✓	✓	✓	✓	✓	✗	✓
[25]	Hyperchaotic with quantum coding	✓	✓	✓	✓	✓	✓	✓
[26]	PWLCM system	✓	✓	✓	✓	✓	✓	✓
[27]	Standard memristive chaotic system	✓	✓	✓	✓	✓	✓	✓
[28]	Semitensor product theory	✗	✗	✓	✓	✓	✗	✓
[30]	3D bit-plane permutation	✓	✓	✓	✓	✓	✗	✓
[32]	Chaotic map	✗	✗	✓	✓	✓	✗	✓
[33]	Block level diffusion operation	✓	✓	✓	✓	✓	✗	✗
[34]	Coupled map lattices	✓	✓	✓	✓	✓	✓	✓
[35]	Mixed memristive chaotic circuit	✓	✓	✗	✓	✓	✓	✓
[36]	Chaotic map	✓	✓	✓	✓	✗	✗	✗
[37]	Edge maps	✓	✓	✗		✗	✗	✗
[38]	Bit level Brownian motion	✗	✗	✓	✓	✓	✗	✗

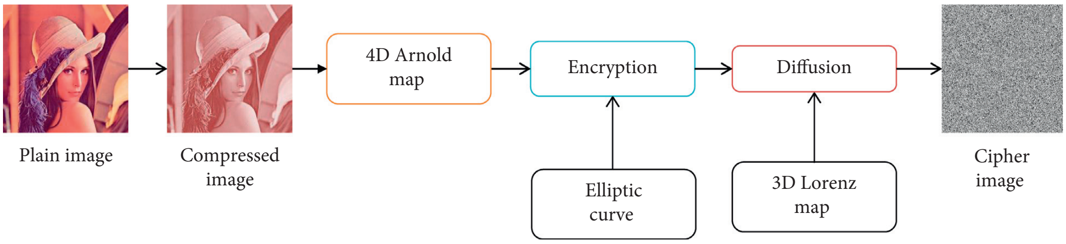


Figure 5: Chaotic map and elliptic curve-based image encryption.

extension related with ElGamal cryptosystem. Reyad and Kotulski [44] studied an image encryption approach which depends on computational tasks (such as add, double, and multiply) that depend on ECC.

Kumar et al. [45] implemented an image encryption approach utilizing ECC and DNA encoding. The approach initially encodes the RGB image utilizing DNA encoding. Thereafter, the elliptic curve Diffie–Hellman encryption (ECDHE) is utilized to perform encryption. Zhang and Wang [46] presented an improved ECC-based image encryption approach. The Diffie–Hellman approach is utilized to generate the secret key. The chaotic map is also applied in the combination with ECC to perform permutation and diffusion. Laiphrakpam and Khumanthem [47] developed an image encryption approach based on a chaotic framework and elliptic curve over a limited field.

Dawahdeh et al. [48] developed an image encryption approach by combining ECC with Hill cipher (ECCHC). Toughi et al. [39] implemented image encryption by utilizing an elliptic curve to obtain a series of random numbers established on curves. Liu et al. [49] designed a Menezes–Vanstone elliptic curve cryptosystem. In this, the 2D fractional triangle function is utilized with a discrete chaotic map. Wu et al. [40] proposed a color image encryption approach based on chaotic systems and elliptic curve ElGamal approach. Firstly, the original image is compressed and then the compressed image is encrypted by utilizing the improved 4D cat map.

Nagaraj et al. [50] combined the elliptic curve and magic matrix operation to encrypt the images. The input image lies on the points on the elliptic curve utilizing the transform approach. The image is decomposed into information matrices. Every single pixel of an image is permuted by the magic matrix. At last, each pixel is diffutilized to produce a cipher image by utilizing ECC. Table 3 demonstrates the comparison among various ECC-based image encryption approaches. It is observed that there exists no such approach which has considered all parameters.

3.1.3. Cellular Automata-Based Image Encryption Approaches. Cellular automata have been widely used in image encryption as a pseudorandom generator. These models are complex which have a degree of efficiency and robustness. Cellular automata use rules to produce random sequences. Due to the properties of cellular automata such as parallelism and easy and simple hardware structure, it is significant for encryption approaches [51]. Figure 6 demonstrates the general framework of cellular automata-based image encryption. The confusion and diffusion operations are performed by utilizing the key generator and cellular automata, respectively. Cellular automata generate random sequences to diffuse the pixel values of the image [7].

Khan et al. [52] designed a hybrid image encryption approach by merging a logistic sine system with 2D cellular automata. Mondal et al. [51] implemented an image encryption approach that is exceptionally secure based on a chaotic skew tent map and cellular automata. Zhang et al. [53] utilized 1D chaotic map for generating the pseudorandom number. To perform permutation-substitution, bit-level cellular automata are utilized to generate an encrypted image.

Su et al. [54] designed a deterministic image encryption approach based on reversible cellular automata (DERCA). This approach addresses the problem of similarity search on encrypted images. It finds one-to-many mapping between histograms of encrypted and original images. Ramírez et al. [55] presented a partial image encryption strategy depending on the cellular machine rule. The security examination demonstrates that this cryptosystem is impervious to various tests. Wang et al. [56] evaluated the image cryptosystem on the two-dimensional partitioned cellular automaton.

Yaghouti Niyat et al. [57] implemented a nonuniform cellular automata system to illuminate the major drawbacks of cellular automata in cryptography. It incorporates a predetermined number of inversion rules. Chai et al. [58] utilized a memristive hyperchaotic system, cellular automata, and DNA sequence operations to encrypt the images. Wei et al. [59] designed a double-color image-enciphering method depending on off-axis Fourier holography and maximum length cellular automata (MLCA). The color image is separated into red, green, and blue, three channels, and all channels are autonomously scrambled by utilizing MLCA.

Chen et al. [60] developed an encryption and compression approach based on a combination of Kronecker CS (KCS) with elementary cellular automata (ECA). Souyah and Faraoun [61] evaluated the symmetric approach for enciphering digital images by combining chaos and cellular automata (CA) under the situation of one round encryption or decryption. Tralic and Grgic [62] presented an approach for image encryption based on a 2D cellular automaton and pixel division. Application of the balanced 2D cellular automata with amplified Moore neighborhood for each degree of pseudorandom key-image makes it different from existing approaches.

Yang et al. [63] proposed an encryption approach for grayscale images based on 1D quantum cellular automata. Murugan et al. [64] designed an image encryption approach by combining chaos and cellular automata. Logistic map and Conway's game-of-life cellular automata are utilized in the permutation process and the Chebyshev map and Lorenz equation are utilized for diffusion.

Souyah and Faraoun [65] discussed the approach for image encryption that combines the image's quadtree decomposition approach with reversible memory cellular automata mechanism. Enayatifar et al. [66] designed an encryption approach by utilizing a Tinkerbell hybrid model of a chaotic map, deoxyribonucleic acid (DNA), and cellular automata. Table 4 demonstrates the comparison of various cellular automata-based image encryption approaches. It is found that no approach has utilized every performance metric.

3.1.4. DNA-Based Image Encryption Approaches. Deoxyribonucleic acid (DNA) cryptography has become very popular due to its properties such as massive parallelism, huge storage, and ultra-low power consumption. The complementary rules of DNA are utilized to perform encoding and decoding [73]. Figure 7 demonstrates the block diagram of DNA-based image encryption process. Firstly, the color image is decomposed into three channels red (R), blue (B), and green (G). After that, DNA encoding and XOR operations are utilized to encode the channels. A chaotic map is utilized to scramble matrices. Finally, three R, G, and B channels are combined to obtain the cipher image [74].

Wu et al. [75] designed the color image encryption approach by utilizing three one-dimensional chaotic maps with DNA sequences. The original image and keystream are changed into matrices utilizing DNA operation. Complementary and XOR encoding operations are connected to rearrange the matrices. The matrices are decayed into blocks and rearrange them arbitrarily. DNA addition and XOR encoding operations are performed on these scrambled matrices to get the cipher image. Mondal and Mandal [76] utilized two pseudorandom number sequences with DNA for the encryption of sensitive images. Wu et al. [77] implemented an encryption approach for color images

Table 3: Comparison among various elliptic curve-based image encryption approaches.

Ref.	Technique	HA	KA	IE	NA	NPCR	UACI	CA	PSNR	MSE
[41]	Elliptic curve cryptography	✓	✓	✓	✓	✓	✓	✓	✗	✗
[42]	Elliptic with EIGamal encryption	✓	✓	✓	✓	✓	✓	✓	✗	✓
[43]	Elliptic curve with Mersenne Twister	✓	✓	✓	✓	✓	✗	✓	✓	✗
[39]	Elliptic curve and AES	✓	✓	✗	✗	✓	✗	✓	✗	✗
[45]	DNA with elliptic curve	✗	✗	✓	✓	✗	✗	✓	✓	✓
[40]	Elliptic curve EIGamal encryption	✓	✓	✓	✓	✓	✗	✓	✓	✗
[46]	Elliptic curve with ECC	✓	✓	✓	✓	✓	✗	✓	✗	✗
[47]	Elliptic curve over finite field	✗	✗	✓	✓	✓	✓	✗	✓	✗
[48]	Elliptic with Hill cipher	✗	✓	✗	✗	✓	✗	✗	✓	✗
[49]	Menezes–Vanstone elliptic curve	✓	✓	✗	✓	✓	✗	✗	✗	✗

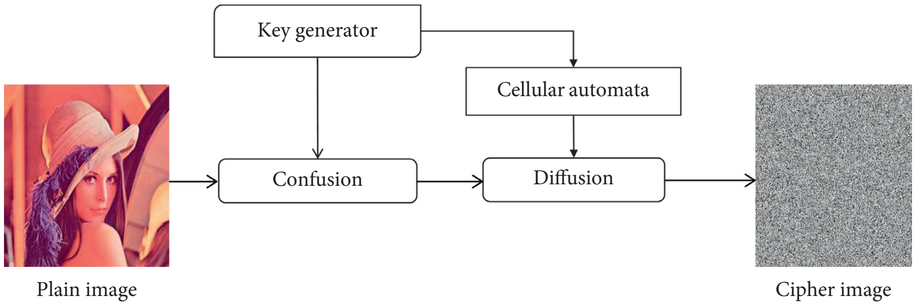


Figure 6: General framework of cellular automata-based image encryption.

Table 4: Comparison of various cellular automata-based image encryption approaches.

Ref.	Technique	KA	NPCR	UACI	IE	CA	HA	PSNR	MSE
[54]	Reversible cellular automata	✓	✓	✓	✗	✓	✓	✗	✗
[55]	Cellular automata	✗	✗	✗	✓	✗	✗	✗	✗
[67]	Logistic mapped convolution	✗	✗	✓	✓	✓	✗	✓	✓
[56]	Partitioned cellular automata	✗	✗	✗	✗	✗	✓	✗	✗
[57]	Hybrid chaotic system	✗	✗	✓	✓	✓	✓	✗	✗
[58]	Memristive hyperchaotic	✓	✓	✓	✓	✗	✗	✓	✓
[59]	Holography	✗	✗	✓	✓	✗	✓	✗	✗
[60]	Kronecker compressed sensing	✓	✓	✓	✓	✓	✓	✓	✗
[61]	Weighted histogram	✗	✗	✓	✓	✓	✓	✗	✗
[62]	Pixel separation	✓	✓	✓	✓	✓	✓	✗	✗
[68]	8-layer cellular automata	✓	✓	✓	✓	✓	✓	✗	✗
[63]	Quantum cellular automata	✓	✓	✓	✗	✗	✓	✓	✓
[64]	Conway’s cellular automata	✓	✓	✓	✗	✓	✓	✗	✗
[65]	Quadtree decomposition	✗	✗	✗	✓	✓	✗	✗	✗
[69]	DNA and cellular automata	✓	✓	✓	✗	✗	✗	✗	✗
[66]	Hybrid DNA	✓	✓	✓	✓	✓	✗	✗	✗
[70]	3D cellular automata	✓	✓	✓	✓	✗	✗	✗	✗
[51]	Chaotic skew tent map	✓	✓	✓	✓	✓	✓	✓	✓
[71]	4^{th} order cellular automata	✓	✓	✓	✓	✓	✗	✗	✗
[72]	Linear cellular chaos-based automata	✗	✗	✓	✗	✗	✗	✗	✗
[52]	FSM-based DNA	✗	✗	✓	✓	✗	✓	✗	✗
[53]	Coupled logistic Bernoulli map	✓	✓	✗	✗	✗	✗	✗	✗

utilizing DNA. This approach contains four steps: key generation, DNA sequence for permutation, DNA sequence for diffusion, and diffusion process for pixel level.

Li et al. [78] enhanced the security of image encryption utilizing complex chaotic maps and quaternary coding in DNA. Wang et al. [79] implemented a DNA sequence and CML-based color image encryption approach. Initially, R, G, and B components of the color image are utilized to form the matrix. DNA encoding operation is utilized further to encrypt rows and columns of the matrix to get an encrypted image. Nematzadeh et al. [80] developed a DNA and binary search tree- (DNA-BST-) based image encryption approach. Rehman et al. [15] designed a block cipher image encryption approach for gray images based on DNA complementary

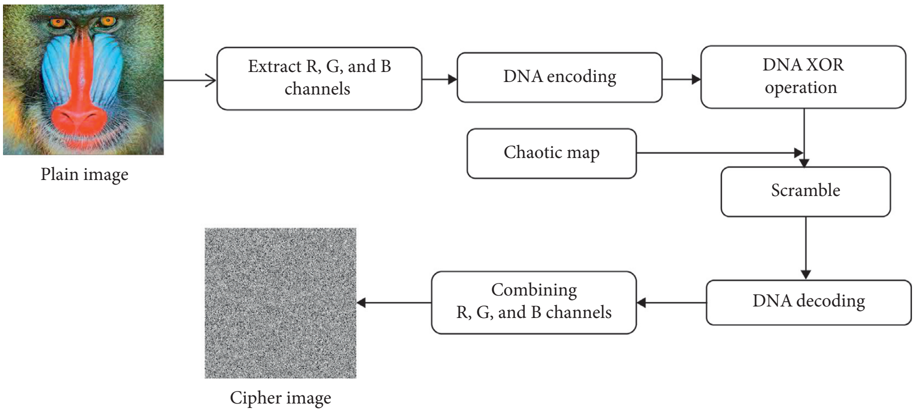


Figure 7: DNA-based image encryption.

rules and piece-wise linear chaotic map. Jain and Rajpal [81] implemented an image encryption approach utilizing DNA and a 2D chaotic map. In this, the input image encoded by utilizing DNA operation as a resultant matrix was generated. Resultant matrix permuted by utilizing a 2D chaotic map followed by DNA decoding operation to get the cipher image.

Wang and Liu [82] designed an image encryption approach based on DNA operation and chaos mapping. In this approach, eight DNA rules are applied to encode the rows of the plain image. The selection of DNA rule is done through a chaotic map. Zhang et al. [83] designed image encryption utilizing permutation and diffusion based on DNA encoding and Feistel network. Chai et al. [84] utilized a chaotic system and DNA to design an image encryption approach. In this approach, permutation and diffusion are done by the DNA matrix which makes it different from the traditional approach. Zhang et al. [85] implemented an image encryption approach utilizing DNA encoding, Lorenz chaotic approach, and Chan's hyperchaotic approach. DNA encoding rules are utilized randomly to improve the security of the encryption process.

Liu et al. [86] designed an image encryption approach by utilizing dynamic S-boxes calm of DNA and chaotic system. Dynamic S-box calm of DNA encoding operation is utilized to confuse the pixel values of the image for the encryption process. Norouzi and Mirzakuchaki [87] presented an image encryption by utilizing DNA sequence operation and cellular neural network. Liu et al. [88] presented the color image encryption approach based on DNA masking and a hybrid model of multidirectional circular permutation. The initial position of pixels of an image is rotated by circular permutation, and by DNA sequence operation, the values of the pixel are substituted to attain an encrypted image. Zhang et al. [89] designed an image encryption approach based on bit permutation and dynamic DNA encoding. This approach is exceptionally successful against noise and known-plain-text attacks. Table 5 demonstrates the comparison of various DNA-based image encryption approaches. It demonstrates that there exist only two approaches that have utilized every performance metric.

Table 5: Comparison of various DNA-based image encryption approaches.

Ref.	KA	NPCR	UACI	IE	CA	HA	PSNR	MSE
[90]	✓	✓	✓	✓	✓	✓	✗	✗
[76]	✓	✓	✓	✗	✓	✗	✗	✗
[15]	✓	✓	✓	✓	✓	✓	✗	✗
[81]	✓	✓	✗	✓	✗	✓	✗	✗
[77]	✓	✓	✗	✓	✓	✓	✗	✗
[73]	✓	✓	✓	✓	✓	✓	✓	✓
[79]	✓	✓	✓	✗	✓	✓	✗	✗
[80]	✓	✓	✗	✓	✓	✓	✗	✗
[82]	✗	✗	✗	✓	✓	✗	✗	✗
[91]	✓	✓	✓	✓	✓	✓	✓	✓
[83]	✓	✓	✗	✓	✓	✓	✗	✗
[84]	✓	✓	✓	✓	✗	✓	✗	✗
[92]	✓	✓	✗	✗	✗	✗	✗	✗
[93]	✓	✓	✗	✓	✓	✓	✗	✗
[85]	✓	✗	✗	✓	✗	✓	✗	✗
[94]	✓	✓	✓	✓	✓	✓	✓	✗
[86]	✓	✓	✗	✓	✓	✗	✓	✓
[95]	✓	✓	✓	✓	✓	✓	✓	✓
[87]	✓	✓	✓	✓	✓	✓	✓	✓
[78]	✗	✗	✓	✓	✓	✓	✗	✗
[88]	✓	✓	✗	✓	✓	✓	✗	✗
[89]	✗	✗	✓	✓	✓	✓	✗	✗

3.1.5. Metaheuristics-Based Image Encryption Approaches. Metaheuristic approaches are mainly utilized in a situation where we need optimized results. Recently, the use of such approaches has been increased in the image encryption. There are two aspects to use metaheuristic approaches in image encryption: (a) generate multiple cipher images and then select optimized one and (b) optimize the initial parameters of chaotic maps to generate efficient keys. Researchers have implemented the image encryption approaches based on metaheuristic approaches, considering different aspects.

Medical images are encrypted by [96] utilizing coupled map lattices and modified genetic algorithm. The genetic algorithm selects the encrypted image which has high entropy. The algorithm approach is utilized by [97] to generate the optimized key. ECC is utilized to perform the encryption process utilizing an optimized key. Kaur and Kumar [98] utilized differential evolution to optimize the beta-chaotic map to generate efficient secret keys. They further utilized a nondominated sorting genetic algorithm (NSGA) [99] to optimize the initial parameters of the intertwining logistic map.

Genetic approach is applied by [5] to optimize the beta chaotic map. To generate the efficient keys, Nematzadeh et al. [100] utilized NSGA-II for intertwining logistic map. Adaptive differential evolution is utilized by [4] to optimize the initial parameters of the Lorenz chaotic map. In [101], medical gray images are optimized utilizing a genetic algorithm. Talarposhti et al. [102] proposed an image encryption approach based on a dynamic harmony search (DHS) combined with a chaotic map.

Memetic differential evolution is applied by [103] to generate the optimized cipher image. Pareto evolutionary algorithm-II was utilized by [13] to obtain the optimized encrypted image. In [104], NSGA-III is utilized to generate the optimal cipher image by optimizing the hyperchaotic map. In [6], a 5D chaotic map is optimized by combining NSGA and local chaotic maps. Table 6 demonstrates the comparison of various metaheuristic image encryption approaches. Most of the approaches have not applied all the performance metrics.

Table 6: Comparison of various metaheuristic image encryption approaches.

Ref.	NPCR	UACI	KA	HA	IE	CA
[11]	✓	✓	✗	✓	✓	✗
[105]	✓	✓	✓	✗	✓	✓
[102]	✗	✗	✓	✗	✓	✓
[106]	✓	✓	✗	✓	✗	✓
[100]	✓	✓	✓	✓	✓	✓
[96]	✓	✓	✗	✗	✗	✓
[107]	✓	✓	✓	✓	✗	✓
[108]	✗	✗	✓	✓	✗	✓
[109]	✓	✓	✓	✓	✗	✓
[110]	✓	✓	✓	✓	✓	✗
[111]	✓	✓	✗	✓	✓	✓

3.2. Compressive Sensing-Based Image Encryption Approaches. Compressive sensing can perform compression as well as encryption at the same time [112]. It uses a measurement matrix and reconstruction approach to perform the same. The measurement matrix is utilized to perform the compression. At the same time, when the measurement matrix is utilized as a secret key between the sender and receiver, it works as a cryptosystem [113]. Various image encryption approaches based on compressive sensing have been proposed by the researchers. Some of the approaches are discussed in this section.

Ponuma and Amutha [114] utilized chaotic compressive sensing to encrypt the color images. It also performed compression at the same time. The chaotic measurement matrix constructed utilizing one-dimensional chaotic map is utilized. This approach is further enhanced in [115] by making it visually meaningful. Wavelet transform is utilized to hide the encrypted image into a cover image. Shao et al. [116] utilized analog-digital hybrid electro-optic chaotic sources and compressive sensing to encrypt the images. Zhu et al. [117] designed a hybrid approach for image compression and encryption by utilizing block compressive sensing. The nonuniform sampling approach is utilized to improve the efficiency of compression.

Shen et al. [118] utilized nonuniform quantization and compressive sensing to encrypt the images. It reduces the data precision in cipher images while evaluating the true compression ratio (CR). Jiang et al. [119] proposed a color image encryption approach based on compressive sensing and multi-image cross pixel scrambling approach. A discrete wavelet transform is also applied to process the subimages. Yao et al. [120] implemented an image encryption approach depending upon improved two-dimensional closed-loop modulation coupling approach. The approach was combined with compressive sensing to build a fast image encryption process.

Xu et al. [121] implemented an image encryption strategy utilizing compressive sensing, wavelet transform, and chaotic map. Gong et al. [122] utilized compressive sensing and RSA approach for optical image compression and encryption. To sample the initial image, the optical compressive imaging is utilized. Zhu and Zhu [123] encrypted the images based on compressive sensing and cyclic shift. Sparse transform and Gauss matrix is applied to perform compression. Wang et al. [124] proposed an image encryption strategy based on parallel compressive sensing. The logistic-tent system and 3D cat map are utilized to generate the measurement matrices.

Luo et al. [125] designed compression and encryption strategy for images based on compressive sensing and Haar wavelet. Ponuma and Amutha [126] utilized sparse coding and compressive sensing to encrypt the images. Sparse coding is utilized to discover the sparse representation of images as a straight combination of iotas from an overcomplete fixed dictionary. Song et al. [127] implemented an image encryption based on entropy coding and compressive sensing.

Han et al. [128] proposed a self-adaptive double-color image encryption approach. In this, each RGB color element of two input images is initially compressed and encrypted by 2D compressive sensing. The complex image is re-encrypted by self-adaptive random phase encoding and discrete fractional random transform (DFrRT) to get the final scrambled image. Zhang et al. [129] designed hybrid image compression and encryption approaches by exploring compressive sensing and Fibonacci-Lucas transform with the advantages of one-dimensional chaotic system. Pan et al. [130] utilized block compressive sensing to design an image encryption approach. The original image is separated into blocks and then each block is rendered sparse. The crisscross encryption strategy is utilized to encrypt pixel positions in all the blocks, and subsequently, dimension reduction is taken by compressive sensing. Table 7 demonstrates the comparison of various compressive sensing encryption approaches. It demonstrates that there is no such approach that has considered all the performance metrics.

Table 7: Comparison of various compressive sensing encryption approaches.

Ref.	HA	KA	IE	NA	NPCR	UACI	CA	PNSR	MSE
[114]	✗	✗	✓	✓	✓	✗	✓	✓	✓
[115]	✗	✗	✓	✗	✗	✓	✓	✓	✓
[116]	✗	✗	✓	✓	✗	✗	✓	✓	✗
[117]	✓	✓	✓	✓	✗	✗	✗	✓	✓
[118]	✗	✗	✓	✓	✓	✗	✓	✓	✗
[119]	✗	✗	✓	✓	✗	✓	✓	✗	✓
[120]	✗	✗	✗	✓	✗	✗	✓	✓	✓
[121]	✓	✓	✓	✓	✓	✓	✓	✓	✗
[123]	✓	✓	✓	✓	✓	✗	✗	✗	✗
[124]	✗	✗	✓	✓	✓	✓	✓	✓	✗
[125]	✗	✗	✓	✓	✗	✓	✓	✓	✗
[126]	✓	✓	✓	✓	✓	✗	✓	✓	✗
[127]	✗	✗	✗	✓	✓	✗	✗	✓	✗
[12]	✗	✗	✓	✓	✗	✓	✓	✓	✓
[128]	✗	✗	✗	✗	✓	✗	✓	✓	✓
[129]	✓	✓	✓	✗	✓	✓	✗	✗	✗
[130]	✗	✗	✓	✓	✗	✗	✗	✓	✗

3.3. Optical Image Encryption Approaches. Optical approaches are widely utilized in the field of cryptography due to its good computational speed and parallel processing. In this, a double random-phase encoding (DRPE) approach is utilized to convert the plain image into stationary white noise [131]. It uses two random phase masks that place in the input and Fourier plane. These random phase masks act as a key in DRPE. This method has been deeply researched and various optical encryption approaches have also been proposed [131].

Wu et al. [132] proposed a scalable asymmetric compressing and encrypting strategy for images by utilizing the nonlinear operation of phase truncation after cylindrical diffraction and discrete wavelet transform (DWT). Wu et al. [133] designed an asymmetric multiple-image encryption approach by utilizing compressed sensing and phase truncation. In this, a single ciphertext is attained by topsy-turvy operation of phase truncation after cylindrical diffraction. Yu et al. [134] implemented an image encryption approach depending on the hyperchaotic framework and the phase-truncated short-time fractional Fourier transform.

Wang et al. [135] implemented an image encryption based on phase-truncated Fresnel transform and random amplitude mask (RAM). Huang et al. [136] proposed a nonlinear optical multi-image encryption utilizing a chaotic map and 2D straight canonical transform. Wang et al. [137] increased the computational time and encryption capacity by reducing the number of iterations in image encryption with an improved amplitude-phase retrieval approach.

Mehra and Nishchal [138] implemented a gyrator wavelet transform by optical processing to encrypt an image depending on amplitude- and phase-truncation. The proposed approach consists of four basic factors: type and level of the mother wavelet, gyrator transform order, and position of different frequency bands. These factors are utilized as a secret key for image encryption. Sui et al. [139] designed a color image encryption approach by utilizing Yang-Gu mixture amplitude-phase retrieval approach and gyrator transform domain. A logistic map is utilized to generate the keys for encryption and decryption processes.

Wang et al. [137] designed an asymmetric optical image encryption approach by utilizing an improved amplitude-phase retrieval approach. In this, the public encryption key is generated by utilizing two random phases. To encrypt an input image into a ciphertext, an iterative amplitude and phase retrieval process are utilized. Verma and Sinha [140] proposed a nonlinear image encryption approach based on phase-truncated Fourier transform (PTFT) and natural logarithms. Liansheng et al. [141] developed an image encryption approach based on two random phases. It is free from an amplitude-phase recovery attack. Wang et al. [142] presented an encryption approach that abolishes the risk of information loss by utilizing phase-truncation approach. Table 8 demonstrates the comparison of various optical-based image encryption approaches. It demonstrates that there exists no such approach which has utilized all performance metrics.

3.4. Transform-Based Image Encryption Approaches. In transform-based image encryption approaches, the input image changed from spatial to frequency space by utilizing one of the transform domains. In most of the approaches, a color image is decomposed into three channels (i.e., R, G, and B channels). Each color channel is then encrypted through permutation and diffusion processes. The color channels can be independently processed or may be dependent on each other. After encrypting the channels, the final encrypted image is obtained by applying the inverse of transform. Some of the approaches based on the transform domain are discussed as follows.

Ran et al. [143] designed a solution for the information-independency problem in an image encryption by utilizing nonseparable fractional Fourier transform (NFrFT). Li et al. [144] implemented an encryption approach to encrypt multiple images based on cascaded FrFt. The input images are divided into two-phase masks. First phase mask is for secret key generation and second phase mask is for encrypting the images. Li and Lee [145] implemented an image encryption approach based on modified computational integral imaging reconstruction (CIIR) to solve the problem of occlusion in double-image encryption. But, the drawback of this method is transmission overhead in the network which is increasing. Chen et al. [146] designed double-image encryption by utilizing the gyrator transform (GT) and local pixel encrypting approach. It offers the clarification for crosstalk disorder found in phase-based images. In this approach, two images are combined to get complex functions.

Aburab [147] utilized Hartley transform and GT to encrypt the images. Firstly, the Hartley transform is utilized to scramble the image, and then GT is applied to obtain the final encrypted image. Yao et al. [148] implemented an encrypted approach that encrypts the color images by deduced GT. The process of encryption involves Fourier and gyrator transform. Yaru and Jianhua [149] developed an image encryption approach by utilizing FrDCT via

Table 8: Comparison of various optical-based image encryption approaches.

Ref.	HA	KA	IE	NA	NPCR	UACI	CA	PSNR
[132]	✗	✗	✓	✗	✗	✗	✓	✓
[133]	✗	✗	✗	✗	✗	✗	✓	✓
[134]	✓	✓	✓	✓	✗	✗	✓	✗
[136]	✓	✓	✓	✗	✓	✗	✓	✗
[137]	✗	✗	✓	✗	✗	✓	✓	✗
[135]	✗	✗	✓	✗	✗	✓	✓	✗

Table 9: Comparison of various transform domain-based image encryption approaches.

Ref.	KA	NPCR	UACI	IE	CA	HA	PSNR	MSE	
[143]	✗	✗	✓	✓	✗	✓	✓	✗	✗
[144]	✗	✗	✓	✓	✗	✓	✓	✗	✗
[145]	✗	✗	✓	✗	✗	✓	✓	✗	✗
[149]	✗	✗	✓	✓	✗	✓	✓	✗	✗
[150]	✗	✗	✓	✓	✗	✓	✓	✗	✗
[151]	✓	✓	✓	✓	✓	✗	✓	✗	✗
[152]	✓	✓	✓	✓	✓	✗	✗	✗	✗
[153]	✓	✓	✓	✗	✗	✗	✗	✗	✗
[154]	✗	✗	✗	✗	✗	✗	✗	✓	✗

polynomial interpolation (PI-FrDCT) and dependent scrambling and diffusion (DSD) process. Kanso and Ghebleh [150] utilized lift wavelet transform to make the encryption process visually secure. Mehra et al. [138] suggested the combination of wavelet transform and GT to guard the phase images. Lima et al. [151] designed medical image encryption by utilizing cosine number transform (CNT). It decomposes an image into blocks firstly. Afterward, CNT is applied sequentially to each block. Once image is processed, completely encrypted image is attained.

Luo et al. [152] implemented an integer wavelet transform- (IWT-) based image encryption approach. Initially, the original image is decomposed through IWT to obtain approximation and detailed coefficients. By spatiotemporal chaos, approximation coefficients are diffutilized. Afterward, by applying inverse IWT, the encrypted image is achieved.

Li et al. [153] designed an image encryption approach based on chaotic maps in the wavelet domain. Initially, plain image was decomposed by discrete wavelet transform and reconstructed the low-frequency modules. Afterward, Arnold cat map is utilized to make permutations. At last, the keystream in each diffusion process is generated by a robust chaotic map. Vaish et al. [154] evaluated an encryption approach for quantum images by utilizing quantum geometric transform, phase-shift transforms, and quantum Haar wavelet packet transform. Table 9 demonstrates the comparison of various transform domain-based image encryption approaches. It demonstrates that the development of transform-based image encryption approaches is still an open area of research.

4. Future Scope

From the comprehensive review, it has been found that the existing image encryption approaches suffer from various issues. Also, there is still room for improvement in various fields of image encryption approaches. Therefore, in the near future, one may consider the following issues to continue the research in the field of image encryption.

(i) Application-specific approaches: the current research in the field of image encryption is not done towards the building of application-specific image encryption approaches. So, in the near future, the development of application-aware image encryption approaches is a hot area of research.

(ii) Compressive sensing: development of compressive sensing-based image encryption approaches can be improved further for lightweight devices such as mobiles, spy cameras, and surveillance cameras.

(iii) Hyperparameters tuning: hyperparameters tuning of key generators such as chaotic map can be achieved utilizing the recently developed metaheuristic approaches, machine learning [155], deep learning [156], deep belief networks, or deep-transfer learning [157, 158].

(iv) Parallel processing: due to rapid advancement in the various multimedia applications such as medical and satellite imaging. These applications require high-resolution images; therefore, the development of image encryption approaches for such applications will be computationally extensive. So, the parallel image encryption approaches can be utilized to handle this issue.

(v) Multidimensional multimedia data: development of encryption approaches for multimedia data such as multispectral images is still an undeveloped area. It is required to design high-dimensional hyperchaotic systems for such kind of multidimensional multimedia data.

(vi) Steganography/data hiding: the combination of encryption with reversible data hiding/steganography has become another research direction. Therefore, one may combine both encryption and steganography kind of approaches to obtain more secure results.

5. Conclusion

This paper presented a comprehensive study of the existing image encryption approaches. It was observed that the image encryption approaches require high confusion, zero correlation with the input images, less computational complexity, and high resistance to cryptanalysis process. The comparisons among the image encryption approaches were carried out based on evaluation parameters to show their strength and weaknesses. Future research directions related to image encryption strategies were examined. It was found that the development of image encryption approaches is still an open area for researchers. This paper encourages researchers to understand the challenges involved in image encryption approaches. It will also help them to choose an appropriate approach to develop new encryption models according to an application which saves their time.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] Y. Xie, J. Yu, S. Guo, Q. Ding, and E. Wang, "Image encryption scheme with compressed sensing based on new three-dimensional chaotic system," *Entropy*, vol. 21, no. 9, p. 819, 2019.

[2] M. Kaur and V. Kumar, "A comprehensive review on image encryption techniques," *Archives of Computational Methods in Engineering*, vol. 27, pp. 1–29, 2018.

[3] A. Belazi, A. A. Abd El-Latif, and S. Belghith, "A novel image encryption scheme based on substitution-permutation network and chaos," *Signal Processing*, vol. 128, pp. 155–170, 2016.

[4] M. Kaur and V. Kumar, "Adaptive differential evolution-based lorenz chaotic system for image encryption," *Arabian Journal for Science and Engineering*, vol. 43, no. 12, pp. 8127–8144, 2018.

[5] M. Kaur and V. Kumar, "Beta chaotic map based image encryption using genetic algorithm," *International Journal of Bifurcation and Chaos*, vol. 28, no. 11, Article ID 1850132, 2018.

[6] M. Kaur, D. Singh, K. Sun, and U. Rawat, "Color image encryption using non-dominated sorting genetic algorithm with local chaotic search based 5d chaotic map," *Future Generation Computer Systems*, vol. 107, pp. 333–350, 2020.

[7] X. Wang and D. Luan, "A novel image encryption algorithm using chaos and reversible cellular automata," *Communications in Nonlinear Science and Numerical Simulation*, vol. 18, no. 11, pp. 3075–3085, 2013.

[8] C. Zhu, G. Wang, and K. Sun, "Improved cryptanalysis and enhancements of an image encryption scheme using combined 1d chaotic maps," *Entropy*, vol. 20, no. 11, p. 843, 2018.

[9] S. Zhu and C. Zhu, "Plaintext-related image encryption algorithm based on block structure and five-dimensional chaotic map," *IEEE Access*, vol. 7, pp. 147106–147118, 2019.

[10] T. Sivakumar and R. Venkatesan, "A novel image encryption using calligraphy based scan method and random number," *KSII Transactions on Internet & Information Systems*, vol. 9, no. 6, 2015.

[11] H. Liu, B. Zhao, and L. Huang, "A novel quantum image encryption algorithm based on crossover operation and mutation operation," *Multimedia Tools and Applications*, vol. 78, no. 14, pp. 20465–20483, 2019.

[12] Y. Zhang, B. Xu, and N. Zhou, "A novel image compression-encryption hybrid algorithm based on the analysis sparse representation," *Optics Communications*, vol. 392, pp. 223–233, 2017.

[13] M. Kaur, D. Singh, and R. S. Uppal, "Parallel strength pareto evolutionary algorithm-ii based image encryption," *IET Image Processing*, vol. 14, no. 6, pp. 1015–1026, 2019.

[14] D. Singh and V. Kumar, "A comprehensive review of computational dehazing techniques," *Archives of Computational Methods in Engineering*, vol. 26, no. 5, pp. 1395–1413, 2019.

[15] A. Rehman, X. Liao, A. Kulsoom, and S. A. Abbas, "Selective encryption for gray images based on chaos and dna complementary rules," *Multimedia Tools and Applications*, vol. 74, no. 13, pp. 4655–4677, 2015.

[16] E. Chen, L. Min, and G. Chen, "Discrete chaotic systems with one-line equilibria and their application to image encryption," *International Journal of Bifurcation and Chaos*, vol. 27, no. 3, Article ID 1750046, 2017.

[17] M. Kaur and V. Kumar, "Efficient image encryption method based on improved lorenz chaotic system," *Electronics Letters*, vol. 54, no. 9, pp. 562–564, 2018.

[18] J. Liu, S. Tang, J. Lian, Y. Ma, and X. Zhang, "A novel fourth order chaotic system and its algorithm for medical image encryption," *Multidimensional Systems and Signal Processing*, vol. 30, no. 4, pp. 1637–1657, 2019.

[19] C. Chen, K. Sun, and S. He, "An improved image encryption algorithm with finite computing precision," *Signal Processing*, vol. 168, Article ID 107340, 2020.

[20] K. Xuejing and G. Zihui, "A new color image encryption scheme based on dna encoding and spatiotemporal chaotic system," *Signal Processing: Image Communication*, vol. 80, Article ID 115670, 2020.

[21] X. Wang, Y. Wang, X. Zhu, and C. Luo, "A novel chaotic algorithm for image encryption utilizing one-time pad based on pixel level and DNA level," *Optics and Lasers in Engineering*, vol. 125, Article ID 105851, 2020.

[22] S. M. Ismail, L. A. Said, A. G. Radwan, A. H. Madian, and M. F. Abu-ElYazeed, "A novel image encryption system merging fractional-order edge detection and generalized chaotic maps," *Signal Processing*, vol. 167, Article ID 107280, 2020.

[23] X. Wu, D. Wang, J. Kurths, and H. Kan, "A novel lossless color image encryption scheme using 2d dwt and 6d hyperchaotic system," *Information Sciences*, vol. 349-350, pp. 137–153, 2016.

[24] X.-L. Chai, Z.-H. Gan, Y. Lu, M.-H. Zhang, and Y.-R. Chen, "A novel color image encryption algorithm based on genetic recombination and the four-dimensional memristive hyperchaotic system," *Chinese Physics B*, vol. 25, no. 10, Article ID 100503, 2016.

[25] Y. Luo, S. Tang, J. Liu, L. Cao, and S. Qiu, "Image encryption scheme by combining the hyper-chaotic system with quantum coding," *Optics and Lasers in Engineering*, vol. 124, Article ID 105836, 2020.

[26] K. A. Kumar Patro and B. Acharya, "An efficient colour image encryption scheme based on 1-d chaotic maps," *Journal of Information Security and Applications*, vol. 46, pp. 23–41, 2019.

[27] W. Feng, Y.-G. He, H.-M. Li, and C.-L. Li, "Image encryption algorithm based on discrete logarithm and memristive chaotic system," *The European Physical Journal Special Topics*, vol. 228, no. 10, pp. 1951–1967, 2019.

[28] X. Wang and S. Gao, "Application of matrix semi-tensor product in chaotic image encryption," *Journal of the Franklin Institute*, vol. 356, no. 18, pp. 11638–11667, 2019.

[29] Z. Hua and Y. Zhou, "Design of image cipher using block-based scrambling and image filtering," *Information Sciences*, vol. 396, pp. 97–113, 2017.

[30] Z.-H. Gan, X.-L. Chai, D.-J. Han, and Y.-R. Chen, "A chaotic image encryption algorithm based on 3-d bit-plane permutation," *Neural Computing and Applications*, vol. 31, no. 11, pp. 7111–7130, 2019.

[31] Q. Lu, C. Zhu, and X. Deng, "An efficient image encryption scheme based on the LSS chaotic map and single s-box," *IEEE Access*, vol. 8, pp. 25664–25678, 2020.

[32] Z. Deng and S. Zhong, "A digital image encryption algorithm based on chaotic mapping," *Journal of Algorithms & Computational Technology*, vol. 13, 2019.

[33] K. A. K. Patro, B. Acharya, and V. Nath, "Various dimensional colour image encryption based on non-overlapping block-level diffusion operation," *Microsystem Technologies*, vol. 26, pp. 1–12, 2019.

[34] X. Wang, H. Zhao, L. Feng, X. Ye, and H. Zhang, "High-sensitivity image encryption algorithm with random diffusion based on dynamic-coupled map lattices," *Optics and Lasers in Engineering*, vol. 122, pp. 225–238, 2019.

[35] X. Ye, X. Wang, S. Gao, J. Mou, Z. Wang, and F. Yang, "A new chaotic circuit with multiple memristors and its application in image encryption," *Nonlinear Dynamics*, vol. 99, no. 2, pp. 1489–1506, 2020.

[36] W. Liu, K. Sun, and C. Zhu, "A fast image encryption algorithm based on chaotic map," *Optics and Lasers in Engineering*, vol. 84, pp. 26–36, 2016.

[37] W. Cao, Y. Zhou, C. L. P. Chen, and L. Xia, "Medical image encryption using edge maps," *Signal Processing*, vol. 132, pp. 96–109, 2017.

[38] X. Chai, "An image encryption algorithm based on bit level brownian motion and new chaotic systems," *Multimedia Tools and Applications*, vol. 76, no. 1, pp. 1159–1175, 2017.

[39] S. Toughi, M. H. Fathi, and Y. A. Sekhavat, "An image encryption scheme based on elliptic curve pseudo random and advanced encryption system," *Signal Processing*, vol. 141, pp. 217–227, 2017.

[40] J. Wu, X. Liao, and B. Yang, "Color image encryption based on chaotic systems and elliptic curve elgamal scheme," *Signal Processing*, vol. 141, pp. 109–124, 2017.

[41] U. Hayat and N. A. Azam, "A novel image encryption scheme based on an elliptic curve," *Signal Processing*, vol. 155, pp. 391–402, 2019.

[42] Y. Luo, X. Ouyang, J. Liu, and L. Cao, "An image encryption method based on elliptic curve elgamal encryption and chaotic systems," *IEEE Access*, vol. 7, pp. 38507–38522, 2019.

[43] A. Banik, Z. Shamsi, and D. S. Laiphrakpam, "An encryption scheme for securing multiple medical images," *Journal of Information Security and Applications*, vol. 49, p. 102398, 2019.

[44] O. Reyad and Z. Kotulski, "Image encryption using Koblitz's encoding and new mapping method based on elliptic curve random number generator," in *Proceedings of the International Conference on Multimedia Communications, Services and Security Communications in Computer and Information Science*, pp. 34–45, Springer, Kraków, Poland, November 2015.

[45] M. Kumar, A. Iqbal, and P. Kumar, "A new RGB image encryption algorithm based on DNA encoding and elliptic curve Diffie-Hellman cryptography," *Signal Processing*, vol. 125, pp. 187–202, 2016.

[46] X. Zhang and X. Wang, "Digital image encryption algorithm based on elliptic curve public cryptosystem," *IEEE Access*, vol. 6, pp. 70025–70034, 2018.

[47] D. S. Laiphrakpam and M. S. Khumanthem, "A robust image encryption scheme based on chaotic system and elliptic curve over finite field," *Multimedia Tools and Applications*, vol. 77, no. 7, pp. 8629–8652, 2018.

[48] Z. E. Dawahdeh, S. N. Yaakob, and R. Razif bin Othman, "A new image encryption technique combining elliptic curve cryptosystem with hill cipher," *Journal of King Saud University - Computer and Information Sciences*, vol. 30, no. 3, pp. 349–355, 2018.

[49] Z. Liu, T. Xia, and J. Wang, "Image encryption technique based on new two-dimensional fractional-order discrete chaotic map and Menezes-Vanstone elliptic curve cryptosystem," *Chinese Physics B*, vol. 27, no. 3, Article ID 030502, 2018.

[50] S. Nagaraj, G. S. V. P. Raju, and K. K. Rao, "Image encryption using elliptic curve cryptograhy and matrix," *Procedia Computer Science*, vol. 48, pp. 276–281, 2015.

[51] B. Mondal, S. Singh, and P. Kumar, "A secure image encryption scheme based on cellular automata and chaotic skew tent map," *Journal of Information Security and Applications*, vol. 45, pp. 117–130, 2019.

[52] S. Khan, L. Han, H. Lu, K. K. Butt, G. Bachira, and N.-U. Khan, "A new hybrid image encryption algorithm based on 2D-CA, FSM-DNA rule generator, and FSBI," *IEEE Access*, vol. 7, pp. 81333–81350, 2019.

[53] W. Zhang, Z. Zhu, and H. Yu, "A symmetric image encryption algorithm based on a coupled logistic-Bernoulli map and cellular automata diffusion strategy," *Entropy*, vol. 21, no. 5, p. 504, 2019.

[54] Y. Su, Y. Wo, and G. Han, "Reversible cellular automata image encryption for similarity search," *Signal Processing: Image Communication*, vol. 72, pp. 134–147, 2019.

[55] M. T. Ramírez, M. Mejía Carlos, J. S. Murguía Ibarra, and L. J. Ontañón García Pimentel, "Partial image encryption using cellular automata," *Computación y Sistemas*, vol. 23, no. 4, 2019.

[56] Y. Wang, Y. Zhao, Q. Zhou, and Z. Lin, "Image encryption using partitioned cellular automata," *Neurocomputing*, vol. 275, pp. 1318–1332, 2018.

[57] A. Yaghouti Niyat, M. H. Moattar, and M. Niazi Torshiz, "Color image encryption based on hybrid hyper-chaotic system and cellular automata," *Optics and Lasers in Engineering*, vol. 90, pp. 225–237, 2017.

[58] X. Chai, Z. Gan, K. Yang, Y. Chen, and X. Liu, "An image encryption algorithm based on the memristive hyperchaotic system, cellular automata and DNA sequence operations," *Signal Processing: Image Communication*, vol. 52, pp. 6–19, 2017.

[59] R. Wei, X. Li, and Q.-H. Wang, "Double color image encryption scheme based on off-axis holography and maximum length cellular automata," *Optik*, vol. 145, pp. 407–417, 2017.

[60] T. Chen, M. Zhang, J. Wu, C. Yuen, and Y. Tong, "Image encryption and compression based on kronecker compressed sensing and elementary cellular automata scrambling," *Optics & Laser Technology*, vol. 84, pp. 118–133, 2016.

[61] A. Souyah and K. M. Faraoun, "An image encryption scheme combining chaos-memory cellular automata and weighted histogram," *Nonlinear Dynamics*, vol. 86, no. 1, pp. 639–653, 2016.

[62] D. Tralic and S. Grgic, "Robust image encryption based on balanced cellular automaton and pixel separation," *Radioengineering*, vol. 25, no. 3, pp. 548–555, 2016.

[63] Y.-G. Yang, J. Tian, H. Lei, Y.-H. Zhou, and W.-M. Shi, "Novel quantum image encryption using one-dimensional quantum cellular automata," *Information Sciences*, vol. 345, pp. 257–270, 2016.

[64] B. Murugan, A. G. Nanjappa Gounder, and S. Manohar, "A hybrid image encryption algorithm using chaos and Conway's game-of-life cellular automata," *Security and Communication Networks*, vol. 9, no. 7, pp. 634–651, 2016.

[65] A. Souyah and K. M. Faraoun, "Fast and efficient randomized encryption scheme for digital images based on quadtree

decomposition and reversible memory cellular automata," *Nonlinear Dynamics*, vol. 84, no. 2, pp. 715–732, 2016.

[66] R. Enayatifar, H. J. Sadaei, A. H. Abdullah, M. Lee, and I. F. Isnin, "A novel chaotic based image encryption using a hybrid model of deoxyribonucleic acid and cellular automata," *Optics and Lasers in Engineering*, vol. 71, pp. 33–41, 2015.

[67] S. Hanis and R. Amutha, "Double image compression and encryption scheme using logistic mapped convolution and cellular automata," *Multimedia Tools and Applications*, vol. 77, no. 6, pp. 6897–6912, 2018.

[68] X. Zhang, H. Zhang, and C. Xu, "Reverse iterative image encryption scheme using 8-layer cellular automata," *KSII Transactions on Internet & Information Systems*, vol. 10, no. 7, 2016.

[69] S. Zhou, B. Wang, X. Zheng, and C. Zhou, "An image encryption scheme based on DNA computing and cellular automata," *Discrete Dynamics in Nature and Society*, vol. 2016, Article ID 5408529, 9 pages, 2016.

[70] A. M. Del Rey and G. R. Sánchez, "An image encryption algorithm based on 3d cellular automata and chaotic maps," *International Journal of Modern Physics C*, vol. 26, no. 1, Article ID 1450069, 2015.

[71] M. N. Aslam, A. Belazi, S. Kharbech, M. Talha, and W. Xiang, "Fourth order mca and chaos-based image encryption scheme," *IEEE Access*, vol. 7, pp. 66395–66409, 2019.

[72] X. Li, Y. Wang, Q.-H. Wang, Y. Liu, and X. Zhou, "Modified integral imaging reconstruction and encryption using an improved sr reconstruction algorithm," *Optics and Lasers in Engineering*, vol. 112, pp. 162–169, 2019.

[73] X. Chai, Y. Chen, and L. Broyde, "A novel chaos-based image encryption algorithm using DNA sequence operations," *Optics and Lasers in Engineering*, vol. 88, pp. 197–213, 2017.

[74] R. Guesmi, M. A. B. Farah, A. Kachouri, and M. Samet, "A novel chaos-based image encryption using DNA sequence operation and secure hash algorithm sha-2," *Nonlinear Dynamics*, vol. 83, no. 3, pp. 1123–1136, 2016.

[75] X. Wu, H. Kan, and J. Kurths, "A new color image encryption scheme based on DNA sequences and multiple improved 1D chaotic maps," *Applied Soft Computing*, vol. 37, pp. 24–39, 2015.

[76] B. Mondal and T. Mandal, "A light weight secure image encryption scheme based on chaos & DNA computing," *Journal of King Saud University - Computer and Information Sciences*, vol. 29, no. 4, pp. 499–504, 2017.

[77] X. Wu, K. Wang, X. Wang, and H. Kan, "Lossless chaotic color image cryptosystem based on DNA encryption and entropy," *Nonlinear Dynamics*, vol. 90, no. 2, pp. 855–875, 2017.

[78] X. Li, L. Wang, Y. Yan, and P. Liu, "An improvement color image encryption algorithm based on DNA operations and real and complex chaotic systems," *Optik*, vol. 127, no. 5, pp. 2558–2565, 2016.

[79] X.-Y. Wang, H.-L. Zhang, and X.-M. Bao, "Color image encryption scheme using cml and DNA sequence operations," *Biosystems*, vol. 144, pp. 18–26, 2016.

[80] H. Nematzadeh, R. Enayatifar, M. Yadollahi, M. Lee, and G. Jeong, "Binary search tree image encryption with DNA," *Optik*, vol. 202, p. 163505, 2020.

[81] A. Jain and N. Rajpal, "A robust image encryption algorithm resistant to attacks using DNA and chaotic logistic maps," *Multimedia Tools and Applications*, vol. 75, no. 10, pp. 5455–5472, 2016.

[82] X. Wang and C. Liu, "A novel and effective image encryption algorithm based on chaos and DNA encoding," *Multimedia Tools and Applications*, vol. 76, no. 5, pp. 6229–6245, 2017.

[83] X. Zhang, Z. Zhou, and Y. Niu, "An image encryption method based on the Feistel network and dynamic DNA encoding," *IEEE Photonics Journal*, vol. 10, no. 4, pp. 1–14, 2018.

[84] X. Chai, Z. Gan, Y. Lu, Y. Chen, and D. Han, "A novel image encryption algorithm based on the chaotic system and DNA computing," *International Journal of Modern Physics C*, vol. 28, no. 5, Article ID 1750069, 2017.

[85] J. Zhang, D. Hou, and H. Ren, "Image encryption algorithm based on dynamic DNA coding and chen's hyperchaotic system," *Mathematical Problems in Engineering*, vol. 2016, 2016.

[86] Y. Liu, J. Wang, J. Fan, and L. Gong, "Image encryption algorithm based on chaotic system and dynamic s-boxes composed of DNA sequences," *Multimedia Tools and Applications*, vol. 75, no. 8, pp. 4363–4382, 2016.

[87] B. Norouzi and S. Mirzakuchaki, "An image encryption algorithm based on DNA sequence operations and cellular neural network," *Multimedia Tools and Applications*, vol. 76, no. 11, pp. 13681–13701, 2017.

[88] W. Liu, K. Sun, Y. He, and M. Yu, "Color image encryption using three-dimensional sine ICMIC modulation map and DNA sequence operations," *International Journal of Bifurcation and Chaos*, vol. 27, no. 11, Article ID 1750171, 2017.

[89] X. Zhang, F. Han, and Y. Niu, "Chaotic image encryption algorithm based on bit permutation and dynamic DNA encoding," *Computational Intelligence and Neuroscience*, vol. 2017, Article ID 6919675, 11 pages, 2017.

[90] H. R. Shakir, "A color-image encryption scheme using a 2D chaotic system and Dna coding," *Advances in Multimedia*, vol. 2019, 2019.

[91] S. Chirakkarottu and S. Mathew, "A novel encryption method for medical images using 2D Zaslavski map and DNA cryptography," *SN Applied Sciences*, vol. 2, no. 1, p. 1, 2020.

[92] M. Xu, "Cryptanalysis of an image encryption algorithm based on dna sequence operation and hyper-chaotic system," *3D Research*, vol. 8, no. 2, p. 15, 2017.

[93] P. Zhen, G. Zhao, L. Min, and X. Jin, "Chaos-based image encryption scheme combining dna coding and entropy," *Multimedia Tools and Applications*, vol. 75, no. 11, pp. 6303–6319, 2016.

[94] X. Wu, K. Wang, X. Wang, H. Kan, and J. Kurths, "Color image dna encryption using nca map-based cml and one-time keys," *Signal Processing*, vol. 148, pp. 272–287, 2018.

[95] X. Wu, J. Kurths, and H. Kan, "A robust and lossless dna encryption scheme for color images," *Multimedia Tools and Applications*, vol. 77, no. 10, pp. 12349–12376, 2018.

[96] S. Mozaffari, "Parallel image encryption with bitplane decomposition and genetic algorithm," *Multimedia Tools and Applications*, vol. 77, no. 19, pp. 25799–25819, 2018.

[97] K. Shankar and P. Eswaran, "An efficient image encryption technique based on optimized key generation in ECC using genetic algorithm," in *Artificial Intelligence and Evolutionary Computations in Engineering Systems, Advances in Intelligent Systems and Computing*, pp. 705–714, Springer, Berlin, Germany, 2016.

[98] M. Kaur and V. Kumar, "Colour image encryption technique using differential evolution in non-subsampled contourlet transform domain," *IET Image Processing*, vol. 12, no. 7, pp. 1273–1283, 2018.

[99] M. Kaur and V. Kumar, "Fourier-Mellin moment-based intertwining map for image encryption," *Modern Physics Letters B*, vol. 32, no. 9, Article ID 1850115, 2018.

[100] H. Nematzadeh, R. Enayatifar, H. Motameni, F. G. Guimarães, and V. N. Coelho, "Medical image encryption using a hybrid model of modified genetic algorithm and coupled map lattices," *Optics and Lasers in Engineering*, vol. 110, pp. 24–32, 2018.

[101] N. K. Pareek and V. Patidar, "Medical image protection using genetic algorithm operations," *Soft Computing*, vol. 20, no. 2, pp. 763–772, 2016.

[102] K. Mirzaei Talarposhti and M. Khaki Jamei, "A secure image encryption method based on dynamic harmony search (dhs) combined with chaotic map," *Optics and Lasers in Engineering*, vol. 81, pp. 21–34, 2016.

[103] M. Kaur, V. Kumar, and L. Li, "Color image encryption approach based on memetic differential evolution," *Neural Computing and Applications*, vol. 31, no. 11, pp. 7975–7987, 2019.

[104] A. Gupta, D. Singh, and M. Kaur, "An efficient image encryption using non-dominated sorting genetic algorithm-iii based 4-d chaotic maps," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 3, pp. 1309–1324, 2020.

[105] M. Mahmud, M. Atta-ur-Rahman, M. Lee, and J.-Y. Choi, "Evolutionary-based image encryption using rna codons truth table," *Optics & Laser Technology*, vol. 121, p. 105818, 2020.

[106] M. Kaur and V. Kumar, "Parallel non-dominated sorting genetic algorithm-ii-based image encryption technique," *The Imaging Science Journal*, vol. 66, no. 8, pp. 453–462, 2018.

[107] X.-L. Chai, Z.-H. Gan, Y. Lu, M.-H. Zhang, and Y.-R. Chen, "A novel color image encryption algorithm based on genetic recombination and the four-dimensional memristive hyperchaotic system," *Chinese Physics B*, vol. 25, no. 10, p. 100503, 2016.

[108] J. Wang, "Digital image encryption algorithm design based on genetic hyperchaos," *International Journal of Optics*, vol. 2016, 2016.

[109] X. Wang and H.-l. Zhang, "A novel image encryption algorithm based on genetic recombination and hyper-chaotic systems," *Nonlinear Dynamics*, vol. 83, no. 1-2, pp. 333–346, 2016.

[110] X. Zhang, X. Wang, and Y. Cheng, "Image encryption based on a genetic algorithm and a chaotic system," *IEICE Transactions on Communications*, vol. E98.B, no. 5, pp. 824–833, 2015.

[111] R. Premkumar and S. Anand, "Secured and compound 3-d chaos image encryption using hybrid mutation and crossover operator," *Multimedia Tools and Applications*, vol. 78, no. 8, pp. 9577–9593, 2019.

[112] L. Wang, L. Li, J. Li, J. Li, B. B. Gupta, and X. Liu, "Compressive sensing of medical images with confidentially homomorphic aggregations," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1402–1409, 2018.

[113] L. Gong, K. Qiu, C. Deng, and N. Zhou, "An image compression and encryption algorithm based on chaotic system and compressive sensing," *Optics & Laser Technology*, vol. 115, pp. 257–267, 2019.

[114] R. Ponuma and R. Amutha, "Encryption of image data using compressive sensing and chaotic system," *Multimedia Tools and Applications*, vol. 78, no. 9, pp. 11857–11881, 2019.

[115] R. Ponuma, R. Amutha, S. Aparna, and G. Gopal, "Visually meaningful image encryption using data hiding and chaotic compressive sensing," *Multimedia Tools and Applications*, vol. 78, no. 18, pp. 25707–25729, 2019.

[116] W. Shao, M. Cheng, C. Luo et al., "An image encryption scheme based on hybrid electro-optic chaotic sources and compressive sensing," *IEEE Access*, vol. 7, pp. 156582–156591, 2019.

[117] L. Zhu, H. Song, X. Zhang, M. Yan, L. Zhang, and T. Yan, "A novel image encryption scheme based on nonuniform sampling in block compressive sensing," *IEEE Access*, vol. 7, pp. 22161–22174, 2019.

[118] Q. Shen, W. Liu, Y. Lin, and Y. Zhu, "Designing an image encryption scheme based on compressive sensing and non-uniform quantization for wireless visual sensor networks," *Sensors*, vol. 19, no. 14, p. 3081, 2019.

[119] H. Jiang, Z. Nie, N. Zhou, and W. Zhang, "Compressive-sensing-based double-image encryption algorithm combining double random phase encoding with josephus traversing operation," *Optica Applicata*, vol. 49, no. 3, 2019.

[120] S. Yao, L. Chen, and Y. Zhong, "An encryption system for color image based on compressive sensing," *Optics & Laser Technology*, vol. 120, p. 105703, 2019.

[121] Q. Xu, K. Sun, C. Cao, and C. Zhu, "A fast image encryption algorithm based on compressive sensing and hyperchaotic map," *Optics and Lasers in Engineering*, vol. 121, pp. 203–214, 2019.

[122] L. Gong, K. Qiu, C. Deng, and N. Zhou, "An optical image compression and encryption scheme based on compressive sensing and rsa algorithm," *Optics and Lasers in Engineering*, vol. 121, pp. 169–180, 2019.

[123] S. Zhu and C. Zhu, "A new image compression-encryption scheme based on compressive sensing and cyclic shift," *Multimedia Tools and Applications*, vol. 78, no. 15, pp. 20855–20875, 2019.

[124] H. Wang, D. Xiao, M. Li, Y. Xiang, and X. Li, "A visually secure image encryption scheme based on parallel compressive sensing," *Signal Processing*, vol. 155, pp. 218–232, 2019.

[125] Y. Luo, J. Lin, J. Liu et al., "A robust image encryption algorithm based on Chua's circuit and compressive sensing," *Signal Processing*, vol. 161, pp. 227–247, 2019.

[126] R. Ponuma and R. Amutha, "Image encryption using sparse coding and compressive sensing," *Multidimensional Systems and Signal Processing*, vol. 30, no. 4, pp. 1895–1909, 2019.

[127] Y. Song, Z. Zhu, W. Zhang, L. Guo, X. Yang, and H. Yu, "Joint image compression-encryption scheme using entropy coding and compressive sensing," *Nonlinear Dynamics*, vol. 95, no. 3, pp. 2235–2261, 2019.

[128] F. Han, X. Liao, B. Yang, and Y. Zhang, "A hybrid scheme for self-adaptive double color-image encryption," *Multimedia Tools and Applications*, vol. 77, no. 11, pp. 14285–14304, 2018.

[129] T. Zhang, S. Li, R. Ge, M. Yuan, and Y. Ma, "A novel 1d hybrid chaotic map-based image compression and encryption using compressed sensing and fibonacci-lucas transform," *Mathematical Problems in Engineering*, vol. 2016, 2016.

[130] C. Pan, G. Ye, X. Huang, and J. Zhou, "Novel meaningful image encryption based on block compressive sensing," *Security and Communication Networks*, vol. 2019, 2019.

[131] Y. Qin and Q. Gong, "Multiple-image encryption in an interference-based scheme by lateral shift multiplexing," *Optics Communications*, vol. 315, pp. 220–225, 2014.

[132] C. Wu, K.-Y. Hu, Y. Wang, J. Wang, and Q.-H. Wang, "Scalable asymmetric image encryption based on phase-

truncation in cylindrical diffraction domain," *Optics Communications*, vol. 448, pp. 26–32, 2019.

[133] C. Wu, Y. Wang, Y. Chen, J. Wang, and Q.-H. Wang, "Asymmetric encryption of multiple-image based on compressed sensing and phase-truncation in cylindrical diffraction domain," *Optics Communications*, vol. 431, pp. 203–209, 2019.

[134] S.-S. Yu, N.-R. Zhou, L.-H. Gong, and Z. Nie, "Optical image encryption algorithm based on phase-truncated short-time fractional fourier transform and hyper-chaotic system," *Optics and Lasers in Engineering*, vol. 124, p. 105816, 2020.

[135] W. Chen, "Optical multiple-image encryption using three-dimensional space," *IEEE Photonics Journal*, vol. 8, no. 2, pp. 1–8, 2016.

[136] Z.-J. Huang, S. Cheng, L.-H. Gong, and N.-R. Zhou, "Nonlinear optical multi-image encryption scheme with two-dimensional linear canonical transform," *Optics and Lasers in Engineering*, vol. 124, p. 105821, 2020.

[137] Y. Wang, C. Quan, and C. J. Tay, "Asymmetric optical image encryption based on an improved amplitude-phase retrieval algorithm," *Optics and Lasers in Engineering*, vol. 78, pp. 8–16, 2016.

[138] I. Mehra and N. K. Nishchal, "Optical asymmetric image encryption using gyrator wavelet transform," *Optics Communications*, vol. 354, pp. 344–352, 2015.

[139] L. Sui, B. Liu, Q. Wang, Y. Li, and J. Liang, "Color image encryption by using yang-gu mixture amplitude-phase retrieval algorithm in gyrator transform domain and two-dimensional sine logistic modulation map," *Optics and Lasers in Engineering*, vol. 75, pp. 17–26, 2015.

[140] G. Verma and A. Sinha, "Optical image encryption system using nonlinear approach based on biometric authentication," *Journal of Modern Optics*, vol. 64, no. 13, pp. 1321–1329, 2017.

[141] S. Liansheng, Z. bei, W. Zhanmin, and S. qindong, "Amplitude-phase retrieval attack free image encryption based on two random masks and interference," *Optics and Lasers in Engineering*, vol. 86, pp. 1–10, 2016.

[142] Y. Wang, C. Quan, and C. J. Tay, "Optical color image encryption without information disclosure using phase-truncated fresnel transform and a random amplitude mask," *Optics Communications*, vol. 344, pp. 147–155, 2015.

[143] Q. Ran, L. Yuan, and T. Zhao, "Image encryption based on nonseparable fractional Fourier transform and chaotic map," *Optics Communications*, vol. 348, pp. 43–49, 2015.

[144] Y. Li, F. Zhang, Y. Li, and R. Tao, "Asymmetric multiple-image encryption based on the cascaded fractional Fourier transform," *Optics and Lasers in Engineering*, vol. 72, pp. 18–25, 2015.

[145] X.-W. Li and I.-K. Lee, "Modified computational integral imaging-based double image encryption using fractional Fourier transform," *Optics and Lasers in Engineering*, vol. 66, pp. 112–121, 2015.

[146] J.-X. Chen, Z.-L. Zhu, C. Fu, L.-B. Zhang, and H. Yu, "Analysis and improvement of a double-image encryption scheme using pixel scrambling technique in gyrator domains," *Optics and Lasers in Engineering*, vol. 66, pp. 1–9, 2015.

[147] M. R. Abuturab, "An asymmetric single-channel color image encryption based on Hartley transform and gyrator transform," *Optics and Lasers in Engineering*, vol. 69, pp. 49–57, 2015.

[148] L. Yao, C. Yuan, J. Qiang, S. Feng, and S. Nie, "An asymmetric color image encryption method by using deduced gyrator transform," *Optics and Lasers in Engineering*, vol. 89, pp. 72–79, 2017.

[149] L. Yaru and W. Jianhua, "New image encryption combining fractional DCT via polynomial interpolation with dependent scrambling and diffusion," *The Journal of China Universities of Posts and Telecommunications*, vol. 22, no. 5, pp. 1–9, 2015.

[150] A. Kanso and M. Ghebleh, "An algorithm for encryption of secret images into meaningful images," *Optics and Lasers in Engineering*, vol. 90, pp. 196–208, 2017.

[151] J. B. Lima, F. Madeiro, and F. J. R. Sales, "Encryption of medical images based on the cosine number transform," *Signal Processing: Image Communication*, vol. 35, pp. 1–8, 2015.

[152] Y. Luo, M. Du, and J. Liu, "A symmetrical image encryption scheme in wavelet and time domain," *Communications in Nonlinear Science and Numerical Simulation*, vol. 20, no. 2, pp. 447–460, 2015.

[153] C.-L. Li, H.-M. Li, F.-D. Li, D.-Q. Wei, X.-B. Yang, and J. Zhang, "Multiple-image encryption by using robust chaotic map in wavelet transform domain," *Optik*, vol. 171, pp. 277–286, 2018.

[154] A. Vaish, S. Gautam, and M. Kumar, "A wavelet based approach for simultaneous compression and encryption of fused images," *Journal of King Saud University-Computer and Information Sciences*, vol. 31, no. 2, pp. 208–217, 2019.

[155] G. Qi, H. Wang, M. Haner, C. Weng, S. Chen, and Z. Zhu, "Convolutional neural network based detection and judgement of environmental obstacle in vehicle operation," *CAAI Transactions on Intelligence Technology*, vol. 4, no. 2, pp. 80–91, 2019.

[156] Y. Pathak, P. K. Shukla, A. Tiwari, S. Stalin, S. Singh, and P. K. Shukla, "Deep transfer learning based classification model for COVID-19 disease," *IRBM*, 2020.

[157] H. S. Basavegowda and G. Dagnew, "Deep learning approach for microarray cancer data classification," *CAAI Transactions on Intelligence Technology*, vol. 5, no. 1, pp. 22–33, 2020.

[158] Y. Tingting, W. Junqian, W. Lintai, and X. Yong, "Three-stage network for age estimation," *CAAI Transactions on Intelligence Technology*, vol. 4, no. 2, pp. 122–126, 2019.

Hindawi
Mathematical Problems in Engineering
Volume 2021, Article ID 2571863, 12 pages
https://doi.org/10.1155/2021/2571863

Research Article

Dingo Optimizer: A Nature-Inspired Metaheuristic Approach for Engineering Problems

Amit Kumar Bairwa,[1] **Sandeep Joshi**,[1] **and Dilbag Singh**[2]

[1]*Manipal University Jaipur, Rajasthan, Jaipur, India*
[2]*Bennett University, Uttar Pradesh, Noida, India*

Correspondence should be addressed to Sandeep Joshi; sjoshinew@yahoo.com

Received 23 July 2020; Revised 8 October 2020; Accepted 16 December 2020; Published 10 June 2021

Academic Editor: Yann Favennec

Optimization is a buzzword, whenever researchers think of engineering problems. This paper presents a new metaheuristic named dingo optimizer (DOX) which is motivated by the behavior of dingo (*Canis familiaris dingo*). The overall concept is to develop this method involving the collaborative and social behavior of dingoes. The developed algorithm is based on the hunting behavior of dingoes that includes exploration, encircling, and exploitation. All the above prey hunting steps are modeled mathematically and are implemented in the simulator to test the performance of the proposed algorithm. Comparative analyses are drawn among the proposed approach and grey wolf optimizer (GWO) and particle swarm optimizer (PSO). Some of the well-known test functions are used for the comparative study of this work. The results reveal that the dingo optimizer performed significantly better than other nature-inspired algorithms.

1. Introduction

The challenges of the modern world are composed of various goals that must be optimized at the same time. Optimization is a process that seeks one or more solutions to the problem that leads to the extreme values of one or more objective [1]. The optimization can, therefore, be done based on single or multiple objective functions [2–4].

Keeping this in the mind, there is a requirement of new metaheuristic-based solution to reduce the burden of any of the model designing. The objective of this paper is to develop a nature-based algorithm called dingo optimizer, which can be abbreviated as DOX. It is based on dingo's social hierarchy and prey hunting behavior.

Metaheuristic algorithms are remarkably common due to its nature of flexibility, simplicity, less mathematical complexity, and avoidance of local optima. If we talk about flexibility, then it means we can use such algorithms in a wide variety of engineering problems. Such algorithms provide satisfactory results for many of the complex problems [5]. It is simple because it is inspired by nature like animal behavior to accomplish a particular task, physical phenomena, and other evolutionary behavior.

One of the main reasons to use the metaheuristics in real-life problems is that almost all the optimization solutions start with the random processes, and for such solutions, there is no need to find out the optimum. Metaheuristic algorithms are very powerful in terms of finding local optima compared with the traditional optimization algorithms. Finding the real search space in the real world problem is very much complicated because of finding with lots of local optima in the search. That is the reason metaheuristic algorithms are most suitable to find out such challenging issues.

There are so many metaheuristic algorithms proposed every year, and they show the promising result with respect to the engineering problem. However, day by day the nature and complexity of new applications are introducing with new challenges. And, it might not be possible to solve the particular problem with the guarantee. This motivates us to develop a new metaheuristics algorithm as dingo optimizer (DOX). Also, the method which is mathematically modeled and inspired by the social hierarchy of dingoes is also motivated to solve a real-time engineering problem.

This paper is arranged as follows. Section 2 provides a literature survey that explains the brief principles of DOX focused on dog pack hunting. The proposed multiobjective DOX is presented in Section 3. In section 4, the performance of DOX is tested with various benchmark functions and experimental results are compared with other state-of-the-art algorithms. In Section 5, the conclusion and future research directions discussed.

2. Literature Review

In the past few years, the problems related to real-life have increased, and it is motivating researchers to develop a better metaheuristic technique with the concept of randomization and local search. Such approaches have been used to determine the best solutions to real-life engineering problems that are feasible. These methods are more widely accepted due to their difficulty and reliability relative to other existing methods.

Metaheuristic algorithms can be categorized into evolutionary-based [13], physical algorithms [54], bio-inspired algorithms, swarm-based [25], and others methods. The evolutionary algorithm gives approximately close solutions to all types of optimization models since these approaches are not dependent on basic fitness and assumptions [55]. Bio-inspired algorithms are also popular to solve various complicated problems, which are motivated by biological evolution such as selection, replication, mutation, and recombination. Physical algorithms are motivated by evolutionary algorithms by the principle of natural selection in which a species attempts to live in different environments based on the fitness test. The most common parameters such as inertia force, electromagnetic force, and gravitational force help the algorithm to search the agent's coordinate and search around the space. Swarm-based algorithms are inspired by the self-organized nature of the social creatures, which shows the collective behavior of decentralization. Corporate wisdom is influenced by the contact of swarms with each other and their surroundings. Some of the popular swarm intelligence technique is quite similar to the nature-inspired algorithm. Generally, swarm-based algorithms are more popular as they have fewer operators (i.e., discovery, encircling, and exploitation). Table 1 lists all these algorithms which are divided into single objective and multi-objectives depending on the number of objective functions.

3. Dingo Optimizer (DOX)

3.1. Motivation. Nature is always the most powerful teacher from the beginning. Ever species surviving on the Earth have its way of unique mechanism for survival. Social relationships are one of them, which is dynamic. Based on the general study of the social behavior of the animal, it can be segmented into some of the categories. The first category is depending on the environmental factors, i.e., nearby resource availability and challenges created by other species. Another category depends on individual behavior or quality.

Keeping this in the mind, dingo is motivation to our work which follows strictly the social relationship. Dingo is the dog's sort. The scientific name of dingo is *Canis lupus* (wolf) *dingo*, changed recently from *Canis familiaris* (dog). Dingoes are complicated, intelligent, and highly social animals. Dingoes are skillful hunters living in a pack of the average size 12–15.

Social hierarchy is highly structured, alpha is on the top of the hierarchy, and they might be male or female. They can be identified based on the responsibility like making decisions, sleeping places, and hunting. The most dominant and strongest member in the pack is called alpha, considered as the leader of the pack of dingoes. It reflects that the discipline and organization are more important than power. The decision taken by the alpha is dictated to the pack. In general, all the members of a pack acknowledge the alpha by holding their tails down.

Beta dingoes are at the second level in the hierarchy, which played a role of intermediate between alpha and another pack for the related tasks. It plays the important role as an adviser of alpha and maintains the discipline for the whole pack. The beta confirms the orders of the alpha in the group and communicates to the alpha. The beta dingo is second in the hierarchy after the alpha. If alpha does not survive due to any of the reasons, all the commands will be handed over by the beta to control the other lower-level dingoes.

If a dingo does not belong to an alpha or beta, they are considered as subordinate. These subordinates follow alphas and betas. Scouts shall be liable for observing the area of the territories and shall alert the group in case of any threat circumstance. Hunters shall support the alphas and betas to catch the prey and provide food for the group.

Based on the studies, dingoes have an accurate sense of communication. They communicate with each other through sensing different sound intensities in the air. In DOX, dingo creates sound feedback in such a way that dingoes exchange their knowledge with others to create common community details. The amplitude of the vibration is modified by the strength of the person as the dingo enters a new location from the previous one.

Group hunting is an interesting social behavior of dingoes, which makes its more extension to the social behavior of dingoes. Hunting strategy is categorized in their phases as follows:

Chasing and approaching

Encircling and harassing

Attack

The above steps are properly shown in Figure 1. Also, hunting behavior and the social arrangements of dingo are modeled mathematically, to develop DOX to perform nature-inspired optimization.

Exploration and exploitation are the two main components of DOX. In the exploration part, the algorithm reaches several expected solutions in the search space but exploitation allows searching for optimal solutions within the given space. To find out the best solution for any real-life problem, both the components are required with fine-

Table 1: Classification of different metaheuristic approaches.

Types of algorithms	Single objective	Multiobjective
Evolutionary algorithms (EA) [6]	Genetic algorithm (GA) [7]	NSGA II [2, 8]
	Genetic programming (GP) [9]	Multiobjective GP [10]
	Differential evolution (DE) [11]	Multiobjective DE [12]
	Evolutionary strategy (ES) [13]	Multiobjective ES [14]
Bio-inspired algorithms (BIA)	Artificial immune system (AIS) algorithm [15]	Multiobjective AIS [16]
	Bacterial foraging algorithm (BFA) [17]	Multiobjective BFA [18]
	Dendritic cell algorithm (DCA) [19]	
	Krill herd algorithm (KHA) [20]	
Physical algorithms	Simulated annealing (SA) [21]	Multiobjective SA
	Memetic algorithm (MA) [22]	Multiobjective MA
	Shuffled frog-leaping algorithm [23]	Multiobjective SFA
Swarm intelligence (SI)	Ant colony optimization (ACO) [24]	Multiobjective ACO
	Particle swarm optimization (PSO) [25]	Multiobjective PSO [1]
	Artificial bee colony (ABC) [26]	Multiobjective ABC
	Fish swarm algorithm (FSA) [27]	Multiobjective FSA
	Grey wolf optimizer (GWO) [28]	
	Dragonfly algorithm (DA) [29]	
Other nature-inspired algorithms	Firefly algorithm [30]	Multiobjective firefly [31]
	Whale optimization algorithm (WOA) [32]	
	Gravitational search algorithm [33]	Multiobjective GSA
	Bat algorithm (BA) [34]	Multiobjective Bat
	Cuckoo search algorithm (CSA) [35]	Multiobjective cuckoo
	Cat swarm algorithm (CSA) [36]	Multiobjective CSO
Human behavior-inspired algorithms	Harmony search (HS) [37]	Multiobjective HS [38]
	Tabu search (TS) [39]	Multiobjective TS [40]
	Parameter adaptive harmony search (PAHS) [41]	Multiobjective PAHS [41]
	Group search optimizer (GSO) [42]	Multiobjective GSO
	Exchange market algorithm (EMA) [43]	Multiobjective EMA
	Imperialist competitive algorithm (ICA) [44]	Multiobjective ICA
	Soccer league competition algorithm (SLCA) [45]	Multiobjective SLCA
	League championship algorithm (LCA) [46]	Multiobjective LCA
	Social-based algorithm (SBA) [47]	Multiobjective SBA
	Firework algorithm (FA) [48]	Multiobjective FA
	Colliding bodies optimization (CBO) [49]	Multiobjective CBO
	Soccer league competition algorithm (SLCA) [45]	Multiobjective SLCA
	Interior search algorithm (ISA) [50]	Multiobjective ISA
	Artificial ecosystem-based optimization (AEO) [51]	
	Spiral optimization algorithm (SOA) [52]	
	Adolescent identity search algorithm (AISA) [53]	

tuning. However, it is a challenge to make a balance between the components of the proposed algorithm due to its stochastic nature. To solve a real-life engineering problem, this inspiring fact motivates a hybridize metaheuristics algorithm design.

3.2. Mathematical Models. The representation of the searching, encircling, and attacking prey is designed mathematically to perform the dingo optimization in this section.

3.2.1. Encircling. Dingoes are enough capable to find the location of the prey. After tracing the location, the pack followed by alpha encircles the prey. To model dingo's social hierarchy, it is assumed that the existing best agent approach is the goal or aim prey, which is similar to the optimal since the quest area is not known a priori. In the meantime, other quest agencies are still seeking to refresh their strategies on the next possible approach. This behavior of the dingoes is modeled by the following mathematical equations (1)–(5). Also, a detailed description of the nomenclatures used in the equation is provided in Table 2

$$\vec{D}_d = \left|\vec{A} \cdot \vec{P}_p(x) - \vec{P}(i)\right|, \tag{1}$$

$$\vec{P}(i+1) = \vec{P}_p(i) - \vec{B} \cdot \vec{D}(d), \tag{2}$$

$$\vec{A} = 2 \cdot \vec{a}_1, \tag{3}$$

$$\vec{B} = 2\vec{b} \cdot \vec{a}_2 - \vec{b}, \tag{4}$$

$$\vec{b} = 3 - \left(I * \left(\frac{3}{I_{\max}}\right)\right). \tag{5}$$

Positions of the neighborhood dingoes can be illustrated using Figure 2, which is represented using a two-dimensional

(a)

(b)

(c)

(d)

(e)

Figure 1: Hunting behavior of dingoes: (a) dingo; (b) prey; (c) chasing and approaching; (b–d) harassing and encircling; (e) attack.

Table 2: Nomenclature of equations (1)–(5).

Elements	Description
$\overrightarrow{D}_d$	Distance between the dingo and prey
$\overrightarrow{P}^p$	Position vector (prey)
$\overrightarrow{P}$	Position vector (dingo)
$\overrightarrow{A}$	Coefficient vector
$\overrightarrow{B}$	Coefficient vector
$\overrightarrow{a}_1$	Random vector in [0,1]
$\overrightarrow{a}_2$	Random vector in [0,1]
b	Linearly decrement from 3 to 0 at every iteration
‖	Absolute value and multiplication with vectors
I	$1, 2, 3 \ldots, I_{\max}$
$I_{\max}$	Maximum no. of iteration

position vector. According to the position of the prey (P^*, Q^*), a dingo can update its position at the position of (P, Q). All the possible locations are marked in the diagram around the best agent, concerning the current location by changing the value of $\overrightarrow{A}$ and $\overrightarrow{B}$ vectors. For example, by setting $\overrightarrow{A} = (1, 0)$ and $\overrightarrow{B} = (1, 1)$, dingo can be reached at $(P^* - P, Q^*)$. It can also be represented using 3-dimensional space as in Figure 3. It is clearly illustrated how random vectors a1 and a2 enable dingoes to enter any place between the points. Equations (1) and (2) help dingoes to change their locations inside the quest area around the prey in any random location. To reach a search space with N dimensions, the same equations can be used and the dingo will move in hypercubes around the best result got so far.

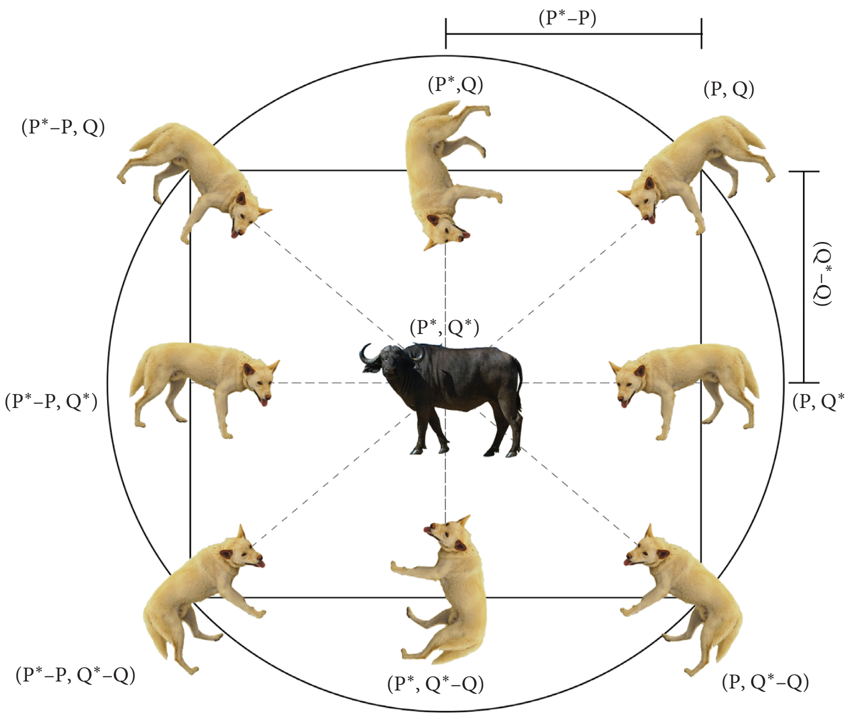


Figure 2: 2D position vectors of dingoes.

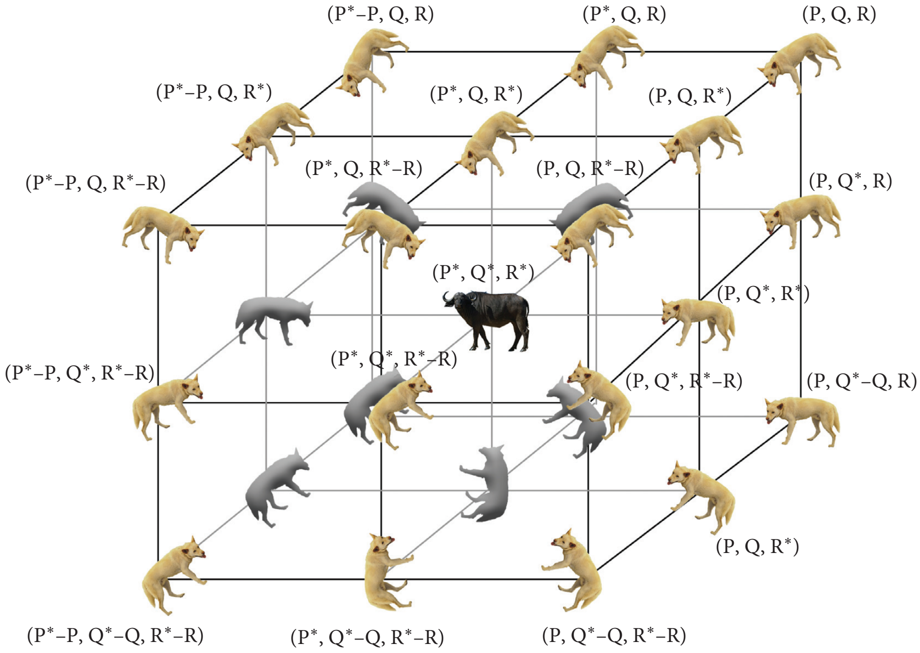


Figure 3: 3D position vectors of dingoes.

3.2.2. Hunting. However, in the search space according to the concept, agents do not normally have a calculation of the position of the prey (optimum). Designing the dingoes hunting plan mathematically, we assume that all the pack members including alpha, beta, and others have good knowledge about the potential location of prey. The alpha dingo always commands the hunting. However, sometimes beta and other dingoes might also participate in hunting. Hence, we consider the first two best values achieved so far. As per the location of the best search agent, other dingoes also need to update their position. According to the discussion, equations (6)–(14) are modeled in this concern. Also, a detailed description of the nomenclatures used in the equation is provided in Table 3.

$$\vec{D}_\alpha = \left|\vec{A}_1 \cdot \vec{P}_\alpha - \vec{P}\right|, \tag{6}$$

$$\vec{D}_\beta = \left|\vec{A}_2 \cdot \vec{P}_\beta - \vec{P}\right|, \tag{7}$$

$$\vec{D}_o = \left|\vec{A}_3 \cdot \vec{P}_o - \vec{P}\right|, \tag{8}$$

$$\vec{P}_1 = \left|\vec{P}_\alpha - \vec{B} \cdot \vec{D}_\alpha\right|, \tag{9}$$

$$\vec{P}_2 = \left|\vec{P}_\beta - \vec{B} \cdot \vec{D}_\beta\right|, \tag{10}$$

$$\vec{P}_3 = \left|\vec{P}_o - \vec{B} \cdot \vec{D}_o\right|. \tag{11}$$

To calculate the intensity of each dingo, following equations are being used:

$$\vec{I}_\alpha = \log\left(\frac{1}{F_\alpha - (1E - 100)} + 1\right), \tag{12}$$

$$\vec{I}_\beta = \log\left(\frac{1}{F_\beta - (1E - 100)} + 1\right), \tag{13}$$

$$\vec{I}_o = \log\left(\frac{1}{F_o - (1E - 100)} + 1\right). \tag{14}$$

The position update in the 2D search space is described in Figure 4. In this, we can easily visualize the position updated of alpha, beta, and other dingoes. It can also be understood that dingoes (alpha, beta, and others) update their positions randomly and calculate the position of the prey in the search space.

Table 3: Description of equations (1)–(4).

Elements and description	
$\vec{D}_d$	Distance between the dingo and prey
$\vec{P}_p$	Positioning of a prey vector
$\vec{P}$	Positioning of a dingo vector
$\vec{A}$	Coefficient vector
$\vec{B}$	Coefficient vector
$\vec{a}_1$	Random vector in [0, 1]
$\vec{a}_2$	Random vector in [0, 1]
$\vec{b}$	Linear decrease over the course of iterations
$\lVert$	Absolute value
I	$1, 2, 3, \ldots, I_{\max}$
$I_{\max}$	Maximum no. of iteration
F_α	Fitness value of alpha (α) dingo
F_β	Fitness value of alpha (β) dingo
F_o	Fitness value of other dingoes

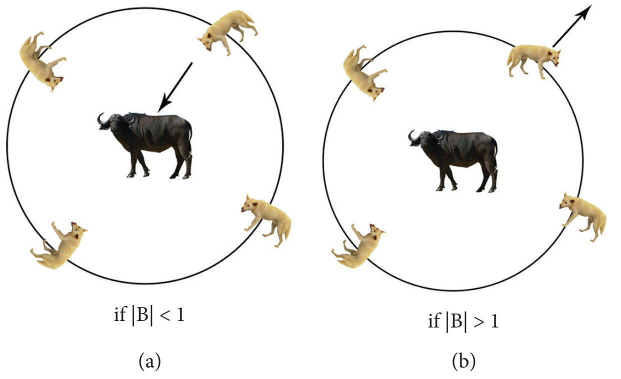


Figure 4: 2D position vectors of dingoes.

3.2.3. Attacking Prey. If there is no position update, it means dingo finished the hunt by attacking the prey. To mathematically formulate the strategy, the value of $\vec{b}$ is decreased linearly. Point to be noted is that the alteration range of $\vec{D}_\alpha$ is also decreased by $\vec{b}$. This may also be known as $\vec{D}\alpha$ which is a random value in the [-3b, 3b] interval where b is reduced from 3 to 0 during iterations. When random values of $\vec{D}\alpha$ are in [1, 1], a search agent's next position may be in any position between its current and the prey's position.

The proposed encircling method does indeed reveal exploration to some extent; however, to accentuate exploration, DOX requires more operators. Figure 4 is the illustration that shows that <1 drives the dingo to strike the prey. The DOX assists its quest agents in changing their location based on the positioning of α, β, others, and the targeted prey. Even, with these operators, the DOX can inactivate local solutions.

3.2.4. Searching. Dingoes hunt for the prey mostly according to the pack's location. They always travel forward to hunt for and strike predators. Accordingly, $\vec{B}$ is used for random values where, if the value is less than −1, it means prey is moving away from the search agent, but if the value is greater than 1, it means pack approaches the prey. This intervention helps the DOX to scan the targets globally. To find out which prey is better suited, Figure 4 reflects that 1 lets dingoes avoid the predators. Another component of DOX that makes exploration likely is $\vec{A}$. In equation (3), the vector $\vec{A}$ can produce any random number between [0, 3] for arbitrary prey weights. DOX represents a stochastic function, regarded as vector ≤1 precedes than ≥1 to explain the impact of the gap formulated in equation (1).

This would be good for searching and avoidance of nearby optima. Depending on a dingo's location, it will arbitrarily agree on the prey's value and make it necessary to meet dingo rigidly or beyond. Intentionally, we used $\vec{A}$ to provide stochastic exploration values from the initial to the final iterations. This method is effective in protecting the solution from local optima. Eventually, the DOX terminates itself whenever it meets the termination criteria.

3.3. Optimization Algorithm. The DOX pseudo code demonstrates how it can solve optimization problems, and several points can be mentioned in Algorithm 1. Here, stopping criteria belong to the maximum number of iterations. The dingo optimization algorithm process is discussed in the following steps.

```
Input: The population of dingoes $D_n$ (n = 1, 2, . . ., n)
Output: The best dingo. (Here, the best values is minimum)
Generate initial search agents $D_{\text{in}}$
Initialize the value of $\vec{b}$, $\vec{A}$ and $\vec{B}$.
While Termination condition not reached do
Evaluate each dingo's fitness and intensity cost.
$D_\alpha$ = Dingo with the best search
$D_\beta$ = Dingo with the second best search
$D_o$ = Dingoes search results afterwords
Iteration1
repeat
for i = 1: $D_{\text{in}}$ do
    Renew the latest search agent status.
endfor
  Estimate the fitness and intensity cost of dingoes.
  Record the value of $S_\alpha$, $S_\beta$, $S_\delta$
  Record the value of $\vec{b}$, $\vec{A}$, and $\vec{B}$.
  Iteration = Iteration +1
  check if, Iteration ⩾ Stopping criteria
  output
endwhile
```

Algorithm 1: Dingo optimizer (DOX).

4. Results and Discussion

4.1. Experimental Setup. The overall simulation is done in MATLAB, taking into account the various parameters which will be explained in the setup of the simulation. The proposed DOX is implemented in Windows 10 with memory 8 GB RAM and processor Intel CPU 2.50 GHz. To generate the solutions for each predefined benchmark function, the DOX uses 25 individual runs and each run applies 500 times of iterations.

4.2. Results. The DOX is the algorithm that has been tested on 23 well-known test functions [56]. These test functions are the classical functions used by various research groups. The results of the model being suggested using the dingo algorithm are shown as follows. Such testing functions were chosen to align our experiments with the current metaheuristics despite the convenience. These benchmark functions are defined in appendix Tables 4–6 where Dim indicates the function size, Range is the search space boundary, and the maximum is $f_{\max}$. Figure 5 represents comparison of convergence curves of DOX obtained in some of the benchmark problems. The benchmark functions are typical functions of minimization and can be segmented into four different categories.

5. DOX for Engineering Problems

Here, DOX was tested on a small engineering design problem called a pressure vessel. Such kind of problem is having different design constraints to handle the optimization.

5.1. Pressure Vessel Design. This is the problem which is used by many researchers to validate the solution that was proposed by Kannan and Kramer [57] to minimize the total cost, including cost of material, forming, and welding of cylindrical vessel which are capped at both ends by hemispherical heads.

(1) p1: thickness of the shell

(2) p2: thickness of the head

(3) p3: inner radius

(4) p4: length of the cylindrical section without considering the head

The mathematical formulation of this problem is formulated as follows.

Consider $p = [p_1 p_2 p_3 p_4]$.

Minimize the following function:

$$f(p) = 0.6224 p_1 p_3 p_4 + 1.7781 p_2 p_3^2 + 3.1661 p_1^2 p_4 + 19.84 p_1^2 p_3, \tag{15}$$

Subject to

$$\begin{aligned} f_1(z) &= -f_1 + 0.0193 p_3 \leq 0, \\ f_2(z) &= -f_3 + 0.00954 f_3 \leq 0, \\ f_3(z) &= -\Pi f_3^2 f_4 - \frac{4}{3} \Pi f_3^3 + 1296000 \leq 0, \\ f_4(z) &= f_4 - 240 \leq 0. \end{aligned} \tag{16}$$

Variable range is as follows:

$$\begin{aligned} 0 &\leq f1 \leq 99, \\ 0 &\leq f2 \leq 99, \\ 0 &\leq f3 \leq 200, \\ 0 &\leq f4 \leq 200. \end{aligned} \tag{17}$$

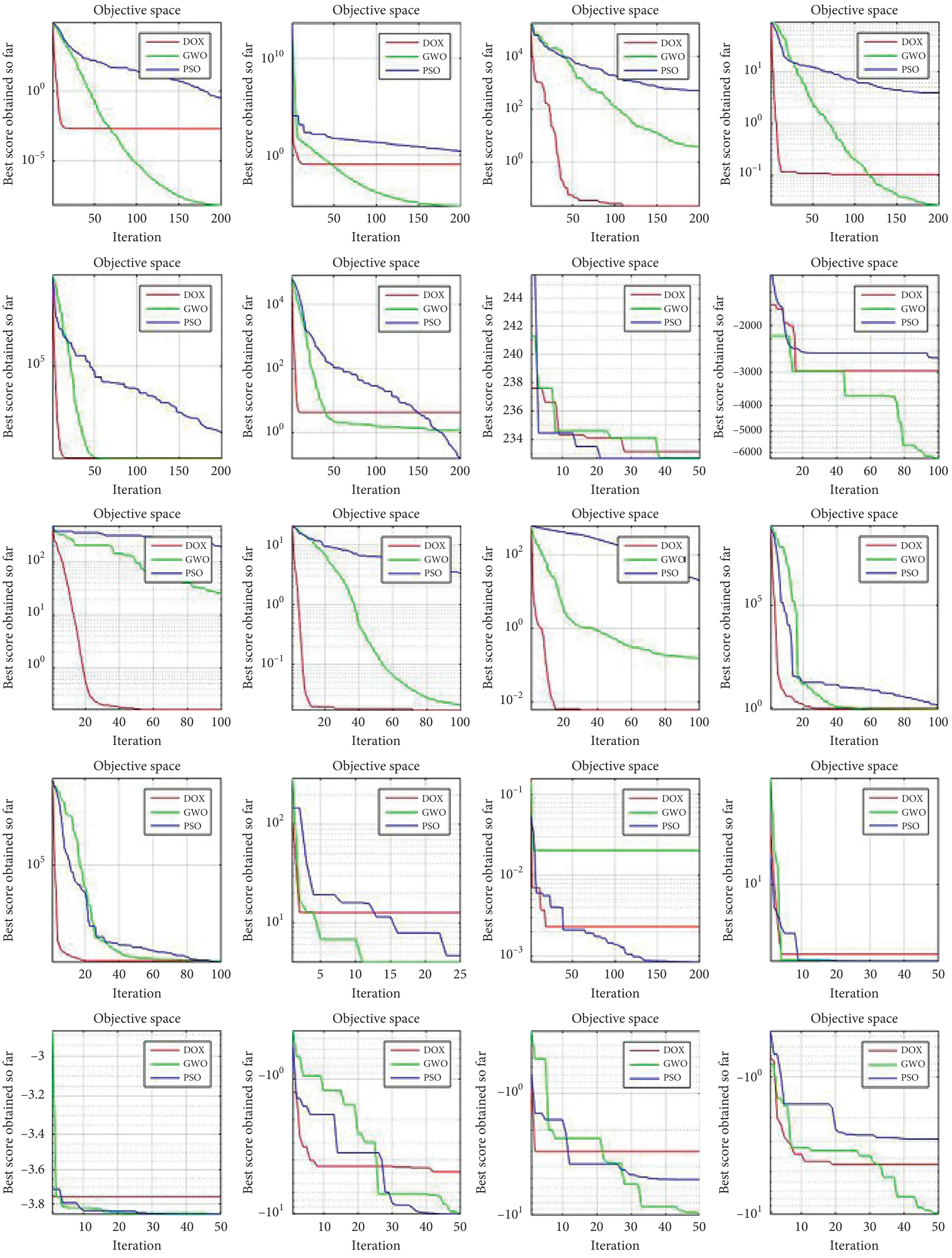


Figure 5: Comparison of convergence curves of DOX obtained in some of the benchmark problems.

Table 4: Unimodal functions.

Function	Dim	Range	$f_{\min}$
$F_1(k) = \sum_{j=1}^{m} k_j^2$	30	$[-100, 100]$	0
$F_2(k) = \sum_{j=1}^{m} \lvert k\rvert + \prod_{j=1}^{m} \lvert k\rvert$	30	$[-10, 10]$	0
$F_3(k) = \sum_{j=1}^{m} (\sum_{j=1}^{m} k_j^2)^2$	30	$[-100, 100]$	0
$F_4(k) = \max_j \lvert k_j\rvert,\ 1 \le j \le m$	30	$[-100, 100]$	0
$F_5(k) = \sum_{j=1}^{m-1} [100(k_{j+1} - k_j^2)^2 + (k_j - 1)^2]$	30	$[-30, 30]$	0
$F_6(k) = \sum_{j=1}^{m} (\lvert k_i + 0.5\rvert)^2$	30	$[-100, 100]$	0
$F_7(k) = \sum_{j=1}^{m} ik_i^4 + \text{random}[0, 1]$	30	$[-1.28, 1.28]$	0

Table 5: Multimodal benchmark functions.

Function	Dim	Range	$f_{\min}$
$F_8(k) = \sum_{j=1}^{m} -k_j \sin(\sqrt{\lvert k_j\rvert})$	30	$[-500, 500]$	-418.982×5
$F_9(k) = \sum_{j=1}^{m} [k_j^2 - 10\cos(2\pi k_j) + 10]$	30	$[-32, 32]$	0
$F_{10}(k) = -20\exp(-0.2\sqrt{(1/m)\sum_{j=1}^{m} k_j^2}) - \exp((1/m)\sum_{j=1}^{m}\cos(2\pi k_j)) + 20 + e$	30	$[-100, 100]$	0
$F_{11}(k) = (1/4000)\sum_{j=1}^{m} k_j^2 - \prod_{j=1}^{m}\cos(k_j/\sqrt{j}) + 1$	30	$[-600, 600]$	0
$F_{12}(k) = (\pi/m)\left\{10\sin(\pi y_1)\sum_{j=1}^{m}(y_1 - 1)^2[1 + 10\sin^2(\pi y_{j+1})] + (y_m - 1)^2\right\}$ $+\sum_{j=1}^{m} u(k_j, 10, 100, 4) y_j = 1 + (k_j + 1/4)$ $u(k_j, a, l, n) = \begin{cases} l(k_j - a)^n, & k_j > a. \\ 0, & -a < k_j < a. \\ l(-k_j - a)^n, & k_j < a \end{cases}$	30	$[-50, 50]$	0
$F_{13}(k) = 0.1\left\{\begin{matrix}\sin^2(3\pi k_j) + \sum_{j=1}^{m}(k_j - 1)^2[1 + \sin^2(3\pi k_j + 1)] \\ +(k_j - 1)^2[1 + \sin^2(2\pi k_j)]\end{matrix}\right\} + \sum_{j=1}^{m} u(k_j, 5, l00, 4)$	30	$[-50, 50]$	0
$F_{14}(k) = -\sum_{j=1}^{m}(\sin(k_j)\cdot \sin(j\cdot k_j^2/\pi))^{2n},\ n = 10$	30	$[-0, \pi]$	-4.687
$F_{15}(k) = [e^{-\sum_{j=1}^{m}(k_j/\beta)^{2n}} - 2e^{-\sum_{j=1}^{m} k_j^2}]\cdot\prod_{j=1}^{m}\cos^2 k_j,\ n = 5$	30	$[-20, 20]$	0
$F_{16}(k) = \left\{[\sum_{j=1}^{m}\sin^2(k_j)] - \exp(-\sum_{j=1}^{m} k_j^2)\right\}\cdot \exp[-\sum_{j=1}^{m}\sin^2\sqrt{\lvert k_j\rvert}]$	30	$[-10, 10]$	-1

This problem has been popular among researchers in various studies.

Table 4 is a comparison of the best optimal solution for DOX and other documented approaches, such as GWO and PSO. According to this table, DOX will find an optimum design at a minimal rate. Table 4 shows DOX comparison of the historical effects of the issue of construction of pressure vessels. The DOX results work better than any other algorithm in terms of the best optimal solution.

The DOX algorithm obtained the near optimal solution in the initial steps of iterations and achieved better results than other optimization methods for pressure vessel problem.

The comparison of best optimal solution among several algorithms is given in Table 4. This problem has been tested with different optimization methods such as GWO and PSO. The comparison for the best solution obtained by such algorithms is presented.

6. Statistical Testing

The ANOVA test was performed to test whether the outcomes obtained from the proposed algorithms vary statistically substantially from the findings of other algorithms. We took 30 as the sample size for the ANOVA test. We used 95% confidence for the ANOVA test. The results of the ANOVA test for the benchmark functions are shown in Table 8. The findings demonstrate that the DOX is statistically important relative to other rival algorithms.

Table 6: Fixed-dimension multimodal benchmark functions.

Function	Dim	Range	$f_{\min}$
$F_{14}(k) = ((1/500) + \sum_{1}^{25} (1/j + \sum_{l=1}^{2} (x_i - a_{ij})^6))^- 1$	2	$[-65, 65]$	1
$F_{15}(k) = \sum_{j=1}^{11} [a_j - (x_1 (b_j^2 + b_j x_2)/b_j^2 + b_j x_2 + x_4)]^2$	4	$[-5, 5]$	0.00030
$F_{16}(k) = 4x_i^2 - 2.1x_1^4 + (1/3)z_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$	2	$[-5, 5]$	−1.0316
$F_{17}(k) = (x_2 - (5.1/4\pi^2)x_1^2 + (5/\pi)x_1 - 6)^2 + 10(1 - (1/8\pi))\cos x_1 + 10$	2	$[-5, 5]$	0.398
$F_{18}(k) = [1 + (x_1 + x_2 + 1)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2) + 3x_2^2] \times [30 + (2x_1 - 3x_2)^2 \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	2	$[-2, 2]$	3
$F_{19}(k) = -\sum_{j=1}^{4} c_j \exp(\sum_{l=1}^{3} a_{jl} (x_l - p_{jl})^2)$	3	$[1, 3]$	−3.86
$F_{20}(k) = -\sum_{j=1}^{4} c_j \exp(\sum_{l=1}^{6} a_{jl} (x_l - p_{jl})^2)$	6	$[0, 1]$	−3.32
$F_{21}(k) = -\sum_{j=1}^{5} [(X - a_j)(X - a_j)^T + c_l]^{-1}$	4	$[0, 10]$	−10.1532
$F_{22}(k) = -\sum_{j=1}^{7} [(X - a_j)(X - a_j)^T + c_l]^{-1}$	4	$[0, 10]$	−10.1532
$F_{23}(k) = -\sum_{j=1}^{10} [(X - a_j)(X - a_j)^T + c_l]^{-1}$	4	$[0, 10]$	−10.1532

Table 7: Results comparison for pressure vessel design problem.

Algo.	p_1	p_2	p_3	p_4	Optimum
DOX	0.7782	0.3848	40.3150	200.0000	5885.5773
GWO	0.7790	0.3846	40.3277	199.6502	5889.3689
PSO	0.7789	0.3847	40.3209	200.0000	5891.3879

Table 8: Unimodal functions.

F	*P* value	DOX	GWO	PSO
F1	1.91E−64	GWO,PSO	DOX,PSO	DOX,GWO
F2	3.62E−65	GWO,PSO	DOX,PSO	DOX,GWO
F3	3.57E−36	GWO,PSO	DOX,PSO	DOX,GWO
F4	2.87E−22	GWO,PSO	DOX,PSO	DOX,GWO
F5	2.15E−24	GWO,PSO	DOX,PSO	DOX,GWO
F6	1.94E−54	GWO,PSO	DOX,PSO	DOX,GWO
F7	1.56E−13	GWO,PSO	DOX,PSO	DOX,GWO
F8	1.59E−43	GWO,PSO	DOX,PSO	DOX,GWO
F9	1.88E−87	GWO,PSO	DOX,PSO	DOX,GWO
F10	1.29E−34	GWO,PSO	DOX,PSO	DOX,GWO
F11	1.87E−35	GWO,PSO	DOX,PSO	DOX,GWO
F12	2.36E−23	GWO,PSO	DOX,PSO	DOX,GWO
F13	1.91E−98	GWO,PSO	DOX,PSO	DOX,GWO
F14	6.35E−36	GWO,PSO	DOX,PSO	DOX,GWO
F15	1.14E−07	GWO,PSO	DOX,PSO	DOX,GWO
F16	2.63E−35	GWO,PSO	DOX,PSO	DOX,GWO
F17	1.83E−58	GWO,PSO	DOX,PSO	DOX,GWO
F18	4.61E−36	GWO,PSO	DOX,PSO	DOX,GWO
F19	9.54E−15	GWO,PSO	DOX,PSO	DOX,GWO
F20	1.99E−73	GWO,PSO	DOX,PSO	DOX,GWO
F21	2.54E−62	GWO,PSO	DOX,PSO	DOX,GWO
F22	8.11E−06	GWO,PSO	DOX,PSO	DOX,GWO
F23	2.70E−18	GWO,PSO	DOX,PSO	DOX,GWO
F24	5.55E−36	GWO,PSO	DOX,PSO	DOX,GWO
F25	1.22E−87	GWO,PSO	DOX,PSO	DOX,GWO
F26	7.66E−45	GWO,PSO	DOX,PSO	DOX,GWO
F27	2.48E−69	GWO,PSO	DOX,PSO	DOX,GWO
F28	2.31E−43	GWO,PSO	DOX,PSO	DOX,GWO
F29	1.14E−47	GWO,PSO	DOX,PSO	DOX,GWO

7. Conclusion and Future Scope

As per the comparison of DOX with other popular metaheuristic algorithms such as PSO and DSO, DOX provides well competitive outcomes as presented in the results. The DOX is analyzed for the exploration and exploitation activity of agents using twenty-three test functions. The concise results, which are based on comparative analysis between the proposed DOX and other optimization algorithms, demonstrate that the approach suggested will cope with different kinds of constraints and provide stronger alternatives than any other optimizer. The suggested methodology is inspired by the real-life problems, which required less computational or mathematical efforts to find the best available optima.

Some other major findings may be preferred for future studies. DOX may be used to address various technological problems. Multiobjective problems can be solved as another future contribution as MODOX. Binary DOX might also be other benchmarks to expand this algorithm.

Appendix

A. Benchmark Functions

A.1. Unimodel Functions. The list of the unimodal test functions (F1–F7) is given in Table 4.

A.2. Multimodal Functions. The list of the multimodal test functions (F8–F16) is given in Table 5.

A.3. Fixed-Dimension Multimodal Functions. The list of the fixed-dimension multimodal test functions (F14–F23) is given in Table 6.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] H. Ali, W. Shahzad, and F. A. Khan, "Energy-efficient clustering in mobile ad-hoc networks using multi-objective particle swarm optimization," *Applied Soft Computing*, vol. 12, no. 7, pp. 1913–1928, 2012.

[2] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: nsga-ii," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[3] G. Qi, H. Wang, M. Haner, C. Weng, S. Chen, and Z. Zhu, "Convolutional neural network based detection and judgement of environmental obstacle in vehicle operation," *CAAI Transactions on Intelligence Technology*, vol. 4, no. 2, pp. 80–91, 2019.

[4] C. Zhu, W. Yan, X. Cai, S. Liu, T. H. Li, and G. Li, "Neural saliency algorithm guide bi-directional visual perception style transfer," *CAAI Transactions on Intelligence Technology*, vol. 5, no. 1, pp. 1–8, 2020.

[5] T. Wiens, "Engine speed reduction for hydraulic machinery using predictive algorithms," *International Journal of Hydromechatronics*, vol. 2, no. 1, pp. 16–31, 2019.

[6] X. Xin Yao, Y. Yong Liu, and G. Guangming Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 82–102, 1999.

[7] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley Longman Publishing Co., Inc., Boston, Massachusetts, USA, 1989.

[8] X. Xue, J. Lu, and J. Chen, "Using NSGA-III for optimising biomedical ontology alignment," *CAAI Transactions on Intelligence Technology*, vol. 4, no. 3, pp. 135–141, 2019.

[9] W. Banzhaf, F. D. Francone, R. E. Keller, and P. Nordin, *Genetic Programming: An Introduction: On the Automatic Evolution of Computer Programs and its Applications*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1998.

[10] C. M. Fonseca and P. J. Fleming, "Genetic algorithms for multiobjective optimization: formulationdiscussion and generalization," in *Proceedings of the 5th International Conference on Genetic Algorithms*, pp. 416–423, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, June 1993.

[11] R. Storn and K. Price, "Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[12] F. Xue, A. C. Sanderson, and R. J. Graves, "Multi-objective differential evolution - algorithm, convergence analysis, and applications," *IEEE Congress on Evolutionary Computation*, vol. 1, pp. 743–750, 2005.

[13] H.-G. Beyer and H.-P. Schwefel, "Evolution strategies –a comprehensive introduction," *Natural Computing*, vol. 1, no. 1, pp. 3–52, 2002.

[14] X. Hu, C. A. C. Coello, and Z. Huang, "A new multi-objective evolutionary algorithm: neighbourhood exploring evolution strategy," *Engineering Optimization*, vol. 37, no. 4, pp. 351–379, 2005.

[15] T. Jansen and C. Zarges, "Artificial immune systems for optimisation," in *Proceedings of the Fourteenth International Conference on Genetic and Evolutionary Computation Conference Companion - GECCO Companion '12*, pp. 1059–1078, Association for Computing Machinery, New York, NY, USA, July 2012.

[16] F. Campelo, F. G. Guimarães, and H. Igarashi, "Overview of artificial immune systems for multi-objective optimization," in *Proceedings of the 4th International Conference on Evolutionary Multi-Criterion Optimization, EMO'07*, pp. 937–951, Springer-Verlag, Berlin, Heidelberg, March 2007.

[17] K. M. Passino, "Bacterial foraging optimization," *International Journal of Swarm Intelligence Research*, vol. 1, no. 1, pp. 1–16, 2010.

[18] "Multi-swarm cooperative multi-objective bacterial foraging optimisation," *International Journal of Bio-Inspired Computation*, vol. 13, no. 1, pp. 21–31, 2019.

[19] Z. Chelly and Z. Elouedi, "A survey of the dendritic cell algorithm," *Knowledge and Information Systems*, vol. 48, no. 3, pp. 505–535, 2016.

[20] A. Gandomi and A. Alavi: Krill Herd Algorithm.

[21] P. J. M. Laarhoven and E. H. L. Aarts, *Simulated Annealing: Theory and Applications*, Kluwer Academic Publishers, New York, NY, USA, 1987.

[22] N. Krasnogor, *Memetic Algorithms*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 905–935, 2012.

[23] M. Eusuff, K. Lansey, and F. Pasha, “Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization,” *Engineering Optimization*, vol. 38, no. 2, pp. 129–154, 2006.

[24] M. Dorigo, M. Birattari, and T. Stützle, “Ant colony optimization,” *IEEE Computational Intelligence Magazine*, vol. 1, no. 4, pp. 28–39, 2006.

[25] J. Kennedy and R. Eberhart, “Particle swarm optimization,” *Proceedings of ICNN’95 - International Conference on Neural Networks*, vol. 4, pp. 1942–1948, 1995.

[26] D. Karaboga and B. Basturk, “Artificial bee colony (abc) optimization algorithm for solving constrained optimization problems,” in *Foundations of Fuzzy Logic and Soft Computing*, P. Melin, O. Castillo, L. T. Aguilar, J. Kacprzyk, and W. Pedrycz, Eds., pp. 789–798, Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[27] M. Neshat, A. Adeli, G. Sepidnam, M. Sargolzaei, and A. Najaran Toosi, “A review of artificial fish swarm optimization methods and applications,” *International Journal on Smart Sensing and Intelligent Systems*, vol. 5, no. 1, pp. 108–148, 2012.

[28] G. wolf optimizer, “Grey wolf optimizer,” *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014, https://doi.org/10.1016/j.advengsoft.2013.12.007.

[29] S. Mirjalili, “Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems,” *Neural Computing and Applications*, vol. 27, no. 4, pp. 1053–1073, 2016.

[30] J. Ma, H. Y. Chen, R. Su, Y. Wang, S. Zhang, and S. Shan, “Improved firefly algorithm and its application,” *Proceedings of the 4th International Conference on Crowd Science and Engineering*, Association for Computing Machinery, in *Proceedings of the 4th International Conference on Crowd Science and Engineering, ICCSE’19*, pp. 180–185, October 2019.

[31] X.-S. Yang, “Firefly algorithms for multimodal optimization,” 2010, https://arxiv.org/abs/1003.1466.

[32] S. Mirjalili and A. Lewis, “The whale optimization algorithm,” *Advances in Engineering Software*, vol. 95, no. C, pp. 51–67, 2016.

[33] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, “Gsa: a gravitational search algorithm,” *Information Sciences*, vol. 179, no. 13, pp. 2232–2248, 2009.

[34] X.-S. Y. Seyedali Mirjalili and S. M. Mirjalili, “Binary bat algorithm,” *Neural Computing and Applications*, vol. 25, no. 3-4, pp. 663–681, 2014.

[35] Y. C. Shih, “A cuckoo search algorithm: effects of coevolution and application in the development of distributed layouts,” *Journal of Algorithms & Computational Technology*, vol. 13, Article ID 1748302619889523, 2019.

[36] S.-C. Chu, P.-W. Tsai, and J.-S. Pan, “Cat swarm optimization,” *Lecture Notes in Computer Science*, in *Proceedings of the 9th Pacific Rim International Conference on Artificial Intelligence, PRICAI’06*, Springer-Verlag, Berlin, Heidelberg, pp. 854–858, August 2006.

[37] V. Kumar, J. K. Chhabra, and D. Kumar, “Parameter adaptive harmony search algorithm for unimodal and multimodal optimization problems,” *Journal of Computational Science*, vol. 5, no. 2, pp. 144–155, 2014.

[38] L. Wang, Y. Mao, Q. Niu, and M. Fei, “A multi-objective binary harmony search algorithm,” in *Lecture Notes in Computer Science Advances in Swarm Intelligence*, Y. Tan, Y. Shi, Y. Chai, and G. Wang, Eds., Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 74–81, 2011.

[39] Y. Fang, G. Liu, He Yi, and Y. Qiu, “Tabu search algorithm based on insertion method,” in *Proceedings of the international conference on neural networks and signal processing*, pp. 420–423, Nanjing, China, December 2003.

[40] S. Carcangiu, A. Fanni, and A. Montisci, “Multiobjective tabu search algorithms for optimal design of electromagnetic devices,” *IEEE Transactions on Magnetics*, vol. 44, no. 6, pp. 970–973, 2008.

[41] V. Kumar, J. K. Chhabra, and D. Kumar, “Parameter adaptive harmony search algorithm for unimodal and multimodal optimization problems,” *Journal of Computational Science*, vol. 5, no. 2, pp. 144–155, 2014.

[42] S. He, Q. H. Wu, and J. R. Saunders, “Group search optimizer: an optimization algorithm inspired by animal searching behavior,” *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 973–990, 2009.

[43] N. Ghorbani and E. Babaei, “Exchange market algorithm,” *Applied Soft Computing*, vol. 19, pp. 177–187, 2014.

[44] E. Atashpaz-Gargari and C. Lucas, “Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition,” in *Proceedings of the 2007 IEEE Congress on Evolutionary Computation*, pp. 4661–4667, Singapore, September 2007.

[45] N. Moosavian, “Soccer league competition algorithm for solving knapsack problems,” *Swarm and Evolutionary Computation*, vol. 20, pp. 14–22, 2015.

[46] A. Husseinzadeh Kashan, “League championship algorithm (lca): an algorithm for global optimization inspired by sport championships,” *Applied Soft Computing*, vol. 16, pp. 171–200, 2014.

[47] F. Ramezani and S. Lotfi, “Social-based algorithm (sba),” *Applied Soft Computing*, vol. 13, no. 5, pp. 2837–2856, 2013.

[48] Y. Tan, C. Yu, S. Zheng, and K. Ding, “Introduction to fireworks algorithm,” *International Journal of Swarm Intelligence Research*, vol. 4, no. 4, pp. 39–70, 2013.

[49] A. Kaveh and V. Mahdavi Dahoei, Colliding Bodies Optimization, 2015.

[50] A. Gandomi, Interior Search Algorithm, 2014.

[51] W. Zhao, L. Wang, and Z. Zhang, “Artificial ecosystem-based optimization: a novel nature-inspired meta-heuristic algorithm,” *Neural Computing and Applications*, vol. 32, pp. 1–43, 2020.

[52] L. Benasla, A. Belmadani, and R. Mostefa, “Spiral optimization algorithm for solving combined economic and emission dispatch,” *International Journal of Electrical Power and Energy Systems*, vol. 62, pp. 163–174, 2014.

[53] E. Bogar and S. Beyhan, “Adolescent identity search algorithm (aisa): a novel metaheuristic approach for solving optimization problems,” *Applied Soft Computing*, vol. 95, Article ID 106503, 2020.

[54] R. Wang, H. Yu, G. Wang, G. Zhang, and W. Wang, “Study on the dynamic and static characteristics of gas static thrust bearing with micro-hole restrictors,” *International Journal of Hydromechatronics*, vol. 2, p. 189, 2019.

[55] S. Osterland and J. Weber, “Analytical analysis of single-stage pressure relief valves,” *International Journal of Hydromechatronics*, vol. 2, p. 32, 2019.

[56] C. S. Marcin Molga, Test Functions for Optimization Needs, 2005.

[57] B. Kannan and S. Kramer, “An augmented Lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design,” *Journal of Mechanical Design*, vol. 116, pp. 405–411, 1994.

Hindawi
Mathematical Problems in Engineering
Volume 2021, Article ID 1743673, 12 pages
https://doi.org/10.1155/2021/1743673

Research Article

Computationally Efficient Approximations Using Adaptive Weighting Coefficients for Solving Structural Optimization Problems

Guirong Dong,[1] Chengyang Liu,[2] Yijie Liu,[3] Ling Wu,[1] Xiaoan Mao,[4] and Dianzi Liu[2,5]

[1]*Faculty of Printing, Packaging Engineering and Digital Media Technology, Xi'an University of Technology, Xi'an, China*
[2]*School of Engineering, University of East Anglia, Norwich, UK*
[3]*Department of Engineering Mechanics, Guangzhou University, Guangzhou, China*
[4]*Faculty of Engineering, University of Leeds, Leeds LS2 9JT, UK*
[5]*School of Mechanical Engineering, Xi'an University of Science and Technology, Xi'an, China*

Correspondence should be addressed to Dianzi Liu; dianzi.liu@uea.ac.uk

Received 24 April 2020; Revised 27 June 2020; Accepted 16 September 2020; Published 11 March 2021

Academic Editor: Manjit Kaur

With rapid development of advanced manufacturing technologies and high demands for innovative lightweight constructions to mitigate the environmental and economic impacts, design optimization has attracted increasing attention in many engineering subjects, such as civil, structural, aerospace, automotive, and energy engineering. For nonconvex nonlinear constrained optimization problems with continuous variables, evaluations of the fitness and constraint functions by means of finite element simulations can be extremely expensive. To address this problem by algorithms with sufficient accuracy as well as less computational cost, an extended multipoint approximation method (EMAM) and an adaptive weighting-coefficient strategy are proposed to efficiently seek the optimum by the integration of metamodels with sequential quadratic programming (SQP). The developed EMAM stems from the principle of the polynomial approximation and assimilates the advantages of Taylor's expansion for improving the suboptimal continuous solution. Results demonstrate the superiority of the proposed EMAM over other evolutionary algorithms (e.g., particle swarm optimization technique, firefly algorithm, genetic algorithm, metaheuristic methods, and other metamodeling techniques) in terms of the computational efficiency and accuracy by four well-established engineering problems. The developed EMAM reduces the number of simulations during the design phase and provides wealth of information for designers to effectively tailor the parameters for optimal solutions with computational efficiency in the simulation-based engineering optimization problems.

1. Introduction

Solving nonlinear optimization problems is a hot issue in design optimization of practical engineering systems. In this class of optimization problems, both the objective function and the constraints are nonlinear and extremely expensive when solved using numerical methods, for example, finite element methods. In order to obtain solutions with high computational accuracy in reasonable time, the hybrid optimization method has become increasingly popular for solving nonlinear optimization problems because it can reduce the computational burden during the analysis by replacing the complex physical systems with the mathematical models and improve the accuracy of the optimal solution with the use of the combined heuristic methods and mathematical programming techniques.

The multipoint approximation method (MAM) [1, 2] is one of the best-known metamodel-based optimization methods with the integration of sequential quadratic programming (SQP) technique, and it replaces the original optimization problem with a sequence of mathematical approximations that use much simpler objective and constraint functions. MAM stemmed from previous work [3, 4] and was further generalized to multipoint approximations

[2, 5]. Recently, Liu and Toropov [6] have implemented the discrete capability into the MAM to solve mixed continuous-discrete optimization problems. In MAM, the process of constructing metamodels can be described as an assembly of multiple metamodels into a single metamodel using linear regression. The coefficients of the model assembly are not weights of the individual models but tuning parameters determined by the least squares method.

In inexpensive engineering design problems, such as a cantilever beam design [7], the hypersonic wing [8], and the wind farm layout design [9], evolutionary algorithms can be a good choice to find globally optimal solutions. Genetic algorithm (GA) [10, 11] is inspired by natural evolution in biology, and the population of candidate solutions experiences a process similar to natural selection and genetic variation. GA has been well-recognized as an optimization method handling nonsmooth and nonlinear problems, where traditional methods generally fail.

Similarly, intrigued by the group foraging such as fish schooling and bird flocking, particle swarm optimization (PSO) technique developed by Kennedy [12] has become one of the dominant optimization algorithms in many fields. The advantages of using various variants of this technique have been validated in the civil engineering applications in terms of convergence rate and success rate. Chen et al. [13] proposed an improved particle swarm optimization- (IPSO-) based form-finding method for suspension bridge design and construction with the test on the design analysis on Yingwuzhou Yangtze River Bridge. Ghamisi and Benediktsson [14] applied integration PSO on feature selection and demonstrated the usefulness on road detection. Meanwhile, PSO has also been widely applied for solving structural mechanics problems [15]. Firefly algorithm (FA), inspired by social behavior of fireflies which is related to the rate and rhythmic of flash [16], is another very promising method. This novel technique has played an important role in the research of truss structures [17] and composite reinforced bridges [18]. Furthermore, the integration of numerical algorithms with neural networks to solve complex problems has been recently investigated by Moghadas et al. [19], Cao et al. [20], and Li et al. [21].

Besides the aforementioned metaheuristic algorithms, metamodel-based algorithms have become increasingly popular in recent years. Widely used metamodels include polynomial regression (PR) [22], radial basis function (RBF) [23], Kriging [24], multivariate adaptive regression splines (MARS) [25], artificial neural networks (ANNs) [26], and support vector regression (SVR) [27]. Currently, there are lots of novel techniques and approaches in the area of metamodel-based optimization. Jones et al. [28] proposed efficient global optimization (EGO), which employs the Kriging metamodel for solving black-box problems. The optimization progress is guided by both the prediction and error estimations. Regis [29] developed COBRA, an efficient solver which makes use of RBF interpolation to approximate objective and constraint functions. A new iterate in COBRA is selected according to the violation of constraints within some small margins. An application of multifidelity metamodel can be found in [30–32], where genetic algorithms are responsible for exploring the global design space.

As stated in Haftka et al. [33], there is still much room for the development of efficient and accurate algorithms to tackle high-fidelity design optimization. To address complex nonlinear optimization problems involving multiscale/multilevel/multidisciplinary analysis within reasonable time, much attention has been paid to the research in relevant fields. Taking into account the above situations, MAM has been gradually developed and become one of the algorithms demonstrating good performance on efficiently solving mid-range constrained engineering optimization problems with the use of the combined heuristic methods and SQP technique. Based on the authors' previous work [2, 6], an extended MAM (EMAM) framework is proposed in this paper to further improve the computational efficiency during the entire simulation process. First, a novel metamodel model inspired by Taylor's expansion technique is developed to effectively construct the approximations. To implement Taylor's expansion metamodel into the framework of MAM, the function of Euclidean distance for the determination of weighting coefficients during the process of approximations is replaced by a proposed strategy for adaptive selection of weighting coefficients. Then, the SQP technique is applied on the approximations to obtain the optimal solutions. The correctness of this enhanced EMAM is validated by comparing with the results from several nonconvex benchmark problems [34, 35], which were successfully solved by researchers in use of the state-of-the-art algorithms, such as genetic algorithms (GAs) [36–38], evolution strategies (ESs) [39], particle swarm optimization (PSO) [40], charged system search (CSS) [41], colliding bodies optimization (CBO) [42], and firefly algorithm (FA) [43]. Although there were some primary tests and rudimentary findings in previous work [44], robust numerical results are found in this paper to extensively demonstrate the advantages and superiority of the developed hybrid algorithm over evolutionary algorithms and MAM in terms of the computational efficiency and accuracy during the optimization process. With the implementation of the effective Taylor's expansion in the current MAM optimization framework, the developed EMAM does not deteriorate the ability to solve the mid-range optimization problems, which is the distinctive feature of MAM optimization framework.

2. Multipoint Approximation Method (MAM)

Based on response surface methodology [22], the multipoint approximation method (MAM) aims at constructing mid-range approximations [4, 5] and is suitable to solve large-scale optimization problems by producing better-quality approximations that are sufficiently accurate in a current trust region and inexpensive in terms of computational costs required for their building. These approximation functions have a relatively small number ($N+1$, where N is the number of design variables) of regression coefficients to be determined and the corresponding least squares problem can be solved easily.

In general, an optimization problem can be formulated as

$$\min \mathbf{F}_0(\mathbf{x}),\ \mathbf{F}_\mathbf{j}(\mathbf{x}) \le 1\,(\mathbf{j} = 1, \ldots, \mathbf{M}), \quad \mathbf{A}_\mathbf{i} \le \mathbf{x}_\mathbf{i} \le \mathbf{B}_\mathbf{i}\,(\mathbf{i} = 1, \ldots, \mathbf{N}), \tag{1}$$

where $\mathbf{x}$ refers to the vector of design variables; $\mathbf{A}_\mathbf{i}$ and $\mathbf{B}_\mathbf{i}$ are the given lower and upper bounds on the design variable $\mathbf{x}_\mathbf{i}$;$\mathbf{N}$ is the total number of the design variables; $\mathbf{F}_0(\mathbf{x})$ is an objective function; $\mathbf{F}_\mathbf{j}(\mathbf{x})$ is the constraint function; and $\mathbf{M}$ is the total number of the constraint functions.

In order to present the detailed physical model using the response functions and reduce the number of calls for the response function evaluations, the MAM replaces the optimization problem by a sequence of approximate optimization problems:

$$\min \widetilde{F}_0^\mathbf{k}(\mathbf{x}), \widetilde{F}_\mathbf{j}^\mathbf{k}(\mathbf{x}) \le 1\,(\mathbf{j} = 1, \ldots, \mathbf{M}), \quad \mathbf{A}_\mathbf{i}^\mathbf{k} \le \mathbf{x}_\mathbf{i} \le \mathbf{B}_\mathbf{i}^\mathbf{k}, \mathbf{A}_\mathbf{i}^\mathbf{k} \ge \mathbf{A}_\mathbf{i}, \mathbf{B}_\mathbf{i}^\mathbf{k} \le \mathbf{B}_\mathbf{i}\,(\mathbf{i} = 1, \ldots, \mathbf{N}), \tag{2}$$

where $\widetilde{F}_0^\mathbf{k}(\mathbf{x})$ and $\widetilde{F}_\mathbf{j}^\mathbf{k}(\mathbf{x})$ are the functions which approximate the functions $\mathbf{F}_0(\mathbf{x})$ and $\mathbf{F}_\mathbf{j}(\mathbf{x})$ defined in equation (1), $\mathbf{A}_\mathbf{i}^\mathbf{k}$ and $\mathbf{B}_\mathbf{i}^\mathbf{k}$ are the side constraints of a trust subregion, and k is the iteration number.

The selection strategy of the approximate response functions $\widetilde{F}_\mathbf{j}^\mathbf{k}(\mathbf{x})\,(\mathbf{j} = 0, \ldots, \mathbf{M})$ outlines that their evaluations are inexpensive as compared to the evaluations of the actual response functions $\mathbf{F}_\mathbf{j}(\mathbf{x})$ and are intended to be adequate in a current trust region. This is achieved by appropriate planning of numerical experiments and use of the trust region defined by the side constraints $\mathbf{A}_\mathbf{i}^\mathbf{k}$ and $\mathbf{B}_\mathbf{i}^\mathbf{k}$.

In the present work, constructing the metamodels for the objective and constraint functions includes two stages. In the first stage, the parameters a_j involved in building a single metamodel ϕ_l are formulated as follows:

$$\sum_{\mathbf{p}=1}^{\mathbf{P}} \mathbf{w}_\mathbf{p} \left[\mathbf{F}(\mathbf{x}_\mathbf{p}) - \boldsymbol{\varphi}_\mathbf{l}(\mathbf{x}_\mathbf{p}, \mathbf{a}_\mathbf{j})\right]^2 \longrightarrow \min, \tag{3}$$

where F is the function to be approximated; P means the total number of sampling points; the coefficient w_p denotes the weight of each point $\mathbf{x}_p$; in other words, it represents the inequality of each sample point in the sample space [45]; and a_j indicates the tuning parameter associated with the specific metamodel $\varphi_\mathbf{l}$ in equation (4), and it is determined by the weighted least squares method.

$$\begin{aligned}
\varphi_1(\mathbf{x}) &= a_0 + \sum_{i=1}^{N} a_i x_i, \\
\varphi_2(\mathbf{x}) &= a_0 + \sum_{i=1}^{N} a_i x_i^2, \\
\varphi_3(\mathbf{x}) &= a_0 + \sum_{i=1}^{N} \frac{a_i}{x_i}, \\
\varphi_4(\mathbf{x}) &= a_0 + \sum_{i=1}^{N} \frac{a_i}{x_i^2}, \\
\varphi_5(\mathbf{x}) &= a_0 \prod_{i=1}^{N} x_i^{a_i}.
\end{aligned} \tag{4}$$

In the second stage, different approximate models are assembled into one metamodel described by equations (5) and (6). Also, equation (5) is built in the same manner as equation (3). It should be noted here that the design of experiments is fixed when a different approximate model ϕ_l is constructed.

$$\sum_{\mathbf{p}=1}^{\mathbf{P}} \mathbf{w}_\mathbf{p} \left[\mathbf{F}(\mathbf{x}_\mathbf{p}) - \widetilde{F}(\mathbf{x}_\mathbf{p}, \mathbf{b}_\mathbf{l})\right]^2 \longrightarrow \mathbf{min}, \tag{5}$$

where the assembly metamodel $\widetilde{\mathbf{F}}$is expressed as

$$\widetilde{F}(\mathbf{x}) = \sum_{\mathbf{l}=1}^{\mathbf{NF}} \mathbf{b}_\mathbf{l} \cdot \boldsymbol{\varphi}_\mathbf{l}(\mathbf{x}), \tag{6}$$

where NF is the number of regressors in the model bank $\{\varphi_\mathbf{l}(\mathbf{x})\}$ and the coefficients $\mathbf{b}_\mathbf{l}$ are regression coefficients that should not be considered as weight factors, e.g., could be positive or negative.

Finally, the above two-step metamodel building strategy leads to solving the linear system of NF equations with NF unknowns $\mathbf{b}_\mathbf{l}$.

3. Extended MAM and Adaptive Selection of Weighting Coefficients

Moving least-squares method (MLSM) is a metamodel building technique that has been suggested for use in the meshless form of the finite element method [46] and then advocated to build the highly dependent metamodels around the specific point in the local space for design optimization [47–49]. Intrigued by MLSM, an extended MAM (EMAM) is proposed in this paper to explore the full potential of the polynomial regression-based metamodels through the entire optimization process. Since MLSM can more accurately predict the response function around the point at which the approximation is made, EMAM has the ability to capture values of the response function around the point with a high level of accuracy.

In current research, a novel metamodel called Taylor's expansion metamodel is developed to construct the linearly combined metamodel and it is given as follows:

$$\boldsymbol{\varphi}(\mathbf{x}) = \boldsymbol{\varphi}\left(\mathbf{x}^0\right) + \sum_{\mathbf{i}=1}^{\mathbf{N}} \left(\frac{\partial \boldsymbol{\varphi}(\mathbf{x})}{\partial \mathbf{x_i}} |_{\mathbf{x}=\mathbf{x}^0} \cdot \Delta_\mathbf{i} \right), \quad \Delta_\mathrm{i} = \mathrm{x_i} - \mathrm{x}_i^0, \quad (7)$$

where $\varphi(\mathbf{x}^0)$ is the initial function value at the starting point $\mathbf{x}^0$, which is the suboptimal point obtained in the previous iteration during the optimization loop, and N is the number of design variables. It is noted that the quality of the above metamodel highly depends on the suboptimal point $\mathbf{x}^0$ because the approximation around $\mathbf{x}^0$ is constructed with high levels of accuracy and efficiency by linear expansion. This enables EMAM to outperform other metamodel methods in local search for the optimal solution and also improve the quality of the optimal solution.

As described in Section 2, the explicit metamodel will be determined on the basis of implicit response evaluations by the weighted least-squares fitting. Apparently, values of the weighting coefficient $\mathbf{w_p}$ in equations (3) and (5) directly control the quality of the approximation. Generally, the optimal solution in an optimization problem lies on the boundary of the feasible region. In other words, there is at least one constraint to be activated when the optimum is found in the constrained optimization problems. Therefore, it is necessary to propose a strategy for adaptive selection of weighting coefficients in the current optimization framework so that the approximation function $\widetilde{F}(\mathbf{x})$ could improve its accuracy near the promising region. As a result, the optimal solution will be more likely to locate in the vicinity of a boundary.

To implement Taylor's expansion metamodel into this approximation-based framework, the weighting coefficient w_p is defined as follows:

$$\mathbf{w_p} = \prod_{\mathbf{j}=1}^{\mathbf{M}} \mathbf{w_p^j},$$
$$\mathbf{w_p^j} = \begin{cases} \left(\mathbf{F_j}(\mathbf{x}) + 0.1\right)^{\boldsymbol{\alpha}}, & \mathbf{if}\ 0.9 \le \mathbf{F_j}(\mathbf{x}) < 1, \\ \mathbf{F_j^{-\beta}}(\mathbf{x}), & \mathbf{if}\ \mathbf{F_j}(\mathbf{x}) > 1, \\ 1, & \mathbf{else}, \end{cases} \quad (8)$$

where α and β are user-defined positive constants and $\alpha = 4$ and $\beta = 5$ are determined by the author's experience from a lot of tests performed. It is noted that the bigger weightings should be adaptively assigned to the points which are located more closely to the boundaries between the feasible and infeasible regions. As can be seen from equation (8), the maximum constraint weighting factor $\mathbf{w_p^j}$ is assigned when the constraint evaluation equals 1. With $\beta > \alpha$, the quality of the metamodel to approximate response functions in the feasible region is much 'better' than the one in the infeasible region.

As compared to the formulations of the weighting coefficient $\mathbf{w_p}$ in [45], which is defined in the following equation:

$$\mathbf{w_p} = \mathbf{w_p^o} \cdot \mathbf{w_p^j} \quad (\mathbf{j} = 1, \ldots, \mathbf{M}),$$
$$\mathbf{w_p^j} = \begin{cases} \mathbf{F_j^{\alpha}}(\mathbf{x}) & \mathbf{F_j}(\mathbf{x}) \le 1 \\ \mathbf{F_j^{-\alpha}}(\mathbf{x}) & \mathbf{F_j}(\mathbf{x}) \ge 1 \end{cases}, \quad (9)$$
$$\mathbf{w_p^o} = \left[\frac{\mathbf{F}_0(\mathbf{x}_1)}{\mathbf{F}_0(\mathbf{x_p})}\right]^{\boldsymbol{\beta}},$$

$\alpha = 4$ and $\beta = 1.5$.

The objective weighting factor $\mathbf{w_p^o}$ has not been used in the proposed strategy. There are two reasons: (1) The weighting factor $\mathbf{w_p^o}$ will sometimes be allocated a wrong weight value for an infeasible point. Considering an infeasible solution $\mathbf{x}_p$ with an extremely low objective value $F_0(\mathbf{x}_p)$, this weight $[(F_0(x_1))/(F_0(x_p))]^\beta$ would approach infinity. As a result, the quality of the metamodel is severely damaged. (2) Even if the objective weighting factor $\mathbf{w_p^o}$ is well defined, the influence of $\mathbf{w_p^o}$ on the quality of metamodels is much less than that of the constraint weighting factor $\mathbf{w_p^j}$.

In equation (9), the constraint weighting coefficient $\mathbf{w_p}$ only considers the contribution from each constraint during the process of the constraint metamodel building. Therefore,$\mathbf{w_p}$ is only affected by a single constraint for a given design point. In the proposed strategy for adaptive selection of weighting coefficients, all information of different constraints is considered by the product of a bunch of weighting factor $\mathbf{w_p^j}$. Obviously, one design point and its corresponding constraints are not isolated one from another. The optimal behavior of a design point should be judged by the information of the whole set of constraints, rather than the information from a single constraint. By combining the constraints with multiplication shown in equation (8), the more the constraints are active, the larger the weighting of a point. On the contrary, the less weighting value is given when the point is far away from the feasible region. It is noted that the proposed strategy for adaptive selection of weighting coefficients leads to the approximation function with a high level of accuracy in the feasible region, which results in a high probability of identifying feasible solutions during the optimization process.

Based on these facts, the flowchart of the EMAM as an enhanced optimization framework is shown in Figure 1. At the beginning, an initial feasible design is given to trigger the entire optimization process and the corresponding trust region is defined. Then, a number of sampling points ($N + 5$, where N means the number of design variables) are uniformly distributed over the trust region. The objective and constraint values at these points are obtained by evaluating the response functions. In this paper, we assume the response functions are computationally expensive in simulation-based optimization and any design point will never cause a crash during the simulations. Based on the obtained data about design variables and responses, Taylor's expansion regressor defined by equation (7) and five other forms of regressors represented by equation (4) are built in sequence. Following that, these six regressors are assembled into one metamodel for the

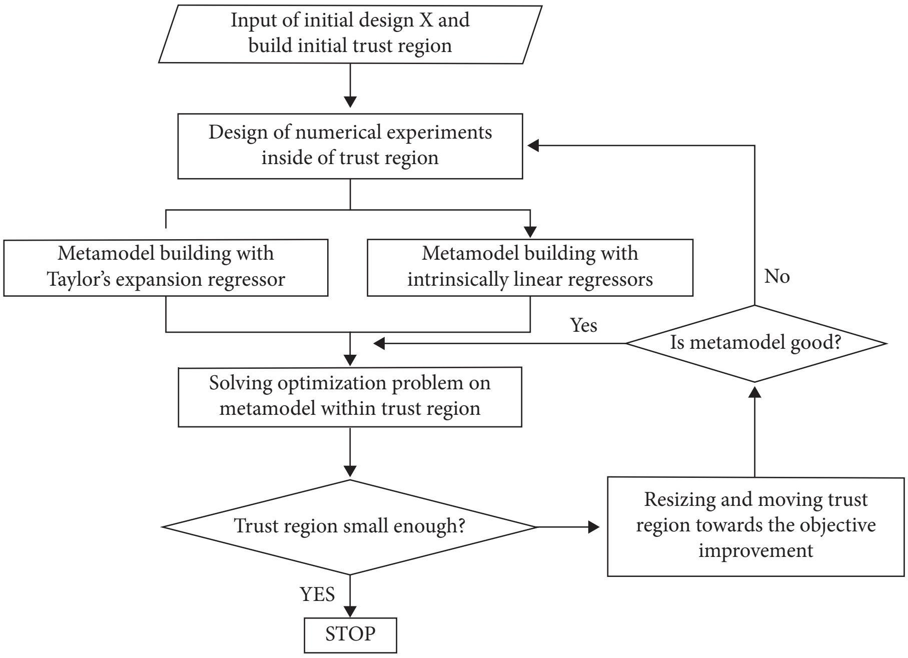

Input of initial design X and build initial trust region
Design of numerical experiments inside of trust region
Metamodel building with Taylor's expansion regressor
Metamodel building with intrinsically linear regressors
No
Yes
Is metamodel good?
Solving optimization problem on metamodel within trust region
Trust region small enough?
Resizing and moving trust region towards the objective improvement
YES
STOP

Figure 1: Extended MAM flowchart.

evaluations of all response functions of the interests. Thus, numerical simulations are performed on the metamodels inside the trust region and the optimal solution of the subproblem is found by SQP (sequential quadratic programming) solver. To update the trust region in next iteration, it will be resized and moved based on several indicators [2] and then, the next iteration starts. When the size of the trust region is small enough, the entire optimization process will terminate and the final solution will be achieved.

4. Examples

4.1. Design of a Tension/Compression Spring. This problem first described by Belegundu [34] and Arora [50] has arisen from the wide applications of vibration-resistant structures in civil engineering. The design objective is to minimize the weight of a tension/compression spring subject to constraints on shear stress, surge frequency, and minimum deflections as shown in Figure 2. The design variables include the wire diameter d, the mean coil diameter D, and the number of active coils N. Detailed information on constraint functions g_1, g_2, g_3, and g_4 can be found in reference [50].

In Table 1, the results obtained by extended MAM are compared to those by other methods, such as mathematical programming methods [34, 50], genetic algorithms (GAs) [36–38], evolution strategies (ESs) [39], and charged system search (CSS) [41]. As is shown in Table 1, the optimal design (0.0126653) found by extended MAM has a good agreement with the one by MAM and it also represents the lightest weight design among all the feasible solutions indicated in Table 2. Actually, Kaveh and Talatahari [41] obtained a slightly better design (0.0126384) using CSS. However, this optimal design can be noted that at least 0.11% design constraint violation (g_2) was clearly observed in Table 2.

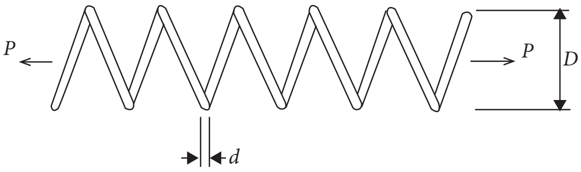


Figure 2: Schematic of the tension/compression spring.

By choosing different starting points that are randomly generated for each example in this section, both extended MAM and MAM have the ability to obtain the lightest design (0.0126653) shown in Table 3, when eight sampling points are selected to build the metamodels in each iteration of the optimization process. Taking into account the randomness in the developed algorithm, the mean value and standard deviation (SD) of the results have also been provided in Table 3 to reveal the method's robustness. To compare methods using a probabilistic metric, more details can be found in [51]. In general, the number of evaluations called by the extended MAM is less than the number of analyses by MAM and the former can obtain the more robust optimum as well. In conclusion, the extended MAM effectively enhances search performance with the higher robustness and accuracy of the optimal solution than metaheuristic algorithms.

4.2. A Reactor Pressure Vessel Example. The second case study focuses on the design optimization of a cylindrical pressure vessel capped at both ends by hemispherical heads (Figure 3). The main purpose of this research is to minimize the total manufacturing cost of the vessel including the combination of welding, material, and forming costs. The design variables consist of the shell thickness T_s, the

Table 1: Comparison of optimal designs of the spring using different algorithms.

Methods	**d**	**D**	**N**	Weight
Mathematical programming [34]	0.050000	0.315900	14.250000	0.0128334
Mathematical programming [50]	0.053396	0.399180	9.185400	0.0127303
GA-based [37]	0.051480	0.351661	11.632201	0.0127048
GA-based [38]	0.051989	0.363965	10.890522	0.0126810
ES-based [39]	0.051643	0.355360	11.397926	0.012698
CSS [41]	0.051744	0.358532	11.165704	0.0126384
MAM	0.051604352	0.35468326	11.409247	0.0126653
Extended MAM	0.051656017	0.35592318	11.3357128	0.0126653

Table 2: Constraint results of the optimal designs.

Methods	$\mathbf{g}_1$	$\mathbf{g}_2$	$\mathbf{g}_3$	$\mathbf{g}_4$
Mathematical programming [34]	−−0.000014	−0.003782	−3.938302	−0.756067
Mathematical programming [50]	0.000019	−0.000018	−4.123832	−0.698283
GA-based [37]	−0.002080	−0.000110	−4.026318	−4.026318
GA-based [38]	−0.000013	−0.000021	−4.061338	−0.722698
ES-based [39]	−0.001732	−0.0000567	−4.039301	−0.728664
CSS [41]	$8.78603e-6$	0.0011043	−4.063371	−0.726483
MAM	$-1.0843e-7$	$-6.10541e-8$	−4.0497478	−7.291416
Extended MAM	$-6.3091e-7$	$-3.2158e-7$	−4.052208	−7.282805

Table 3: Optimal designs of the spring using MAM and extended MAM algorithms with different starting points.

Starting point (**d**, **D**, **N**)			MAM		Extended MAM	
			Output value	No. of iterations	Output value	No. of iterations
0.05	0.4	9	0.01311	71	0.01269	17
0.08	1.0	10	0.01311	19	0.01289	26
0.06	0.5	11	0.0126684	14	0.0126653	15
0.09	0.7	9	0.01268	14	0.01280	22
0.06	0.6	12	0.0126653	14	0.0126653	14
Average			0.01284674	26.4	0.01274212	18.8
SD			$2.15e-4$	22.4	$8.91e-5$	4.5

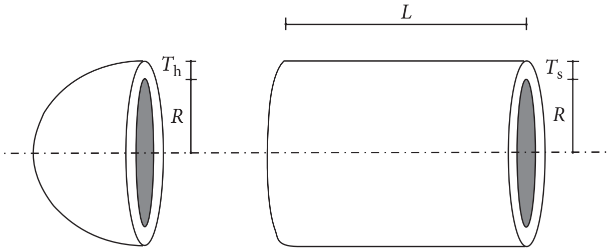


Figure 3: Pressure vessel with the indication of design variables.

spherical head thickness T_h, the radius of cylindrical shell R, and the shell length L. The detailed problem formulation is given in [35].

The comparison of results obtained by the extended MAM and other metamodel-based methods (SCGOSR [52], eDIRECT-C [53], ConstrLMSRBF [53], CORBA [53], and CiMPS [53]) has been presented in Table 4. The cost computed using the extended MAM or MAM has been further reduced to 5885.268 by 0.0009% from 5885.33, which was the best design referred in [52]. To build the metamodels at each iteration of the optimization process, nine sampling points are used in this example. It is noted that the optimized solutions by MAM and extended MAM are the best feasible designs since no violated constraints are observed in Table 5. SCGOSR [52] could find a near-optimal design with the cost of 5885.3653, which is slightly heavier than the result by extended MAM, but the first and second constraints are violated. In average, the extended MAM outperforms the MAM to seek the optimum in terms of the number of iterations used in the case studies with different starting points shown in Table 6. Taking into account the above advantages of extended MAM for seeking optimal solutions, the superiority of the proposed method over other metamodel-based techniques has been demonstrated in terms of the accuracy and efficiency. It is concluded that hybrid algorithms, such as the extended MAM and MAM, are quite robust algorithms to consistently achieve higher accuracy of the solution than other metamodel-based algorithms used for solving problems with multiple local optima, and the extended MAM has a slightly faster rate of convergence than MAM.

4.3. Welded Beam Design. Design optimization of a welded beam shown in Figure 4 is a complex and challenging problem in nature with many variables and constraints.

Table 4: Comparison of the optimal solution with the literature on pressure vessel designs.

Methods	$\mathbf{T_s}$	$\mathbf{T_h}$	$\mathbf{R}$	$\mathbf{L}$	Cost
SCGOSR [52]	0.778187	0.384658	40.320586	199.986548	5885.3653
eDIRECT-C [53]	1.00000	0.62500	51.81347	84.57855	7006.7816
ConstrLMSRBF [53]	1.00000	0.62501	51.81035	84.60683	7007.2309
CORBA [53]	1.00000	0.62503	51.80156	84.66651	7007.8352
CiMPS [53]	1.10000	0.62500	56.99482	51.00125	7163.7390
MAM	0.7781687	0.3846492	40.319619	200.000	5885.268
Extended MAM	0.7781687	0.3846492	40.319619	200.000	5885.268

Table 5: Comparison of present constraint values with the literature for the pressure vessel.

Methods	$\mathbf{g_1}$	$\mathbf{g_2}$	$\mathbf{g_3}$	$\mathbf{g_4}$
SCGOSR [52]	$2.8e-2$	$9.7e-3$	$-6.5e-2$	$-4.0e+1$
eDIRECT-C [53]	$-2.9e-8$	$-1.3e-1$	$-1.0e-1$	$-1.6e+2$
ConstrLMSRBF [53]	$-6.0e-5$	$-1.3e-1$	$-4.7e+1$	$-1.6e+2$
CORBA [53]	$-2.3e-4$	$-1.3e-1$	$-1.2e+1$	$-1.6e+2$
CiMPS [53]	$3.7e-2$	$-8.1e-2$	$-6.2e-2$	$-1.9e+2$
MAM	$-5.3e-8$	-0.0012	-0.01962	-40.000
Extended MAM	$-5.3e\ \ -8$	-0.0012	-0.01962	-40.000

Table 6: Optimal pressure vessel designs using MAM and extended MAM algorithms with different starting points.

Methods-				MAM		Extended MAM	
Starting point $(\mathbf{T_s}, \mathbf{T_h}, \mathbf{R}, \mathbf{L})$				Output value	No. of iterations	Output value	No. of iterations
1.0	1.0	100	150	5885.268	10	5885.268	10
0.8	0.5	50	150	5885.268	10	5885.268	9
0.5	0.5	100	100	5885.268	21	5885.268	17
1.5	1.5	50	50	5885.268	9	5885.268	10
Average				5885.268	12.5	5885.268	11.5
SD				5885.268	4.9	5885.268	3.2

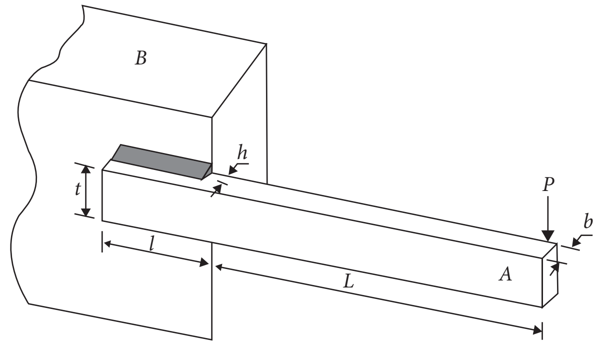


Figure 4: Design variables of a welded beam structure in parametric optimization.

Usually, conventional optimization methods fail to find global optimal solution. Hence, the welded beam design problem is often used to evaluate the performance of different optimization methods. To determine the best set of design variables for minimizing the total fabrication cost of the structure, the minimum cost optimization is performed considering shear stress (τ), bending stress (σ), buckling load (p_c), and end deflection (δ) constraints. The constants in this study are chosen as follows:

$$\begin{aligned} \mathbf{P} &= 6000\, lb, \\ \mathbf{L} &= 14\,\text{in}, \\ \mathbf{E} &= 30 \times 10^6\ \text{psi}, \\ \mathbf{G} &= 12 \times 10^6\ \text{psi}, \\ \boldsymbol{\tau}_{\mathbf{max}} &= 13600\,\text{psi}, \\ \boldsymbol{\sigma}_{\mathbf{max}} &= 30000\,\text{psi}, \\ \boldsymbol{\delta}_{\mathbf{max}} &= 0.25\,\text{in}. \end{aligned} \tag{10}$$

Taking into account design variables $x_1 = h$, $x_2 = l$, $x_3 = t$, and $x_4 = b$, the mathematical optimization of the problem can be formulated as follows.

Objective: minimize the cost

$$\mathbf{cost}(\mathbf{x}) = 1.10471\, \mathbf{x}_1^2 \mathbf{x}_2 + 0.04811\, \mathbf{x}_3 \mathbf{x}_4 (14 + \mathbf{x}_2). \tag{11}$$

The bounds on the design variables are

$$\begin{gathered}0.1 \le \mathbf{x}_1 \le 2,\\ 0.1 \le \mathbf{x}_2 \le 10,\\ 0.1 \le \mathbf{x}_3 \le 10,\\ 0.1 \le \mathbf{x}_4 \le 2.\end{gathered} \tag{12}$$

Subject to

$$\begin{aligned}&\mathbf{g}_1(\mathbf{x}) = \boldsymbol{\tau}(\mathbf{x}) - \boldsymbol{\tau}_{\mathbf{max}} \le 0,\\ &\mathbf{g}_2(\mathbf{x}) = \boldsymbol{\sigma}(\mathbf{x}) - \boldsymbol{\sigma}_{\mathbf{max}} \le 0,\\ &\mathbf{g}_3(\mathbf{x}) = \mathbf{x}_1 - \mathbf{x}_4 \le 0,\\ &\mathbf{g}_4(\mathbf{x}) = \left[0.10471\mathbf{x}_1^2 + 0.04811\mathbf{x}_3\mathbf{x}_4(14 + \mathbf{x}_2)\right] - 5 \le 0,\\ &\mathbf{g}_5(\mathbf{x}) = 0.125 - \mathbf{x}_1 \le 0,\\ &\mathbf{g}_6(\mathbf{x}) = \boldsymbol{\delta}(\mathbf{x}) - \boldsymbol{\delta}_{\mathbf{max}} \le 0,\\ &\mathbf{g}_7(\mathbf{x}) = \mathbf{p} - \mathbf{p}_{\mathbf{c}}(\mathbf{x}) \le 0,\end{aligned} \tag{13}$$

where

$$\begin{aligned}\boldsymbol{\tau}' &= \frac{\mathbf{P}}{\sqrt{2}\mathbf{x}_1\mathbf{x}_2},\\ \boldsymbol{\tau}'' &= \frac{\mathbf{MR}}{\mathbf{J}},\\ \mathbf{M} &= \mathbf{P}\left(\mathbf{L} + \frac{\mathbf{x}_2}{2}\right),\\ \mathbf{R} &= \sqrt{\frac{{\mathbf{x}_2}^2}{4} + \left(\frac{\mathbf{x}_1 + \mathbf{x}_3}{2}\right)^2},\\ \boldsymbol{\tau}(\mathbf{x}) &= \sqrt{(\boldsymbol{\tau}')^2 + 2\boldsymbol{\tau}'\boldsymbol{\tau}''^{(\mathbf{x}_2/2\mathbf{R})} + (\boldsymbol{\tau}'')^2},\\ \mathbf{J} &= 2\left\{\sqrt{2}\,\mathbf{x}_1\mathbf{x}_2\left[\frac{\mathbf{x}_2^2}{12} + \left(\frac{\mathbf{x}_1 + \mathbf{x}_3}{2}\right)^2\right]\right\},\\ \boldsymbol{\sigma}(\mathbf{x}) &= \frac{6\mathbf{PL}}{\mathbf{x}_4\mathbf{x}_3^2},\\ \boldsymbol{\delta}(\mathbf{x}) &= \frac{4\mathbf{PL}^3}{\mathbf{E}\mathbf{x}_3^3\mathbf{x}_4},\\ \mathbf{p}_{\mathbf{c}}(\mathbf{x}) &= \frac{4.013\sqrt{\mathbf{E}\left(\left(\mathbf{x}_3^2\mathbf{x}_4^6\right)/36\right)}}{\mathbf{L}^2}\left(1 - \frac{\mathbf{x}_3}{2\mathbf{L}}\sqrt{\frac{\mathbf{E}}{4\mathbf{G}}}\right).\end{aligned} \tag{14}$$

In this example, the best combination of design variables and the lowest cost by hybrid algorithms (MAM and extended MAM) are compared with those obtained using GA [36–38], PSO [40, 54], FA [43], colliding bodies optimization (CBO) [42], CMA-ES [55], and differential evolution [56] in Table 7. Nine sampling points are applied to construct the metamodels in each iteration of the optimization process. Although Kaveh and Mahdavi [42] claimed that the minimum cost design was 1.724663 indicated in Table 7, the corresponding fabrication cost of the structure was actually 1.724983, which can be easily evaluated by substituting the values of design variables for the optimal design into the objective function cost (x). The cost of this design is higher than the result (1.724852) by the extended MAM, which is one of the best feasible designs shown in Table 7. CMA-ES, IAPSO, and iDEaSm could also find the best design; however, the required number of function evaluations is 4658, 12500, and 4425, respectively. MAM and EMAM only need about 120 function evaluations (about 13 iterations) as shown in Table 8, where the statistical results of four randomly tests are given to demonstrate the robustness of the solution. It is concluded that both extended MAM and MAM demonstrate the superiority over the other methods to solve the complex optimization problem with respect to the efficiency and accuracy of the solution.

4.4. A Ten-Bar Truss Structure. To further demonstrate the computational efficiency of the extended MAM, the well-known ten-bar truss benchmark problem [6] shown in Figure 5 is used to explore the potential. The optimization formulation of this problem is defined to minimize the weight of the structure by varying the cross-sectional areas (from 0.1 in^2 to 12.7 in^2) of the truss members subject to stress constraints. The allowable stress in each truss member is the same in tension and compression and is set to 25 ksi for all members except member 9 for which it is 75 ksi. The density of the truss material is 0.1 (lb/in^3), the member size $L = 360$ in, the loads $P1 = P2 = 100$ Kips, and $P3 = 0$.

In order to demonstrate the superiority of the extended MAM over other optimization methods such as PSO [57], FA [58], and SQP in HyperStudy [59], a comparison of optimal designs of ten-bar truss structure has been given in Table 9. It should be noted that for PSO, FA, MAM, and the extended MAM, the objective function value, the number of iterations, and the number of response analyses are actually the average results over 5 independent runs. The best design (1497.0) was achieved by Haftka [60]; however, some constraints indicated in Table 10 had been violated. The same conclusion can be drawn for the optimal design (1497.6) by SQP in HyperStudy. The results by PSO (1519.2) and FA (1558.1) are feasible solutions; however, they are not the best design. For the best feasible design (1497.6) by extended MAM and MAM, the higher efficiency and accuracy of these two algorithms have been demonstrated, for example, the average number of iterations used by the extended MAM has been reduced by an order of magnitude from 520 (PSO) or 400 (FA) to 28. It is also noted that the average number of response analyses (420) called by extended MAM is 24% less than the one (555) by MAM. In summary, the extended MAM outperforms the other methods in seeking the optimal solution of the complex engineering design problems in terms of the efficiency and accuracy.

Table 7: Comparison of present optimized designs with the literature for the welded beam.

Methods	$\mathbf{x}_1$ ($\mathbf{h}$)	$\mathbf{x}_2$ ($\mathbf{l}$)	$\mathbf{x}_3$ ($\mathbf{t}$)	$\mathbf{x}_4$ ($\mathbf{b}$)	Cost
GA-based [36]	0.248900	6.173000	8.178900	0.253300	2.433116
GA-based [37]	0.208800	3.420500	8.997500	0.210000	1.748309
GA-based [38]	0.205986	3.471328	9.020224	0.20648	1.728226
CPSO [40]	0.202369	3.544214	9.04821	0.205723	1.728024
ES-based [39]	0.199742	3.612060	9.037500	0.206082	1.737300
CSS [41]	0.205820	3.468109	9.038024	0.205723	1.724866
CBO [42]	0.205722	3.47041	9.037276	0.205735	1.724663
FA [43]	0.201500	3.56200	9.041400	0.205700	1.731210
CMA-ES [55]	NA	NA	NA	NA	1.724852
IAPSO [54]	0.2057296	3.47048866	9.03662391	0.20572964	1.724852
IDEaSm [56]	0.20572963	3.4704888	9.0366238	0.20572965	1.724852
MAM	0.2057296	3.4704893	9.0366242	0.2057297	1.724852
Extended MAM	0.2057296	3.4704894	9.0366242	0.2057297	1.724852

Table 8: Optimal designs of the welded beam using MAM and extended MAM algorithms with different starting points.

Methods				MAM		Extended MAM	
Starting point ($\mathbf{h}, \mathbf{l}, \mathbf{t}, \mathbf{b}$)				Output value	Iteration number	Output value	Iteration number
0.6	1.0	5.0	0.6	1.724852	14	1.724853	13
0.5	3.5	9.0	0.5	1.724852	12	1.724853	14
0.6	2.0	7.0	0.6	1.724852	14	1.724852	12
1.0	3.0	7.0	0.5	1.724853	13	1.724853	13
Average				1.724852	13.3	1.724853	13.0
SD				$4.3e-7$	0.8	$4.3e-7$	0.7

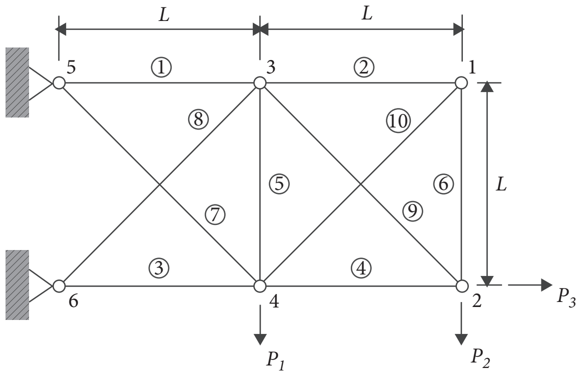


Figure 5: Ten-bar truss structure.

Table 9: Comparison of present optimized designs for ten-bar truss structure.

Design variables	Haftka [60]	HyperStudy [59]	PSO [57]	FA [58]	MAM	Extended MAM
$\mathbf{x}_1$	7.9	7.9	7.5395	7.4269	7.9	7.9
$\mathbf{x}_2$	0.1	0.1	0.4605	0.8070	0.1	0.1
$\mathbf{x}_3$	8.1	8.1	8.4605	8.6498	8.1	8.1
$\mathbf{x}_4$	3.9	3.9	3.5395	3.6580	3.9	3.9
$\mathbf{x}_5$	0.1	0.1	0.1	0.1424	0.1	0.1
$\mathbf{x}_6$	0.1	0.1	0.4605	0.6316	0.1	0.1
$\mathbf{x}_7$	5.8	5.8	6.3081	6.5491	5.798276	5.798275
$\mathbf{x}_8$	5.51	5.52	5.0056	4.7649	5.514327	5.515434
$\mathbf{x}_9$	3.68	3.68	3.3370	3.3244	3.676959	3.676927
$\mathbf{x}_{10}$	0.14	0.14	0.6513	0.8937	0.141421	0.141430
Weight (lb)	1497.0	1497.6	1519.2	1558.1	1497.6	1497.6
No. of iterations	N/A	13	520	400	37	28
No. of response analyses	N/A	144	5200	20000	555	420

Table 10: Constraints results of the continuous optimization using different techniques.

Constraints (ksi)	Haftka [60]	HyperStudy [59]	PSO [57]	FA [58]	MAM	Extended MAM
$\mathbf{g}_1$	$6.4e-4$	$6.4e-4$	$-2.1e-4$	-0.153	$-2.073e-5$	$-2.706e-6$
$\mathbf{g}_2$	-0.12	-0.12	$-1.3e{-}4$	$-4.4e-2$	$-2.255e-5$	$-2.231e-5$
$\mathbf{g}_3$	$-6.2e-4$	$-6.2e-4$	$-2.8e-10$	$-9.86e-3$	$-4.629e-7$	$-4.470e-7$
$\mathbf{g}_4$	$3.1e-3$	$3.1e-3$	$-1.1e-3$	-2.2	$-1.281e-6$	$-1.350e-6$
$\mathbf{g}_5$	-24.93	-24.93	-25.00	-25.00	-25.00	-25.00
$\mathbf{g}_6$	-0.12	-0.12	-0.31	$-9.1e-2$	$-2.255e-5$	$-2.231e-5$
$\mathbf{g}_7$	$-8.6e-3$	$-8.6e-3$	$-2.8e-10$	$-3.2e-2$	$-2.1e-6$	$-2.1e-6$
$\mathbf{g}_8$	$2.6e-2$	$2.6e-2$	$-1.3e-9$	$-9.7e-4$	$-4.77e-7$	$-4.088e-6$
$\mathbf{g}_9$	-37.53	-37.53	-37.66	-36.44	-37.50	-37.50
$\mathbf{g}_{10}$	0.1315	0.1315	$-4.0e-9$	-0.024	$-1.261e-5$	$-1.237e-5$

5. Conclusions

The paper focuses on obtaining the high efficiency and accuracy solution of complex simulation-based optimization problems by developing an extended multipoint approximation method. A novel metamodel inspired by Taylor's expansion technique is proposed, as well as a strategy for adaptive selection of weighting coefficients, so that the approximation of responses of the interests in the computationally expensive design problems can be performed more efficiently. The superiority of the extended MAM over SQP, metaheuristic algorithms, metamodel-based algorithms, and MAM has been demonstrated by four nonconvex benchmark examples in terms of the computational efficiency and accuracy. In the current implementation, there are some limitations of EMAM. First, the optimization performance needs improvement to solve mixed-variable optimization problems. Second, the moving trust region strategy has certain drawbacks of balancing exploration and exploitation. Finally, the metamodel has difficulty in modelling highly multimodal and high-dimensional responses. However, possessing the potential of remarkably reducing the computational effort in the simulation-based optimization, the extended MAM can pose great influence on solving highly nonlinear engineering problems and provide valuable insights into the development of effective algorithms applied during the simulation-driven design process.

Data Availability

The underlying data used in this paper can be made available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] R. T. Haftka, J. A. Nachlas, L. T. Watson, T. Rizzo, and R. Desai, "Two-point constraint approximation in structural optimization," *Computer Methods in Applied Mechanics and Engineering*, vol. 60, no. 3, pp. 289–301, 1987.

[2] A. Polynkin and V. V. Toropov, "Mid-range metamodel assembly building based on linear regression for large scale optimization problems," *Structural and Multidisciplinary Optimization*, vol. 45, no. 4, pp. 515–527, 2012.

[3] G. M. Fadel, M. F. Riley, and J. M. Barthelemy, "Two point exponential approximation method for structural optimization," *Structural Optimization*, vol. 2, no. 2, pp. 117–124, 1990.

[4] L. Wang and R. V. Grandhi, "Improved two-point function approximations for design optimization," *AIAA Journal*, vol. 33, no. 9, pp. 1720–1727, 1995.

[5] F. V. Keulen and V. V. Toropov, "New developments in structural optimization using adaptive mesh refinement and multipoint Approximations," *Engineering Optimization*, vol. 29, no. 1-4, pp. 217–234, 1997.

[6] D. Liu and V. V. Toropov, "Implementation of discrete capability into the enhanced multipoint Approximation method for solving mixed integer-continuous optimization problems," *International Journal of Computational Methods in Engineering Science*, vol. 17, 2016.

[7] L. Wang, Y. Liu, and Y. Liu, "An inverse method for distributed dynamic load identification of structures with interval uncertainties," *Advances in Engineering Software*, vol. 131, pp. 77–89, 2019.

[8] C. Xiong, L. Wang, G. Liu, and Q. Shi, "An iterative dimension-by-dimension method for structural interval response prediction with multidimensional uncertain variables," *Aerospace Science and Technology*, vol. 86, pp. 572–581, 2019.

[9] J. Zhang, S. Chowdhury, J. Zhang, A. Messac, and L. Castillo, "Adaptive hybrid surrogate modeling for complex systems," *AIAA Journal*, vol. 51, no. 3, pp. 643–656, 2013.

[10] Z. Michalewicz, "Genetic algorithms + data structures = evolution programs," *Computational Statistics & Data Analysis*, vol. 24, pp. 372-373, 1996.

[11] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Boston, MA, USA, 1989.

[12] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the ICNN'95 - International Conferance on Neural Networks*, pp. 1942–1948, IEEE, Perth, Australia, 1995.

[13] Z. Chen, H. Cao, K. Ye, H. Zhu, and S. Li, "Improved particle swarm optimization-based form-finding method for suspension bridge installation analysis," *Journal of Computing in Civil Engineering*, vol. 29, Article ID 04014047, 2015.

[14] P. Ghamisi and J. A. Benediktsson, "Feature selection based on hybridization of genetic algorithm and particle swarm optimization," *IEEE Geoscience and Remote Sensing Letters*, vol. 12, no. 2, pp. 309–313, 2015.

[15] L. Luo, W. He, and X. Zhang, "PSO-based approach for buckling analysis of shell structures with geometric imperfections," *Mathematical Problems in Engineering*, vol. 2019, Article ID 4073919, 8 pages, 2019.

[16] X. S. Yang, *Nature-Inspired Optimization Algorithms*, Luniver Press, Beckington, UK, 2008.

[17] A. Baghlani, M. H. Makiabadi, and M. Sarcheshmehpour, "Discrete optimum design of truss structures by an improved firefly algorithm," *Advances in Structural Engineering*, vol. 17, no. 10, pp. 1517–1530, 2014.

[18] R. L. Pedro, J. Demarche, L. F. F. Miguel, and R. H. Lopez, "An efficient approach for the optimization of simply supported steel-concrete composite I-girder bridges," *Advances in Engineering Software*, vol. 112, pp. 31–45, 2017.

[19] R. Moghadas, K. Choong and . S. Mohd, Prediction of optimal design and deflection of space structures using neural networks," *Mathematical Problems in Engineering*, vol. 2012, Article ID 712974, 18 pages, 2012.

[20] Z. Cao, X. Hei, L. Wang, Y. Shi, and X. Rong, "An improved brain storm optimization with differential evolution strategy for applications of ANNs," *Mathematical Problems in Engineering*, vol. 2015, Article ID 923698, 18 pages, 2015.

[21] Q. Li, Y. Da, Y. Zhang, B. Wang, D. Liu, and Z. Qian, "A novel combination of theoretical analysis and data-driven method for reconstruction of structural defects," https://arxiv.org/abs/2009.06276.

[22] R. H. Myers, D. C. Montgomery, G. G. Vining, C. M. Borror, and S. M. Kowalski, "Response surface methodology: a retrospective and literature survey," *Journal of Quality Technology*, vol. 36, no. 1, pp. 53–77, 2004.

[23] N. Dyn, D. Levin, and S. Rippa, "Numerical procedures for surface fitting of scattered data by radial functions," *SIAM Journal on Scientific and Statistical Computing*, vol. 7, no. 2, pp. 639–659, 1986.

[24] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn, "Design and analysis of computer experiments," *Statistical Science*, vol. 4, no. 4, pp. 409–423, 1989.

[25] J. H. Friedman, "Multivariate adaptive regression splines," *The Annals of Statistics*, vol. 19, no. 1, pp. 1–67, 1991.

[26] D. E. Rumelhart, B. Widrow, and M. A. Lehr, "The basic ideas in neural networks," *Communications of the ACM*, vol. 37, no. 3, pp. 87–92, 1994.

[27] S. M. Clarke, J. H. Griebsch, and T. W. Simpson, "Analysis of support vector regression for approximation of complex engineering analyses," *Journal of Mechanical Design*, vol. 127, no. 6, pp. 1077–1087, 2005.

[28] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *Journal of Global Optimization*, vol. 13, no. 4, pp. 455–492, 1998.

[29] R. G. Regis, "Constrained optimization by radial basis function interpolation for high-dimensional expensive black-box problems with infeasible initial points," *Engineering Optimization*, vol. 46, no. 2, pp. 218–243, 2014.

[30] Q. Zhou, Y. Wang, S.-K. Choi et al., "A robust optimization approach based on multi-fidelity metamodel," *Structural and Multidisciplinary Optimization*, vol. 57, no. 2, pp. 775–797, 2018.

[31] Q. Zhou, J. Wu, T. Xue, and P. Jin, "A two-stage adaptive multi-fidelity surrogate model-assisted multi-objective genetic algorithm for computationally expensive problems," *Engineering with Computers*, vol. 37, pp. 623–639, 2019.

[32] J. Qian, J. Yi, Y. Cheng, J. Liu, and Q. Zhou, "A sequential constraints updating approach for Kriging surrogate model-assisted engineering optimization design problem," *Engineering with Computers*, vol. 36, no. 3, pp. 993–1009, 2020.

[33] R. T. Haftka, D. Villanueva, and A. Chaudhuri, "Parallel surrogate-assisted global optimization with expensive functions - a survey," *Structural and Multidisciplinary Optimization*, vol. 54, no. 1, pp. 3–13, 2016.

[34] A. D. Belegundu, *A Study of Mathematical Programming Methods for Structural Optimization*, University of Iowa, Iowa, Iowa, 1982.

[35] B. K. Kannan and S. N. Kramer, "An augmented Lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design," *Journal of Mechanical Design*, vol. 116, no. 2, pp. 405–411, 1994.

[36] K. Deb, "Optimal design of a welded beam via genetic algorithms," *AIAA Journal*, vol. 29, no. 11, pp. 2013–2015, 1991.

[37] C. A. Coello Coello, "Use of a self-adaptive penalty approach for engineering optimization problems," *Computers in Industry*, vol. 41, no. 2, pp. 113–127, 2000.

[38] C. A. Coello Coello and E. Mezura Montes, "Constrained optimization based on a multiobjective evolutionary algorithm," *Congress on Evolutionary Computation*, vol. 3, 7 pages, 2003.

[39] E. Mezura-Montes and C. A. C. Coello, "An empirical study about the usefulness of evolution strategies to solve constrained optimization problems," *International Journal of General Systems*, vol. 37, no. 4, pp. 443–473, 2008.

[40] Q. He and L. Wang, "An effective co-evolutionary particle swarm optimization for constrained engineering design problems," *Engineering Applications of Artificial Intelligence*, vol. 20, no. 1, pp. 89–99, 2007.

[41] A. Kaveh and S. Talatahari, "A novel heuristic optimization method: charged system search," *Acta Mechanica*, vol. 213, no. 3-4, pp. 267–289, 2010.

[42] A. Kaveh and V. R. Mahdavi, "Colliding bodies optimization: a novel meta-heuristic method," *Computers & Structures*, vol. 139, pp. 18–27, 2014.

[43] A. H. Gandomi, X.-S. Yang, and A. H. Alavi, "Mixed variable structural optimization using Firefly Algorithm," *Computers & Structures*, vol. 89, no. 23-24, pp. 2325–2336, 2011.

[44] C. Liu, D. Liu, X. Mao, and X. Zhou, "Extended multipoint Approximation method," in *Proceedings of the 2nd 4th International Conference On Applied Mathematics, Simulation, Modelling (AMSM 2017)*, pp. 219–225, DEStech Publications, Inc., Phuket, Thailand, 2017.

[45] V. V. Toropov, A. A. Filatov, and A. A. Polynkin, "Multiparameter structural optimization using FEM and multipoint explicit approximations," *Structural Optimization*, vol. 6, no. 1, pp. 7–14, 1993.

[46] T. Liszka, "An interpolation method for an irregular net of nodes," *International Journal for Numerical Methods in Engineering*, vol. 20, no. 9, pp. 1599–1612, 1984.

[47] K. K. Choi, B. D. Youn, and R. Yang, "Moving least square method for reliability-based design optimization, fourth world congr," in *Proceedings of the Fourth World Congress of Structural and Multidisciplinary Optimization*, Dalian, China, June, 2001.

[48] P. Breitkopf, H. Naceur, A. Rassineux, and P. Villon, "Moving least squares response surface approximation: formulation and metal forming applications," *Computers & Structures*, vol. 83, no. 17-18, pp. 1411–1428, 2005.

[49] H. Naceur, S. Ben-Elechi, J. L. Batoz, and C. Knopf-Lenoir, "Response surface methodology for the rapid design of aluminum sheet metal forming parameters," *Materials & Design*, vol. 29, no. 4, pp. 781–790, 2008.

[50] J. S. Arora, *Introduction to Optimum Design*, McGraw-Hill, New York, NY, USA, 1989.

[51] W. J. S. Gomes, A. T. Beck, R. H. Lopez, and L. F. F. Miguel, "A probabilistic metric for comparing metaheuristic optimization algorithms," *Structural Safety*, vol. 70, pp. 59–70, 2018.

[52] H. Dong, B. Song, Z. Dong, and P. Wang, "SCGOSR: surrogate-based constrained global optimization using space reduction," *Applied Soft Computing*, vol. 65, pp. 462–477, 2018.

[53] H. Liu, S. Xu, X. Chen, X. Wang, and Q. Ma, "Constrained global optimization via a DIRECT-type constraint-handling technique and an adaptive metamodeling strategy," *Structural and Multidisciplinary Optimization*, vol. 55, no. 1, pp. 155–177, 2016.

[54] N. Ben Guedria, "Improved accelerated PSO algorithm for mechanical engineering optimization problems," *Applied Soft Computing*, vol. 40, pp. 455–467, 2016.

[55] V. V. De Melo and G. Iacca, "A modified Covariance Matrix Adaptation Evolution Strategy with adaptive penalty function and restart for constrained optimization," *Expert Systems with Applications*, vol. 41, no. 16, pp. 7077–7094, 2014.

[56] N. H. Awad, M. Z. Ali, R. Mallipeddi, and P. N. Suganthan, "An improved differential evolution algorithm using efficient adapted surrogate model for numerical optimization," *Information Sciences*, vol. 451-452, pp. 326–347, 2018.

[57] S. Chen, "Constrained particle swarm optimization," *Matlab Center*, https://www.mathworks.com/matlabcentral/fileexchange/25986, 2018.

[58] X. Yang, "Multiobjective Firefly Algorithm for Continuous Optimization," *Engineering with Computers*, vol. 29, pp. 175–184, 2012.

[59] I. N. C. Altair Engineering, *HyperStudy Version 12.0*, Altair Engineering Inc., Troy, MI, USA, 2012.

[60] R. T. Haftka and Z. Gürdal, "Elements of structural optimization," *Elem. Struct. Optim*, vol. 11, p. 481, 1992.

Hindawi
Mathematical Problems in Engineering
Volume 2021, Article ID 7804540, 9 pages
https://doi.org/10.1155/2021/7804540

Research Article

Multiobjective Genetic Algorithm and Convolutional Neural Network Based COVID-19 Identification in Chest X-Ray Images

Prashant Kumar Shukla,[1] Jasminder Kaur Sandhu ,[2] Anamika Ahirwar ,[3] Deepika Ghai,[4] Priti Maheshwary,[5] and Piyush Kumar Shukla [6]

[1]Department of Computer Science and Engineering, University Institute of Technology, Rajiv Gandhi Proudyogiki Vishwavidyalaya (Technological University of Madhya Pradesh), Bhopal (MP), India
[2]Chitkara University Institute of Engineering and Technology, Chitkara University, Rajpura, Punjab, India
[3]Department of Science and Technology, Jayoti Vidyapeeth Women's University, Jaipur, Rajasthan, India
[4]Lovely Professional University, Jalandhar, India
[5]Department of Computer Science and Engineering, Rabindranath Tagore University, Bhopal, India
[6]Department of Computer Science and Engineering, University Institute of Technology, Rajiv Gandhi Proudyogiki Vishwavidyalaya (Technological University of Madhya Pradesh), Bhopal (MP), India

Correspondence should be addressed to Piyush Kumar Shukla; pphdwss@gmail.com

Received 28 May 2020; Accepted 1 October 2020; Published 25 February 2021

Academic Editor: Manjit Kaur

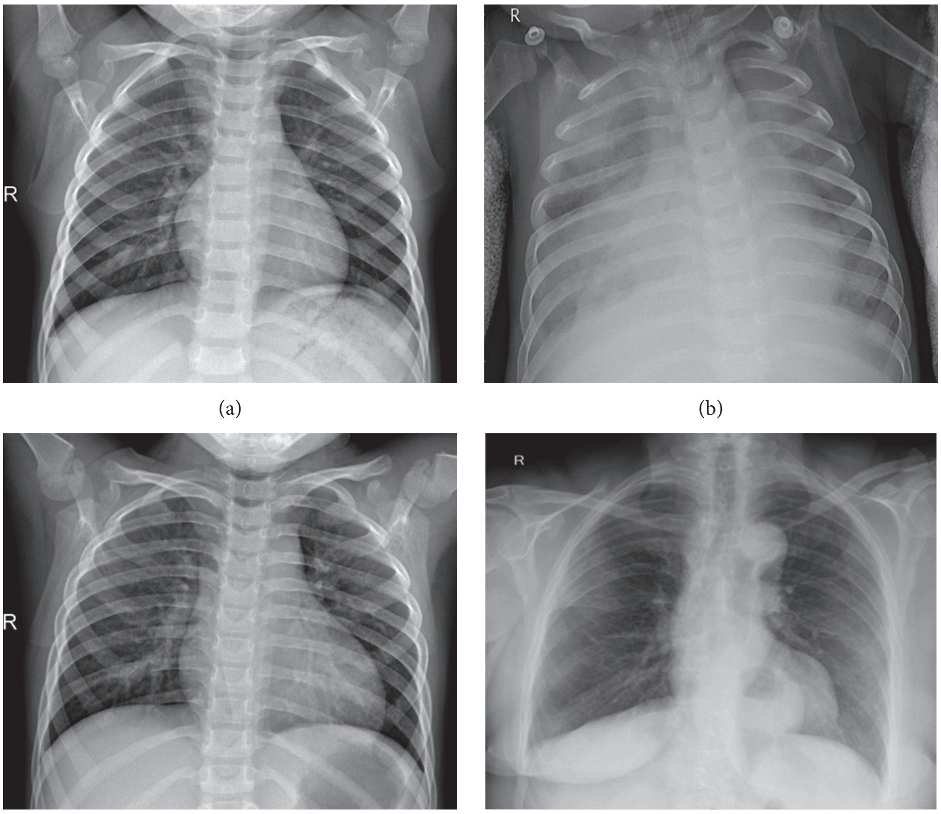
COVID-19 is a new disease, caused by the novel coronavirus SARS-CoV-2, that was firstly delineated in humans in 2019. Coronaviruses cause a range of illness in patients varying from common cold to advanced respiratory syndromes such as Severe Acute Respiratory Syndrome (SARS-CoV) and Middle East Respiratory Syndrome (MERS-CoV). The SARS-CoV-2 outbreak has resulted in a global pandemic, and its transmission is increasing at a rapid rate. Diagnostic testing and approaches provide a valuable tool for doctors and support them with the screening process. Automatic COVID-19 identification in chest X-ray images can be useful to test for COVID-19 infection at a good speed. Therefore, in this paper, a framework is designed by using Convolutional Neural Networks (CNN) to diagnose COVID-19 patients using chest X-ray images. A pretrained GoogLeNet is utilized for implementing the transfer learning (i.e., by replacing some sets of final network CNN layers). 20-fold cross-validation is considered to overcome the overfitting quandary. Finally, the multiobjective genetic algorithm is considered to tune the hyperparameters of the proposed COVID-19 identification in chest X-ray images. Extensive experiments show that the proposed COVID-19 identification model obtains remarkably better results and may be utilized for real-time testing of patients.

1. Introduction

The initial occurrence of COVID-19 disease was found in Wuhan, China, during December 2019. Ever since, it is increasing at a rapid rate in the entire world. The testing of COVID-19 is time-consuming, and also, the results obtained from rapid COVID-19 testing kits are not reliable. Therefore, radiologists and doctors have started using supervised learning techniques to test COVID-19 disease. The prime objective is to identify COVID-19 patients as infected or not, at a rapid rate [1].

The deep learning techniques may be utilized for COVID-19 patient identification [2]. Figure 1 shows the different chest X-ray images. It is found that there exists a significant change in the chest X-ray image of COVID-19-infected patients as compared to other images.

Machine learning and deep learning techniques are extensively employed to implement computer-aided identification [1, 3]. It has been observed that these techniques can save significant time of clinical persons and doctors for the examination of medical images such as X-ray and Computed Tomography scan (CT scan) [3, 4]. However, these learning techniques require a significant amount of medical images for training. Also, efficient feature extraction and selection techniques are desirable to achieve significant results [5, 6]. Recently, metaheuristic techniques are also



(a)

(b)

(c)

(d)

Figure 1: Chest X-ray images: (a) healthy, (b) bacterial pneumonia, (c) viral pneumonia (not COVID-19), and (d) COVID-19 infected.

used to tune the hyperparameters of these machine learning models [2, 7].

In this paper, a COVID-19 identification model from chest X-ray images is proposed. The main contributions of this work are as follows:

(1) The Convolutional Neural Network (CNN) is used to predict COVID-19 disease by using their respective chest X-ray images

(2) A pretrained GoogLeNet is used for implementing the transfer learning (i.e., by replacing some sets of final network CNN layers)

(3) 20-fold validation is considered to overcome the overfitting issue

(4) Finally, the multiobjective genetic algorithm is considered for tuning the hyperparameters of the proposed COVID-19 identification model

(5) Extensive experiments show that the proposed COVID-19 identification model achieves remarkably good results and may be utilized for real-time testing of patients

The rest of this paper is classified into the following sections. Section 2 presents the related work. The proposed COVID-19 identification model is illustrated in Section 3. Performance analysis is manifested in Section 4. Conclusions are outlined in Section 5.

2. Related Work

This section highlights various techniques that are used to diagnose COVID-19-infected patients from chest X-ray images.

Tang et al. [8] employed GoogLeNet to extract the characteristics of the images. Multistage feature fusion is contemplated to recognize the scene from output characteristics. Gao et al. [9] classified breast cancer by utilizing shallow deep CNN. Deepak and Ameer [10] presented an identification technique using GoogLeNet and deep transfer learning for brain MRI images. Cinar and Yildirim [11] proposed a technique to diagnose the brain tumor using ResNet-50. In this model, the last five layers are removed and eight new layers are appended. Nayak et al. [12] implemented an identification technique through CNN with five layers. This technique comprised four convolutional layers and one fully connected layer.

Liu et al. [13] implemented a ResNet model with multiscale spatiotemporal characteristics. Hao et al. [14] proposed optimized CNN based on target region selection for image recognition. Taheri and Toygar [15] proposed directed acyclic graph-based CNN for identification. It is based on the combination of VGG-16 and GoogLeNet. Ciocca et al. [16] applied CNN to diagnose the images and considered a residual network with 50 layers to extract the characteristics. Liu et al. [17] proposed an identification technique using optimization of ResNet-50 for remote sensing images. Han and Shi [18] presented a multilead residual neural network to extract the characteristics of ECG records. Talo et al. [19] considered the pretrained models VGG-16, AlexNet, ResNet-18, ResNet-34, and ResNet-50 to automatically diagnose MRI images. They found that ResNet-50 has better accuracy as compared to the other pretrained models.

Das et al. [20] designed a novel extreme version of Inception (Xception) based COVID-19 identification model. Liu et al. [21] suggested an identification model based on ResNet and transfer learning model. In this, a new data augmentation technique is considered with the help of a filter for small datasets. Togacar et al. [22] considered a deep learning model to detect COVID-19 using the X-ray images. The fuzzy color technique is considered to restructure the data classes. MobileNetV2 and SqueezeNet are applied to build the dataset. Social mimic optimization is considered to obtain the feature sets. Further, Support Vector Machine (SVM) is used to diagnose efficient characteristics. Pannu et al. [7, 23] implemented swarm intelligence-based Adaptive Neuro-Fuzzy Inference System (ANFIS) to diagnose COVID-19-infected people.

It has been observed that supervised learning algorithms may be used to test COVID-19 disease from chest X-ray images. Also, the use of pretrained feature extraction models can improve the identification rate [24–27]. The hyperparameter tuning of these models can achieve significant results. The k-fold validation [25] is used to overcome the overfitting problem.

3. Proposed Deep COVID-19 Classification Model

This work used CNN and GoogLeNet for the identification of COVID-19 disease. In addition, a multiobjective genetic algorithm is considered to tune the hyperparameters of the proposed COVID-19 identification model. The step-by-step flow of the designed COVID-19 identification model is discussed in Algorithm1.

3.1. Transfer Learning Using a Pretrained GoogLeNet. In this work, a GoogLeNet is considered to extract significant characteristics of chest X-ray images. It is a pretrained model, and is used as a transferred source. The characteristics extracted from this layer are considered as transfer learning to build the CNN-based COVID-19 identification model.

3.2. Convolutional Neural Network. CNN is widely used for identification problems [28]. Figure 2 shows the standard architecture of the CNN model. The subsequent sections discuss various layers of CNN.

3.2.1. Convolutional Layer. This layer is considered to build the input characteristics. Various convolution filters are considered to compute the patterns (Figure 3). Each neuron of the convolutional layer is connected with its sibling neurons to process the feature maps [29, 30].

Every time a convolutional operator provides a new feature map, the feature value y_{ij} in the k^{th} feature map is evaluated as follows:

$$y_{ij,k} = \sum\left(\mathbf{w}_k \odot \mathbf{x}_{ij}\right) + b_k, \tag{1}$$

where $\mathbf{w}_k$ and b_k represent the average and bias values of k^{th} mask, $\mathbf{x}_{ij}$ represents the input mask centered at (i, j), and $\odot$ is the Hadamard product of two matrices.

Weights are shared between sibling nodes to minimize the complexity of the model.

3.2.2. Nonlinear Layer. The nonlinear layer uses an activation function and is implemented on the entire set of feature maps. It can deal with the nonlinear dependencies of the feature maps. In this paper, the ReLu activation function is considered.

3.2.3. Pooling Layer. This layer does not come up with any kind of weights. It endeavors to gain shift invariance by minimizing the feature maps and considering activation properties from the local range of CNN. Average and maximum operators are generally considered in the pooling layer. It uses $k \times k$ mask and produces a unique value. In case of a $N \times N$ layer, the output will be a $N/k \times N/k$ layer.

3.2.4. Fully Connected Layer. This layer considers high-level reasoning. There are connections in every input-output pair. After this layer, other nonlinear functions are used.

3.2.5. Loss Layer. Finally, a loss layer is considered to obtain the trained COVID-19 identification model. For COVID-19 identification, a softmax operator is utilized. Assume that θ defines attributes of CNN such as bias and kernel operators. When obtaining N required sets, $\mathbf{y}^{(i)}$ is the target class considering i^{th} input and $\mathbf{o}^{(i)}$ defines the output of CNN; then, the loss of CNN is computed as follows:

$$\mathcal{L} = \frac{1}{N}\sum_{n=1}^{N} \ell\left(\boldsymbol{\theta}; \mathbf{y}^{(n)}, \mathbf{o}^{(n)}\right). \tag{2}$$

3.3. Multiobjective Fitness Function. The proposed COVID-19 identification model suffers from the hyperparameter tuning problem; therefore, in this paper, a multiobjective genetic algorithm is considered. The performance metrics accuracy (A_c) and F-measure (F_m) are considered to design a multiobjective fitness $(f(t))$ function as

$$f(t) = \begin{cases} \text{maximize}\left(A_c\right), \\ \text{maximize}\left(F_m\right), \end{cases} \tag{3}$$

where A_c can be evaluated as follows:

$$A_c = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}, \tag{4}$$

```
(1) Input: chest X-ray images as a labeled dataset
(2) Initially, GoogLeNet is utilized to evaluate the significant characteristics of COVID-19 dataset images
(3) Further, transfer learning is considered to build a CNN-based COVID-19 identification model
(4) Multiobjective genetic algorithm is utilized to tune the designed model
(5) Implement k-fold validation to overcome overfitting
(6) Return tThe constructed COVID-19 identification model for chest X-ray images
```

Algorithm 1: Proposed CNN-based COVID-19 identification model.

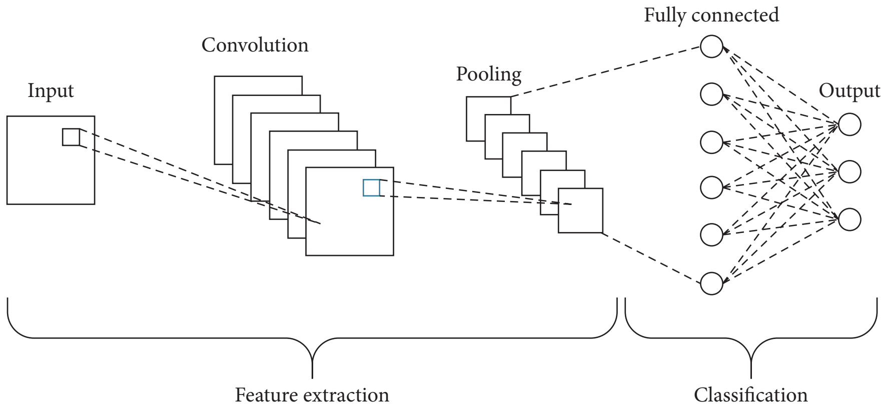


Figure 2: Architecture of convolutional neural networks.

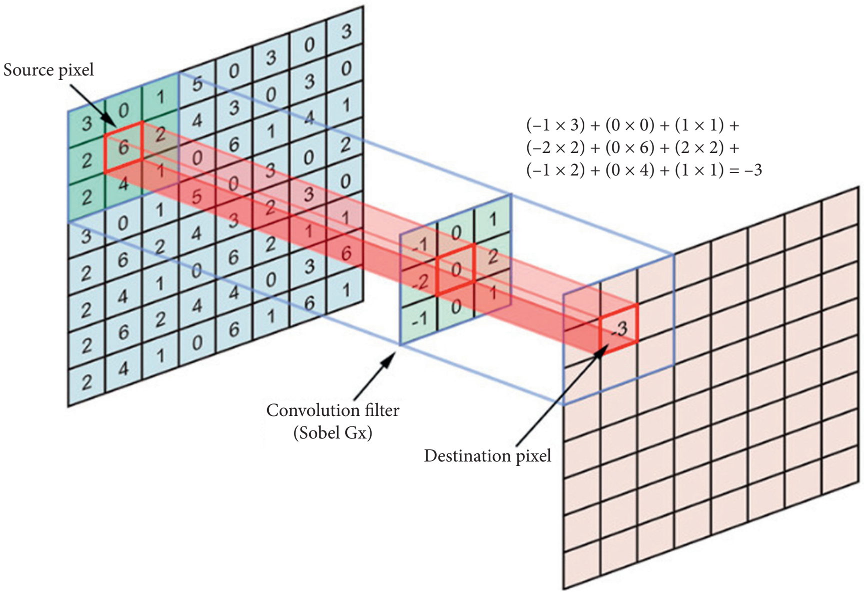


Figure 3: Convolutional layer process.

where TP, FP, TN, and FN are the true-positive, false-positive, true-negative, and false-negative values, respectively.

F_m can be evaluated as follows:

$$F_1 = 2 \cdot \frac{p_r \times r_c}{p_r + r_c}, \tag{5}$$

where p_r and r_c represent precision and recall values, respectively. p_r and r_c can be evaluated as follows:

$$p_r = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}, \tag{6}$$

$$r_c = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}. \tag{7}$$

3.4. Multiobjective Genetic Algorithm.

The genetic algorithm for Pareto optimization is discussed in Algorithms 2 and 3.

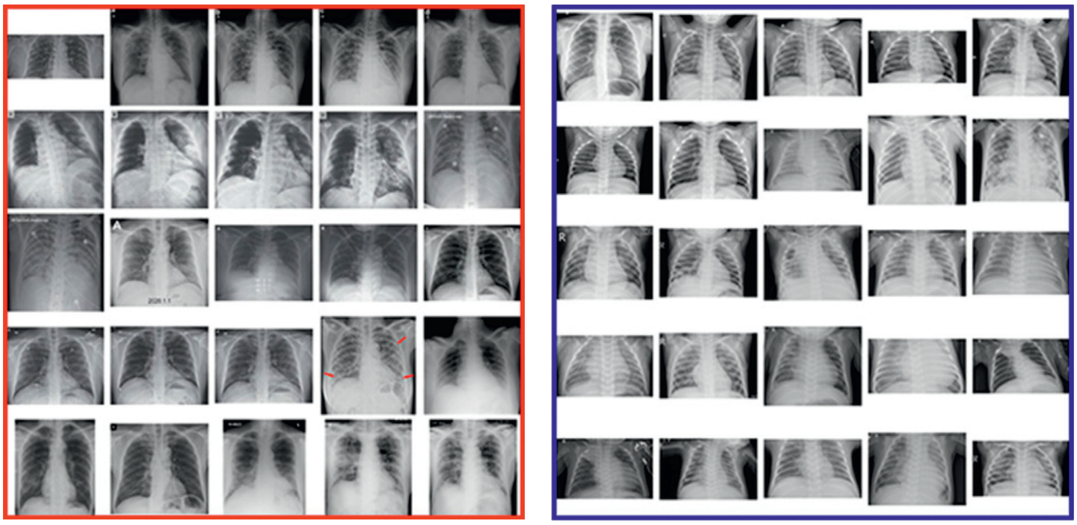
```
output: $P_f$ = {max . $A_c$, max . $F_m$}/* $P_F$ represents the Pareto front. */
input: COVID-19 training dataset, CNN, random population
begin
    Set random solution as hyperparameters of CNN;
    Apply CNN on COVID-19 training dataset;
    Validate CNN on the same fraction of COVID-19 training dataset;
    Evaluate confusion matrix based on the actual and predicted values;
    max . $A_c$ = $A_c$ (Confusion matrix);
    max . $F_m$ = $F_m$ (Confusion matrix);
    return{max . $A_c$, max . $F_m$}
end
        Note. $P_F$ represents the Pareto front.
```

Algorithm 2: Multiobjective optimization.

```
output: optimized population
input: {Fitness function, demand, crossover_ratio}
begin
    Generate the random $I_P$; /*$I_P$ represents the initial population */;
    Calculate the fitness of $I_P$;
    Sort $I_P$;
    /* Selection */
    set $F_P = I_P$; /*$F_P$ denotes final population */
    while $c_e$ ! = 0$orl_G$ == true do
        /*$c_e$ and $l_G$ represent children elimination and last generation */
        Generate random $c$; /*$c$ denotes children */
        set $c_e = 0$;
        for eachc do
            Compute the fitness of $c$;
            if $f_c \le f_{F_P[1]}$ then
                remove $c$;
                set $c_e$+ = 1;
            else
                set $F_P = c$;
            end
        end
        /* Mutation */
        for crossover do
            select $c_1$ and $c_2$ randomly; /*$c_1$, $c_2$, and $c_3$ are children */
            $c_3 = c_1 \oplus c_2$;
            Evaluate the fitness of $c_3$;
            if $f_{c_3} \le f_{c_1}$ or $f_{c_2}$then
                remove $c_3$;
            else
                remove $c_1, c_2$;
            end
        end
        next generation
    end
    /* Ranking */ sort the $F_P$;
    return$F_P$[1] /* returns the most dominant solution w.r.t. fitness function */
end
```

Algorithm 3: Hyperparameter tuning of CNN using multiobjective genetic algorithm.

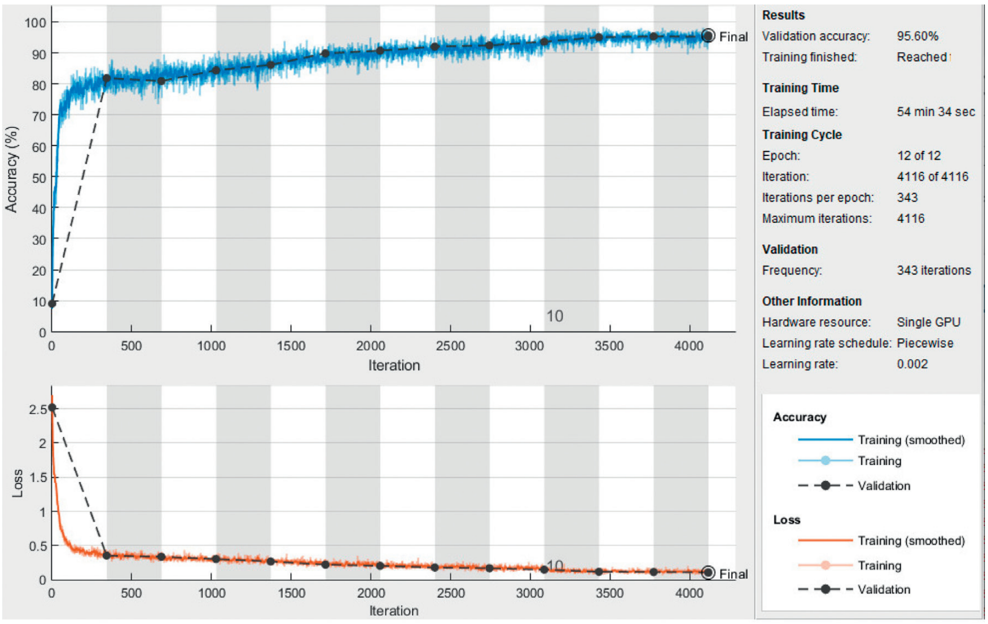
The genetic algorithm contains a group of operators to optimize the given fitness function [31]. Initially, random solutions are obtained using a normal distribution. These solutions are then applied to CNN for evaluating the multiobjective fitness function (see equation (3)). Based on computed values, solutions are ranked for further processing. Thereafter, mutation and crossover operators are applied to the solutions for obtaining child values. Based upon their fitness values, they are



(a)

(b)

Figure 4: Chest X-ray image dataset: (a) normal persons' X-ray images, i.e., COVID-19 (−); (b) COVID-19-infected patients' X-ray images, i.e., COVID-19 (+).



Results
Validation accuracy: 95.60%
Training finished: Reached

Training Time
Elapsed time: 54 min 34 sec

Training Cycle
Epoch: 12 of 12
Iteration: 4116 of 4116
Iterations per epoch: 343
Maximum iterations: 4116

Validation
Frequency: 343 iterations

Other Information
Hardware resource: Single GPU
Learning rate schedule: Piecewise
Learning rate: 0.002

Figure 5: Training and validation analysis of the proposed COVID-19 identification model in terms of accuracy and loss.

ranked [32]. Finally, the most nondominated solution is returned as initial parameters of CNN.

4. Performance Analysis

This section discusses the performance analysis of the COVID-19 identification model. This work uses 20-fold cross-validation to overcome the overfitting problem. 70% of the entire dataset is considered for training purpose. The hyperparameters of the proposed COVID-19 identification model are obtained using a multiobjective genetic algorithm.

4.1. Chest X-Ray Image Dataset. To enhance prognostic analysis, triage and manage patient care, data is the first step for building any identification tool. Therefore, COVID-19 chest X-rays are collected to build COVID-19 identification models. Chest X-ray images of COVID-19-infected patients contain many unique characteristics. Therefore, chest X-ray images may be utilized to diagnose COVID-19-infected patients at a rapid speed.

In this paper, the chest X-ray images are obtained from several datasets such as from [2, 33]; there are 1332 COVID-19 (+) images and 1421 images of normal or pneumonia-infected patients. Figure 4 shows a partial set of X-ray images

Table 1: Training analysis of the COVID-19 identification models by utilizing confusion matrix-based performance metrics when training to testing ratio is 60 : 40.

Model	Accuracy	*F*-measure	Specificity	Sensitivity	AUC
SVM	0.872881	0.871186	0.871404	0.872666	0.872034
ANFIS	0.884746	0.883051	0.883249	0.868455	0.883898
CNN	0.894661	0.894915	0.895093	0.896435	0.895763
AlexNet	0.908475	0.906678	0.906937	0.908319	0.907627
ResNet-34	0.920339	0.918644	0.918782	0.920204	0.919492
GoogLeNet	0.932203	0.930508	0.930626	0.932088	0.931356
VGG-16	0.944068	0.942373	0.942747	0.943973	0.947322
ResNet-50	0.955932	0.954237	0.954315	0.955857	0.955085
Xception	0.967797	0.966102	0.966159	0.967742	0.966949
DenseNet201	0.979661	0.977966	0.978003	0.979626	0.978814
Proposed model	0.991525	0.989831	0.989848	0.991511	0.990678

Table 2: Testing (i.e., validation) analysis of the COVID-19 identification models by utilizing confusion matrix-based performance metrics when training to testing ratio is 60 : 40.

Model	Accuracy	*F*-measure	Sensitivity	Specificity	AUC
SVM	0.857627	0.864407	0.863481	0.858586	0.861017
ANFIS	0.869492	0.876271	0.875427	0.870537	0.872881
CNN	0.881356	0.888136	0.887372	0.882155	0.884746
AlexNet	0.893522	0.932324	0.899317	0.893939	0.895661
ResNet-34	0.905085	0.911864	0.911263	0.905724	0.908475
GoogLeNet	0.916949	0.923729	0.923208	0.917508	0.920339
VGG-16	0.928814	0.935593	0.935154	0.929293	0.932203
ResNet-50	0.940678	0.947458	0.947099	0.941077	0.944068
Xception	0.952542	0.959322	0.959044	0.952862	0.955932
DenseNet201	0.964407	0.971186	0.970949	0.964646	0.967797
Proposed model	0.976271	0.983051	0.982935	0.976431	0.979661

of normal persons and COVID-19-infected patients. It clearly shows that there is a significant change in the X-ray images of normal and COVID-19-infected patients.

4.2. Comparative Analysis. The performance of the proposed model is compared to various machine learning and deep learning approaches. The overall objective is to evaluate the significant improvement of the proposed model against various performance metrics such as accuracy, Area Under the Curve (AUC), *F*-measure, specificity, and sensitivity.

The training and validation analysis of the proposed COVID-19 identification model is illustrated in Figure 5. It demonstrates that the proposed COVID-19 identification model achieves significant training and validation accuracy values. It also indicates that the loss of the proposed COVID-19 identification model is minimum. Further, as the number of epoch increases, it shows improvement in results, but after 3300 epochs, it seems to be constant, i.e., not much improvement in results is observed.

Tables 1 and 2 show model building and testing analysis among the proposed and the existing COVID-19 identification models.

Table 1 reveals that the proposed model achieves significant performance in terms of accuracy, Area Under the Curve (AUC), *F*-measure, specificity, and sensitivity as compared to the existing models.

5. Conclusion

In this paper, a CNN model is used to build COVID-19 identification model using the chest X-ray images. 20-fold cross-validation is used to overcome the overfitting problem. A pretrained GoogLeNet is also considered for implementing the transfer learning (i.e., by replacing some sets of final network CNN layers). Finally, the multiobjective genetic algorithm is used for hyperparameter tuning of the COVID-19 identification model. Performance analysis revealed that the COVID-19 identification model attains significantly good performance than the competitive models. The proposed COVID-19 identification model offered training and testing accuracy up to 98.3827% and 94.9383%, respectively. Thus, the designed identification model can be used for real-time identification of COVID-19 disease.

Data Availability

Data will be made available upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] E. E.-D. Hemdan, M. A. Shouman, and M. E. Karar, "Covidx-net: a framework of deep learning classifiers to diagnose COVID-19 in X-ray images," 2020, http://arxiv.org/abs/2003.11055.

[2] D. Singh, V. Kumar, Vaishali, and M. Kaur, "Classification of COVID-19 patients from chest CT images using multi-objective differential evolution-based convolutional neural networks," *European Journal of Clinical Microbiology & Infectious Diseases*, vol. 39, no. 7, pp. 1379–1389, 2020.

[3] S. Ayyachamy, V. Alex, M. Khened, and G. Krishnamurthi, "Medical image retrieval using Resnet-18," in *Medical Imaging 2019: Imaging Informatics for Healthcare, Research, and Applications*, vol. 10954, Bellingham, WA, USA, International Society for Optics and Photonics, Article ID 1095410, 2019.

[4] Y. Pathak, K. V. Arya, and S. Tiwari, "An efficient low-dose ct reconstruction technique using partial derivatives based guided image filter," *Multimedia Tools and Applications*, vol. 78, no. 11, pp. 14733–14752, 2019.

[5] Y. Yu, H. Lin, J. Meng, X. Wei, H. Guo, and Z. Zhao, "Deep transfer learning for modality classification of medical images," *Information*, vol. 8, no. 3, p. 91, 2017.

[6] Y. Pathak, P. K. Shukla, A. Tiwari, S. Stalin, S. Singh, and P. K. Shukla, "Deep transfer learning based classification model for COVID-19 disease," *Biomedical Engineering and Research*, 2020, In press.

[7] H. S. Pannu, D. Singh, and A. K. Malhi, "Improved particle swarm optimization based adaptive neuro-fuzzy inference system for benzene detection," *Clean—Soil, Air, Water*, vol. 46, no. 5, Article ID 1700162, 2018.

[8] P. Tang, H. Wang, and S. Kwong, "G-MS2F: GoogLeNet based multi-stage feature fusion of deep CNN for scene recognition," *Neurocomputing*, vol. 225, pp. 188–197, 2017.

[9] F. Gao, T. Wu, J. Li et al., "SD-CNN: a shallow-deep CNN for improved breast cancer diagnosis," *Computerized Medical Imaging and Graphics*, vol. 70, pp. 53–62, 2018.

[10] S. Deepak and P. M. Ameer, "Brain tumor classification using deep CNN features via transfer learning," *Computers in Biology and Medicine*, vol. 111, Article ID 103345, 2019.

[11] A. Çinar and M. Yildirim, "Detection of tumors on brain MRI images using the hybrid convolutional neural network architecture," *Medical Hypotheses*, vol. 139, Article ID 109684, 2020.

[12] D. R. Nayak, R. Dash, and B. Majhi, "Automated diagnosis of multi-class brain abnormalities using MRI images: a deep convolutional neural network based method," *Pattern Recognition Letters*, vol. 138, 2020.

[13] B. Liu, Q. Liu, Z. Zhu, T. Zhang, and Y. Yang, "MSST-Resnet: deep multi-scale spatiotemporal features for robust visual object tracking," *Knowledge-Based Systems*, vol. 164, pp. 235–252, 2019.

[14] W. Hao, R. Bie, J. Guo, X. Meng, and S. Wang, "Optimized CNN based image recognition through target region selection," *Optik*, vol. 156, pp. 772–777, 2018.

[15] S. Taheri and Ö. Toygar, "On the use of dag-CNN architecture for age estimation with multi-stage features fusion," *Neurocomputing*, vol. 329, pp. 300–310, 2019.

[16] G. Ciocca, P. Napoletano, and R. Schettini, "CNN-based features for retrieval and classification of food images," *Computer Vision and Image Understanding*, vol. 176-177, pp. 70–77, 2018.

[17] X. Liu, Y. Zhou, J. Zhao et al., "Multiobjective ResNet pruning by means of EMOAs for remote sensing scene classification," *Neurocomputing*, vol. 381, pp. 298–305, 2020.

[18] C. Han and L. Shi, "ML-ResNet: a novel network to detect and locate myocardial infarction using 12 leads ECG," *Computer Methods and Programs in Biomedicine*, vol. 185, Article ID 105138, 2020.

[19] M. Talo, O. Yildirim, U. B. Baloglu, G. Aydin, and U. R. Acharya, "Convolutional neural networks for multi-class brain disease detection using MRI images," *Computerized Medical Imaging and Graphics*, vol. 78, Article ID 101673, 2019.

[20] N. N. Das, N. Kumar, M. Kaur, V. Kumar, and D. Singh, "Automated deep transfer learning-based approach for detection of COVID-19 infection in chest X-rays," *Biomedical Engineering and Research*, 2020, In press.

[21] S. Liu, G. Tian, and Y. Xu, "A novel scene classification model combining ResNet based transfer learning and data augmentation with a filter," *Neurocomputing*, vol. 338, pp. 191–206, 2019.

[22] M. Togacar, B. Ergen, and Z. Cömert, "COVID-19 detection using deep learning models to exploit social mimic optimization and structured chest X-ray images using fuzzy color and stacking approaches," *Computers in Biology and Medicine*, vol. 121, Article ID 103805, 2020.

[23] H. S. Pannu, D. Singh, and A. K. Malhi, "Multi-objective particle swarm optimization-based adaptive neuro-fuzzy inference system for benzene monitoring," *Neural Computing and Applications*, vol. 31, no. 7, pp. 2195–2205, 2019.

[24] S. Saha, K. M. Khabir, S. S. Abir, and A. Islam, "A newly proposed object detection method using faster R-CNN inception with ResNet based on Tensorflow," in *Real-Time Image Processing and Deep Learning 2019, vol. 10996*, International Society for Optics and Photonics, Bellingham, WA, USA, Article ID 109960X, 2019.

[25] T. Wiens, "Engine speed reduction for hydraulic machinery using predictive algorithms," *International Journal of Hydromechatronics*, vol. 2, no. 1, pp. 16–31, 2019.

[26] R. Wang, H. Yu, G. Wang, G. Zhang, and W. Wang, "Study on the dynamic and static characteristics of gas static thrust bearing with micro-hole restrictors," *International Journal of Hydromechatronics*, vol. 2, no. 3, pp. 189–202, 2019.

[27] S. Osterland and J. Weber, "Analytical analysis of single-stage pressure relief valves," *International Journal of Hydromechatronics*, vol. 2, no. 1, pp. 32–53, 2019.

[28] C. Cao, Y. Huang, Z. Wang, L. Wang, N. Xu, and T. Tan, "Lateral inhibition-inspired convolutional neural network for visual attention and saliency detection," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, New Orleans, LA, USA, February 2018.

[29] G. Qi, H. Wang, M. Haner, C. Weng, S. Chen, and Z. Zhu, "Convolutional neural network based detection and judgement of environmental obstacle in vehicle operation," *CAAI Transactions on Intelligence Technology*, vol. 4, no. 2, pp. 80–91, 2019.

[30] C. Zhu, W. Yan, X. Cai, S. Liu, T. H. Li, and G. Li, "Neural saliency algorithm guide bi-directional visual perception style transfer," *CAAI Transactions on Intelligence Technology*, vol. 5, no. 1, pp. 1–8, 2020.

[31] V. Roostapour, A. Neumann, and F. Neumann, "Evolutionary multi-objective optimization for the dynamic knapsack problem," 2020, http://arxiv.org/abs/2004.12574.

[32] X. Xue, J. Lu, and J. Chen, “Using NSGA-III for optimising biomedical ontology alignment,” *CAAI Transactions on Intelligence Technology*, vol. 4, no. 3, pp. 135–141, 2019.

[33] J. P. Cohen, P. Morrison, and L. Dao, “COVID-19 Image Data Collection,” 2020, http://arxiv.org/abs/2003.11597.

Hindawi
Mathematical Problems in Engineering
Volume 2021, Article ID 1305849, 10 pages
https://doi.org/10.1155/2021/1305849

Research Article

Multiobjective Real-Time Scheduling of Tasks in Cloud Manufacturing with Genetic Algorithm

Gilseung Ahn and Sun Hur

Department of Industrial and Management Engineering, Hanyang University, Ansan 15588, Republic of Korea

Correspondence should be addressed to Sun Hur; hursun@hanyang.ac.kr

Received 20 April 2020; Accepted 21 September 2020; Published 15 February 2021

Academic Editor: Manjit Kaur

In cloud manufacturing, customers register customized requirements, and manufacturers provide appropriate services to complete the task. A cloud manufacturing manager establishes manufacturing schedules that determine the service provision time in a real-time manner as the requirements are registered in real time. In addition, customer satisfaction is affected by various measures such as cost, quality, tardiness, and reliability. Thus, multiobjective and real-time scheduling of tasks is important to operate cloud manufacturing effectively. In this paper, we establish a mathematical model to minimize tardiness, cost, quality, and reliability. Additionally, we propose an approach to solve the mathematical model in a real-time manner using a multiobjective genetic algorithm that includes chromosome representation, fitness function, and genetic operators. From the experimental results, we verify whether the proposed approach is effective and efficient.

1. Introduction

Cloud manufacturing (CM) is an advanced model that provides a cloud-based manufacturing platform on which enterprises virtualize and share their manufacturing services such as welding, milling, and machining to produce highly customized products [1–3]. A task to produce a customized product consists of several activities, and each activity provides a related manufacturing service. The task is processed in the following order [4]. First, a platform manager determines the manufacturing services and an appropriate processing order. This step is called a task definition. The manager then establishes a schedule by selecting services from the platform and determines the start times of each service, considering availability, processing time, and service order. This step, called task scheduling, differs from service allocation in that it considers time [5]. In other words, it allows duplicate assignment of a service (in this case, delay may occur), while service allocation does not. Finally, the task sequence begins processing according to the schedule.

Because task scheduling is an important operational problem in CM, many researchers have recently addressed it. Zhou et al. [6] modeled a scheduling problem to minimize the weighted sum of differences between requirement levels and actual levels in terms of time, cost, and quality, and service type is considered a major constraint. They improved a genetic algorithm (GA) to solve the scheduling problem and compared it with particle swarm optimization (PSO) and simulated annealing in an experiment in which each task was given a different weight. They argued that their proposed algorithm outperformed other algorithms in terms of objective value. Cao et al. [7] investigated a task-scheduling problem in terms of time, quality, cost, and service and adopted fuzzy decision-making theory to transform these criteria into degrees of relative superiority. They presented a mathematical model that maximizes the weighted sum of the relative degrees and applied ant colony optimization (ACO) to obtain a solution. Through experiments, they showed that ACO outperformed GA and PSO in every iteration. Liu et al. [8] revealed that workloads affect total completion time, service utilization, and so forth in task-scheduling problems. They compared two scheduling methods based on workload. The first completes the tasks with larger workloads first, while the second completes tasks with smaller workloads first. They concluded that the first method yields a superior performance. Jiang et al. [9] addressed a task-scheduling problem in CM that specializes in disassembly, with the objective of minimizing both total expected makespan and cost. They designed a multiobjective algorithm using the nondominated sorting genetic algorithm II (NSGA-II) to solve the task-scheduling problem. Li et al. [10] studied a task-scheduling problem for distributed robots in CM in which tasks were scheduled by allocating a robot to a subtask, considering geographical location. They solved the problem from three perspectives: difference of workload among robots, overall cost, and overall processing time with a GA.

Although previous research investigated task scheduling from multiple perspectives, the chosen approaches were unrealistic in terms of multiobjective and real-time characteristics. Scheduling problems should consider two or more objectives simultaneously. Some researches such as in [6] considered multiple objectives as a weighted sum, but they also were not practical as it proved difficult to choose the proper weight. Only a few (e.g., [9]) considered simultaneous objectives, when modeling the scheduling problem. In addition, the proper services should be used in real-time process (sub) tasks whenever they are registered because this aspect is an important characteristic of CM platform operations.

Meta-heuristic algorithms have been frequently employed to solve computational engineering problems [11–17]. For example, Hoang [11] integrated history-based adaptive differential evolution and linear population size reduction to find the optimal hyperparameters of the support vector machine for pitting corrosion detection. Chen et al. [12] developed a hybrid of variable neighborhood search (VNS) and estimation of distribution algorithm (EDA), called VNS-EDA, to compose the feature subset for credit risk classification. Jiang et al. [18] improved particle swarm optimization (PSO) by integrating with the gravitational search algorithm with dependent random coefficients. Peng et al. [19] introduced chaotic search for the fuzzy neural network to improve PSO to handle complex engineering problems.

A GA is one of the most famous meta-heuristic algorithms to solve diverse scheduling problems such as project scheduling [20, 21], job-shop scheduling [22], parallel machine scheduling [23], and flow-shop scheduling [24, 25] and usually performs well. In addition, a multiobjective genetic algorithm (MOGA) has also been successfully applied to various multiobjective problems, such as a scheduling problem to minimize both production cost and time. For example, Tang et al. [26] addressed a multiobjective radio frequency identification network-planning problem with the objective of minimizing collision and interference of the network and network cost. They integrated a divide and conquer greedy heuristic algorithm and an MOGA to solve the problem. Zhang et al. [27] addressed multiobjective assembly line-balancing problems to minimize cycle time and rebalancing cost by modifying NSGA-II. Because GA is appropriate for handling the multiobjective scheduling problem, as shown in many previous researches, we choose GA to solve a multiobjective real-time task-scheduling problem in CM.

In this paper, we study a multiobjective real-time task-scheduling problem in CM where tasks are scheduled whenever they are registered. The major research contents and contributions are as follows. First, we formalized the scheduling problem as a binary integer programming model with four objectives to minimize total tardiness, cost, quality, and reliability penalties. In addition, we design an MOGA to solve the problem by focusing on feasibility because most solutions to the problem are infeasible. Finally, we conduct an experiment to verify the proposed approach's performance. Table 1 summarizes how the contribution of our study extends previous research results.

The rest of this paper is organized as follows. Section 2 describes the problem and introduces a mathematical model. Section 3 develops the proposed approach by focusing on designing a multiobjective GA that includes chromosome representation, a fitness function, and genetic operators. Section 4 conducts the experiments and compares the efficiency of the GA, and Section 5 concludes the paper.

2. Problem Description and Mathematical Model

2.1. Notation. Table 2 shows notations used in this paper.

2.2. Problem Description. In cloud manufacturing, customers upload their requirements in the cloud-based manufacturing platform, and then the requirements are converted into a task, each of which consists of several activities and saved in the task pool. Enterprises virtualize and upload their manufacturing services (e.g., milling and cutting), and the services are saved in the service pool. The task-scheduling problem is to make a schedule for each task in the task pool by assigning a set of proper manufacturing services in the service pool. The considered task-scheduling problem is to assign a proper service to each activity of every task to minimize total tardiness and cost, quality, and reliability penalties at each time. Figure 1 shows a typical example of the structure of a task-scheduling problem. The schedule is constructed by a manager (or management system) on a cloud-based manufacturing platform.

Each task is composed of several activities in sequence or in parallel. Task T_2^t, in Figure 1, for example, is composed of three activities: $A_{2,1}^t$, $A_{2,2}^t$, and $A_{2,3}^t$. ($A_{2,1}^t$, $A_{2,2}^t$) are composed sequentially, while ($A_{2,2}^t$, $A_{2,3}^t$) are in parallel. The required service type differs according to the activity. For example, if S_k is the service type needed by $A_{i,j}^t$, then one of $S_{k,v}$ ($v = 1, 2, \ldots, |S_k|$) can be matched to $A_{i,j}^t$. Each activity can be processed only after the preceding activities have been completed. The activity without preceding activities (the root activity) can be processed only after the task to which it belongs is registered. Finally, services may have different performance metrics such as cost and quality even though they are of the same type. Services should be scheduled considering all required levels of tasks.

Table 1: Contribution of this paper to previous research.

Previous study	Considering multiobjective	Considering real-time	Considering various composition types	Improving meta-heuristic algorithm
Zhou et al. [6]		√	√	√
Cao et al. [7]		√	√	√
Liu et al. [8]	√		√	
Jiang et al. [9]	√	√		√
Li et al. [10]	√		√	
This paper	√	√	√	√

Table 2: Notations.

Notation		Description
Service related	S_k	Type k service, $k = 1, 2, \ldots, l$
	$S_{k,v}$	v^{th} type k service, $v = 1, 2, \ldots, \lvert S_k \rvert$
	$c_{k,v}$	Cost of $S_{k,v}$
	$q_{k,v}$	Quality of $S_{k,v}$
	$r_{k,v}$	Reliability of $S_{k,v}$
	$p_{k,v}$	Processing time of $S_{k,v}$
Task and activity related	T_i^t	Task i registered at time $t = 1, 2, \ldots, T$, $i = 1, 2, \ldots, n_t$, where n_t is the number of tasks registered at time t
	$A_{i,j}^t$	Activity j in task T_i^t, $j = 1, 2, \ldots, m_i^t$, where m_i^t is the number of activities in T_i^t
	$\text{Type}(A_{i,j}^t)$	Service type required for $A_{i,j}^t$, $\text{type}(A_{i,j}^t) \in \{S_1, S_2, \ldots, S_l\}$
	$P_{i,j}^t$	Set of precedent activities of $A_{i,j}^t$
	$F_{i,j}^t$	Set of following activities of $A_{i,j}^t$
	D_i^t	Due date of T_i^t
	C_i^t	Maximum allowed cost to complete T_i^t
	Q_i^t	Minimum allowed quality of T_i^t
	R_i^t	Minimum allowed reliability of T_i^t
	$\widehat{D}_i^t$	Actual delivery date of T_i^t
	$\widehat{C}_i^t$	Actual cost to complete T_i^t
	$\widehat{Q}_i^t$	Actual quality of T_i^t
	$\widehat{R}_i^t$	Actual reliability of T_i^t
	$\widetilde{T}_i^t$	Tardiness penalty on T_i^t
	$\widetilde{C}_i^t$	Cost penalty on T_i^t
	$\widetilde{Q}_i^t$	Quality penalty on T_i^t
	$\widetilde{R}_i^t$	Reliability penalty on T_i^t
Decision variables	$x_{i,j,k,v}^{t,\tau}$	= 1, if $S_{k,v}$ is matched to $A_{i,j}^t$ at time $\tau \geq t$, and = 0, otherwise
	$I(S_{k,v}^t)$	= 1, if $S_{k,v}$ is available at time t, and = 0, otherwise

Additional assumptions of the problem are summarized as follows:

(1) Required service type to process each activity is known

(2) The maximum allowed cost, minimum allowed quality and reliability, and due date are given

(3) There is no rework for any activity

(4) Customer satisfaction can be measured by cost, quality, reliability, and due date

(5) Geographical distances among service providers are not considered

(6) The outcome of a task is delivered to the customer once the task is completed

2.3. Mathematical Model. We describe objective functions and constraints of the established mathematical model in Sections 2.3.1 and 2.3.2, respectively.

2.3.1. Objective Functions. The objectives are to minimize the total penalty associated with tardiness, cost, quality, and reliability. In this section, we explain how to obtain objective values from the service schedule.

Actual delivery time $\widehat{D}_i^t$ of task T_i^t is the maximum among completion times of activities that do not have the following activities. Because completion time of $A_{i,j}^t$ is calculated as the sum of service assignment time and processing time, $\widehat{D}_i^t$ is obtained as follows:

$$\widehat{D}_i^t = \max_{F_{i,j}^t=\varnothing}\left(\sum_{k=Type\left(A_{i,j}^t\right),\forall v,\tau\geq t}\left(x_{i,j,k,v}^{t,\tau}\times\left(\tau+p_{k,v}-1\right)\right)\right), \tag{1}$$

where $F_{i,j}^t$ and $p_{k,v}$ denote the set of $A_{i,j}^t$'s following activities and the processing time of $S_{k,v}$, respectively. With equation (1), the tardiness penalty $\widetilde{T}_i^t$ of T_i^t is computed as the difference

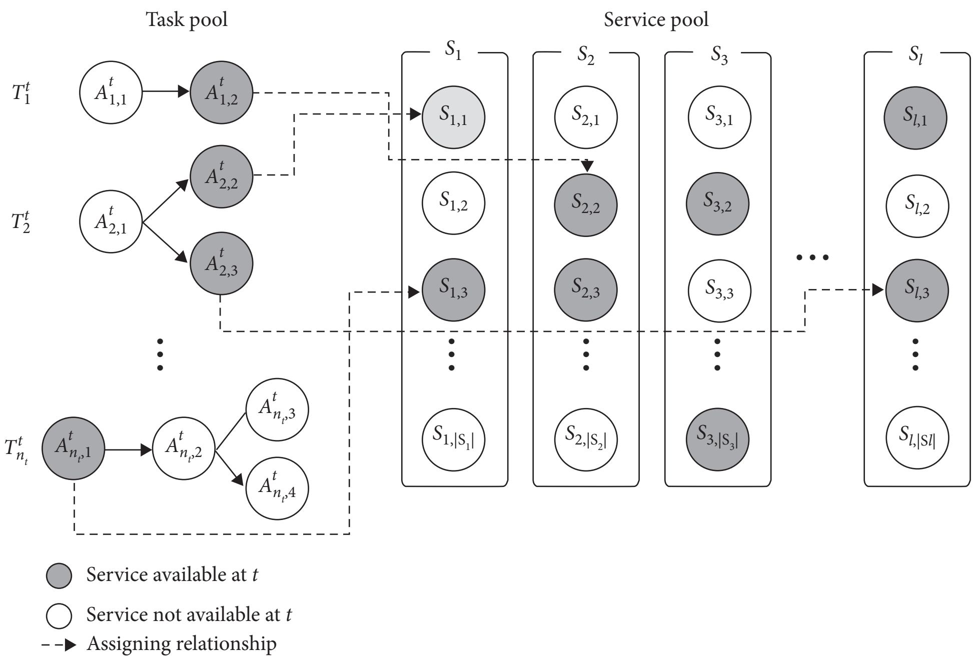


Figure 1: Typical example of task scheduling.

between $\widehat{D}_i^t$ and D_i^t if it is bigger than zero, and zero otherwise:

$$\widetilde{T}_i^t = \max\left(\widehat{D}_i^t - D_i^t, 0\right), \tag{2}$$

where D_i^t is the due date of T_i^t. By expanding equation (2) to all tasks registered at t, we have the first objective function, which is the sum of tardiness penalty for every task registered at t:

$$\text{minimize } Z_1 = \sum_{i=1}^{n_t} \widetilde{T}_i^t. \tag{3}$$

The actual cost $\widehat{C}_i^t$ to complete T_i^t is calculated as the sum of costs of services assigned to the task as given by

$$\widehat{C}_i^t = \sum_{j=1}^{m(t/i)} \sum_{k=Type\left(A_{i,j}^t\right),\forall v} \left(c_{k,v} \times x_{i,j,k,v}^{t,\tau}\right), \tag{4}$$

with equation (4), the cost penalty cp_i^t on T_i^t is

$$\widetilde{C}_i^t = \max\left(\widehat{C}_i^t - C_i^t, 0\right), \tag{5}$$

where C_i^t is the maximum allowed cost to complete T_i^t. By expanding equation (5) to all tasks, the second objective function is obtained as follows:

$$\text{minimize } Z_2 = \sum_{i=1}^{n_t} \widetilde{C}_i^t. \tag{6}$$

Similarly, the actual quality $\widehat{Q}_i^t$ of T_i^t is the average of qualities of services, and actual reliability $\widehat{R}_i^t$ is the product of reliabilities of services assigned to the task as given by

$$\widehat{Q}_i^t = \frac{\sum_{j=1}^{m(t/j)} \sum_{k=\mathrm{Type}\left(A_{i,j}^t\right),\forall v} \left(q_{k,v} \times x_{i,j,k,v}^{t,\tau}\right)}{m_j^t},$$
$$\widehat{R}_i^t = \prod_{j=1}^{m(t/j)} \sum_{k=Type\left(A_{i,j}^t\right),\forall v} \left(r_{k,v} \times x_{i,j,k,v}^{t,\tau}\right), \tag{7}$$

where $\sum_{k=\mathrm{Type}(A_{i,j}^t),\forall v} (r_{k,v} \times x_{i,j,k,v}^{t,\tau})$ is the reliability of the service assigned to activity $A_{i,j}^t$.

The quality penalty is $\widetilde{Q}_i^t = \max(Q_i^t - \widehat{Q}_i^t, 0)$, and the reliability penalty is $\widetilde{R}_i^t = \max(R_i^t - \widehat{R}_i^t, 0)$. The third and fourth objective functions become (8) and (9), respectively:

$$\text{minimize } Z_3 = \sum_{i=1}^{n_t} \widetilde{Q}_i^t, \tag{8}$$

$$\text{minimize } Z_4 = \sum_{i=1}^{n_t} \widetilde{R}_i^t. \tag{9}$$

2.3.2. Constraints. Constraints are related with service type, activity precedence, and service availability. Services of the required type can be assigned to an activity; therefore,

$$\sum_{v=1}^{S_k}\sum_{\tau\geq t}^{T} x_{i,j,k,v}^{t,\tau} = 1, \quad \forall\text{Type}\left(A_{i,j}^{t}\right) = k\ (k = 1, 2, \ldots, l). \quad (10)$$

Another constraint is the activity precedence. A service cannot be allocated to an activity with a preceding activity that has not been completed, which is expressed as follows:

$$\tau_0 \leq \tau_1, \quad \text{if } x_{i,j_0,k_0,v_0}^{t,\tau_0} \times x_{i,j_1,k_1,v_1}^{t,\tau_1} = 1, \text{and } A_{i,j_0}^{t} \in P_{i,j_1}^{t}, \quad (11)$$

where τ_0 is the assignment time for a preceding activity with assignment time τ_1. $x_{i,j_0,k_0,v_0}^{t,\tau_0} \times x_{i,j_1,k_1,v_1}^{t,\tau_1} = 1$ implies that S_{k_0,v_0} and S_{k_1,v_1} are assigned to A_{i,j_0}^{t} and A_{i,j_1}^{t} at τ_0 and τ_1, respectively.

The next constraint is service availability. A service cannot be allocated to other activities until it is released:

$$\sum_{w=0}^{p_{k,v}}\sum_{i,j} x_{i,j,k,v}^{t,\tau-w} \leq 1, \quad \forall\tau, T_i^t, j, k, v. \quad (12)$$

Constraint (12) can also be expressed by means of indicator variables as follows:

$$I\left(S_{k,v}^{\tau}\right) \geq x_{i,j,k,v}^{t,\tau}, \quad \forall\tau, T_i^t, j, k, v. \quad (13)$$

In other words, $x_{i,j,k,v}^{t,\tau} = 0$ if $I(S_{k,v}^{\tau}) = 0$, and $x_{i,j,k,v}^{t,\tau} = 1$ otherwise. The indicator $I(S_{k,v}^{t})$ can prevent the system from allocating an unavailable service and resolve an unexpected situation in which a service becomes unavailable suddenly. For example, a service $S_{k,v}$ that is supposed to be allocated to a task at time $\tau \geq t$ may suddenly become unavailable because of a service-provider issue (e.g., contract expiration). The proposed GA can resolve this by setting $I(S_{k,v}^{\tau}) = 0$ for $\tau \geq t$ and then continuing to search the solutions.

Finally, because every decision variable is binary, we need the following:

$$x_{i,j,k,v}^{t,\tau} \in \{0, 1\}, \quad \forall\tau, T_i^t, k, v, j. \quad (14)$$

3. Genetic Algorithm

This section describes the design procedure of the multiobjective GA to solve the mathematical model developed in the previous section. Figure 2 shows an illustrative flowchart of the proposed multiobjective GA, with which the schedule for a task T_i^t is constructed in the order of the registered time. That is, the proposed GA selects the proper services for a task, which are scheduled to process the task and then service availability is updated. Therefore, a task can be scheduled as soon as it is uploaded (that is, task can be scheduled in a real-time manner).

3.1. Chromosome Representation. The h^{th} chromosome in the g^{th} population is represented as $\chi^t[g][h] = (\chi_1^t[g][h], \chi_2^t[g][h], \ldots, \chi_{n_t}^t[g][h])$, where $\chi_i^t[g][h]$ is a chromosome for scheduling T_i^t. $\chi_i^t[g][h]$ consists of $\chi_{i,1}^t[g][h]$, $\chi_{i,2}^t[g][h]$, $\ldots$, $\chi_{i,m_i^t}^t[g][h]$, and $\chi_{i,1}^t[g][h]$ has two elements such as

$$\chi_{i,j}^t[g][h] = \left(v_{i,j}^t[g][h], \tau_{i,j}^t[g][h]\right), \quad (15)$$

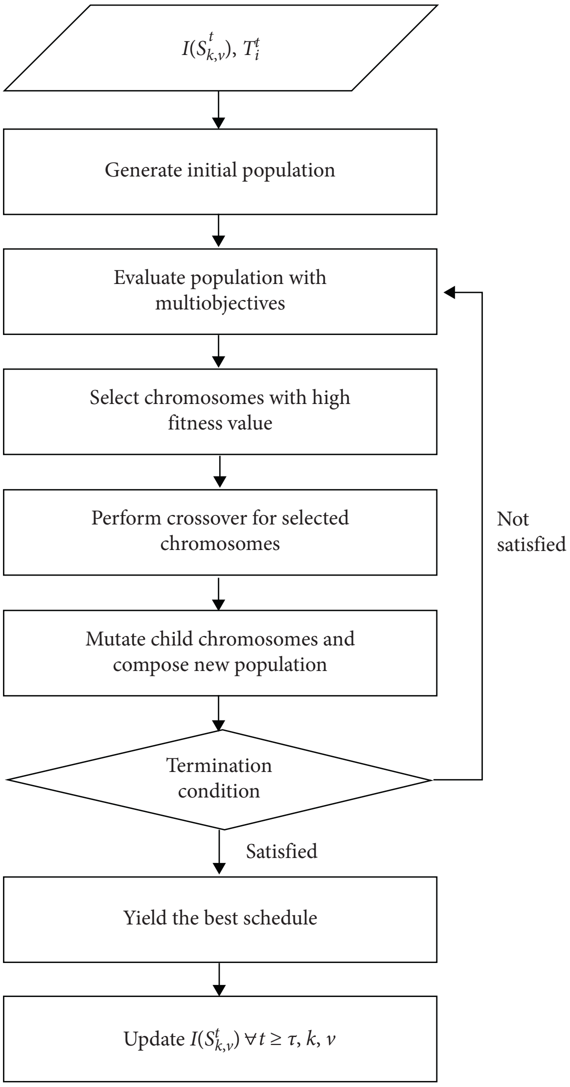

$I(S_{k,v}^t)$, T_i^t
Generate initial population
Evaluate population with multiobjectives
Select chromosomes with high fitness value
Perform crossover for selected chromosomes
Mutate child chromosomes and compose new population
Termination condition
Not satisfied
Satisfied
Yield the best schedule
Update $I(S_{k,v}^t)\ \forall t \geq \tau, k, v$

Figure 2: Flowchart of the proposed approach.

where $v_{i,j}^t[g][h]$ is a service assigned to the activity and $\tau_{i,j}^t[g][h] = (\tau\tau_{i,j}^{t,s}[g][h], \tau_{i,j}^{t,s}[g][h] + 1, \ldots, \tau_{i,j}^{t,e}[g][h])$ is a processing period of $A_{i,j}^t$ by $v_{i,j}^t[g][h]$. Here, $\tau_{i,j}^{t,s}[g][h]$ is the starting time and $\tau_{i,j}^{t,e}[g][h] = \tau_{i,j}^{t,s}[g][h] + p_{k,v} - 1$ is the completion time of $A_{i,j}^t$.

3.2. Generation of Initial Solution. If the GA generates initial solutions randomly, then convergence would take an extended amount of time because most solutions are infeasible. We therefore propose a method to generate initial solutions, $\chi^t[1][h]$, that are feasible, through Algorithm 1.

This algorithm generates solutions satisfying the constraints in Section 3.3 and operates ps times to generate $\chi^t[1][1], \chi^t[1][2], \ldots, \chi^t[1][ps]$, where ps is the number of chromosomes in a population.

3.3. Fitness Function. Pareto ranking is employed as a fitness function to select ps chromosomes and to yield the best chromosome in each generation. This method calculates a fitness value $s^t[g][h]$ of $\chi^t[g][h]$ as follows [28]:

```
Input: $I(S_{k,\nu}^{\tau})$ for all $\tau \geq t, k, \nu$
Procedure:
    Step 1. $i = 1$.
    Step 2. $j = 1$, and save the service type for $A_{i,j}^{t}$ as $k$, and randomly select $\nu_{i,j}^{t}$ and $\tau_{i,j}^{t,s}$ which satisfies:
        (1) $I(S_{k,\nu}^{\tau}) = 1$ for $\tau = \tau_{i,j}^{s}, \tau_{i,j}^{s} + 1, \ldots, \tau_{i,j}^{s} + p_{k,\nu} - 1$, and
        (2) $\tau_{i,j}^{s}$ is one of minimums among $\tau$ satisfying (1), and update $I(S_{k,\nu}^{\tau}) = 0$ for $\nu = \nu_{i,j}^{t}$ and
            $\tau = \tau_{i,j}^{s}, \tau_{i,j}^{s} + 1, \ldots, \tau_{i,j}^{s} + p_{k,\nu} - 1$.
    Step 3. Increase $j$ by 1 and save the service type for $A_{i,j}^{t}$ as $k$.
    Step 4. Randomly select $\nu_{i,j}^{t}$ and $\tau_{i,j}^{s}$ which satisfies:
        (1) $I(S_{k,\nu}^{\tau}) = 1$ for $\tau = \tau_{i,j}^{s}, \tau_{i,j}^{s} + 1, \ldots, \tau_{i,j}^{s} + p_{i,j} - 1$,
        (2) $\tau_{i,j}^{s} \geq \tau_{i,j'}^{s} + p_{i,j'} - 1$ for $A_{i,j'}^{t} \in PA_{i,j}^{t}$,
        (3) $\tau_{i,j}^{s}$ is one of minimums among $\tau$ satisfying both (1) and (2), and update $I(S_{k,\nu}^{\tau}) = 0$ for $\nu = \nu_{i,j}^{t}$ and
            $\tau = \tau_{i,j}^{s}, \tau_{i,j}^{s} + 1, \ldots, \tau_{i,j}^{s} + p_{k,\nu} - 1$.
    Step 5. $\chi_{i,j}^{t}[1][h] = (\nu_{i,j}^{t}, (\tau_{i,j}^{s}, \tau_{i,j}^{s} + 1, \ldots, \tau_{i,j}^{s} + p_{k,\nu} - 1))$.
    Step 6. If $i = n_t$ and $j = m_i^t$, terminate this algorithm. If $j = m_i^t$ and $i \neq n_t$, increase $i$ by 1
        and go to Step 2. If $j \neq m_i^t$, increase $j$ by 1 and go to step 3.
output: $\chi^t[1][h]$.
```

Algorithm 1: Generation of initial solutions of $\chi^t[1][h]$.

$$s^t[g][h] = \frac{1}{\mu\left(\gamma_1\left(\chi^t[g][h]\right), \gamma_2\left(\chi^t[g][h]\right), \gamma_3\left(\chi^t[g][h]\right), \gamma_4\left(\chi^t[g][h]\right)\right)}, \tag{16}$$

where $\gamma_i(\chi^t[g][h])$ $(i = 1, 2, 3, 4)$ is the ranking of the value, Z_i, of the objective function with regard to $\chi^t[g][h]$ among chromosomes in the same generation and $\mu(\cdot)$ denotes a mean function. Table 3 demonstrates how to calculate the fitness value by means of Pareto ranking.

For example, because the Z_1 values of $\chi^t[g][i]$ $(i = 1, 2, 3, 4)$ are 10, 12, 5, and 8, their ranks by $\text{rank}_1(\chi^t[g][i])$ $(i = 1, 2, 3, 4)$ are 3, 4, 1, and 2, respectively (see the first column of Table 3, circled by a bold blue rectangle). The rankings of Z_1, Z_2, Z_3, and Z_4 with regard to $\chi^t[g][1]$ are 3, 1, 3, and 2, respectively, and the fitness value is calculated as $(1/\mu(3, 1, 3, 2)) = 0.444$ (see the first row of Table 3, delineated by a dashed red rectangle). Note that the solution with a smaller objective function value takes a higher ranking because they are to be minimized.

3.4. Genetic Operators. We adopt a uniform crossover operator that selects one of the crossover lines randomly and assigns chromosomes of parents to the children based on those lines [29]. In this paper, we avoided generating infeasible solutions as presented in Algorithm 2.

4. Experiment

In this section, we conduct experiments to verify that the proposed approach is practical under the experiment environment:

OS: Microsoft Windows 10 Home (×64)

CPU: Intel® Core™ i7-6700 CPU

RAM: 16.0 GB

Language: Python 3.6.9

First, a small toy problem is introduced to find all possible solutions and calculate their fitness values by means of a Pareto ranking method in Section 4.1, followed by checking the ranks of the solutions obtained by the proposed approach. Second, we apply the proposed approach to solve a more realistic, large-scale scheduling problem and examine the search time to establish a schedule in Section 4.2. The first experiment shows that our approach can find a schedule that is effective and sufficiently close to the optimal schedule. The second experiment is to verify the given schedule as efficient and appropriate for a large-scale practical problem in real time.

4.1. Effectiveness Verification. Figure 3 shows the task type introduced to compare the proposed method with an exhaustive search method. “S” and “H” are activities that require software and hardware services, respectively.

We assume three tasks, three software services, and three hardware services in this problem. Information about the tasks, such as maximum allowed due date (D_i^t), cost (C_i^t), minimum quality (Q_i^t), and reliability (R_i^t), is summarized in Table 4.

Table 5 supplies information on services such as cost $(c_{k,\nu})$, quality $(q_{k,\nu})$, reliability $(r_{k,\nu})$, and processing time $(p_{k,\nu})$.

Table 3: An example of the Pareto ranking method.

	Objective function value () = ranking of solutions on each objective function					Fitness value
	Z_1	Z_2	Z_3	Z_4		
$\chi^t[g][1]$	10(3)	6(1)	8(3)	16(2)		0.444
$\chi^t[g][2]$	12(4)	10(3)	6(2)	20(4)	⇒	0.308
$\chi^t[g][3]$	5(1)	8(2)	4(1)	15(3)		0.571
$\chi^t[g][4]$	8(2)	15(4)	10(4)	10(1)		0.364

```
Input: parent chromosomes $\chi^t[g][h_1]$ and $\chi^t[g][h_2]$ $(h_1 \neq h_2)$
Procedure:
    Step 1. Initialize $i$ as 1.
    Step 2. Initialize $j$ as 1.
    Step 3. Randomly select an element from $\{h_1, h_2\}$, and let $h$ be the chosen and $\tilde{h}$ be the other one.
    Step 4. $\chi^t_{i,j}[g+1][h_3] = \chi^t_{i,j}[g+1][h]$ and $\chi^t_{i,j}[g+1][h_4] = \chi^t_{i,j}[g+1][\tilde{h}]$.
    Step 5. If $\tau^{t,e}_{i,j}[g][h] \leq \min_{\bar{j}}(\tau^{t,s}_{i,\bar{j}}[g][\tilde{h}])$, $\tau^{t,e}_{i,j}[g][\tilde{h}] \leq \min_{\bar{j}}(\tau^{t,s}_{i,\bar{j}}[g][h])$, $\max_{\tilde{j}}(\tau^{t,e}_{i,\tilde{j}}[g][h]) \leq \tau^{t,s}_{i,j}[g][\tilde{h}]$, and
            $\max_{\tilde{j}}(\tau^{t,e}_{i,\tilde{j}}[g][h]) \leq \tau^{t,s}_{i,j}[g][\tilde{h}]$, then exchange $h$ and $\tilde{h}$ with a probability $\lambda$, where $A^t_{i,\tilde{j}} \in PA^t_{i,j}$ and $A^t_{i,\bar{j}} \in FA^t_{i,j}$.
            Otherwise, do nothing.
    Step 6. If $j = m^t_i$ and $i = n_t$, terminate. If $j = m^t_i$ and $i \neq n_t$, increase $i$ by 1 and go to Step 2. If $j \neq m^t_i$, increase $j$ by 1 and go to
            Step 3.
Output: children chromosomes $\chi^t[g+1][h_3]$ and $\chi^t[g+1][h_4]$.
```

Algorithm 2: Procedure of uniform crossover.

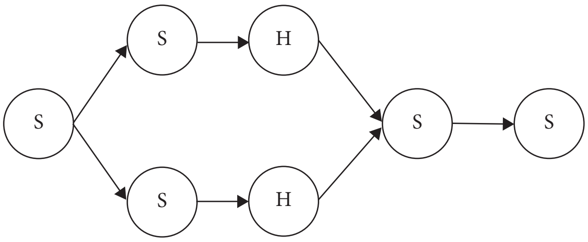


Figure 3: Task type for effectiveness verification.

Table 4: Task information.

Task	D^t_i	C^t_i	Q^t_i	R^t_i
T^1_1	10	160	98.5	0.98
T^1_2	8	170	99.0	0.99
T^2_1	14	150	98.0	0.98

GA parameters in the proposed approach are set as follows: the number of generations is 50; the number of solutions per each generation is 20. Therefore, the GA searches $50 \times 20 = 1000$ solutions. We calculate the ranks of all solutions, including the one obtained by the proposed approach. Any inferior solutions (e.g., those waiting to allocate a service even though it is available, solutions that allocate no service at all, and solutions that allocate a single service to every activity) are ignored when searching all possible solutions. Table 6 summarizes all possible solutions, which are listed in the descending order of fitness values. The values in bold indicate the solution obtained by the proposed GA.

As seen in Table 6, our approach has found the 172th solution among 34,012,224 solutions, which lies in the top 0.00051%. In other words, the proposed algorithm can find a solution which is very close to the optimal. Because the proposed algorithm makes a schedule after a task is uploaded in the task pool, we expect that the number of tasks has little effect on the result. From the experiment result and the real-time property of the algorithm, we can conclude that the proposed algorithm is effective for task scheduling in cloud manufacturing.

4.2. Efficiency Verification. Next, a CM platform is assumed dealing with a more complicated and realistic task, depicted in Figure 4, which was adopted from [30].

We randomly generate six situations where 5, 10, 15, 20, 25, and 30 tasks are uploaded at a day, and we calculate search time to establish the schedules in each situation. We assume that every task is uploaded at t, and D^t_i, C^t_i, Q^t_i, and R^t_i are uniformly distributed in the intervals [30, 60], [400, 450], [95, 100], and [0.95, 1.00], respectively.

A total of 30 software services and 15 hardware services are virtualized and shared in the platform, and every service is idle at first. We assume that $q_{k,v}$, $r_{k,v}$, and $p_{k,v}$ follow uniform distributions in the intervals [95, 100], [0.95, 1.00], and [1, 3], respectively. The costs of hardware and software services are assumed to be different from each other; that is, $c_{k,v}$ of software and hardware service is

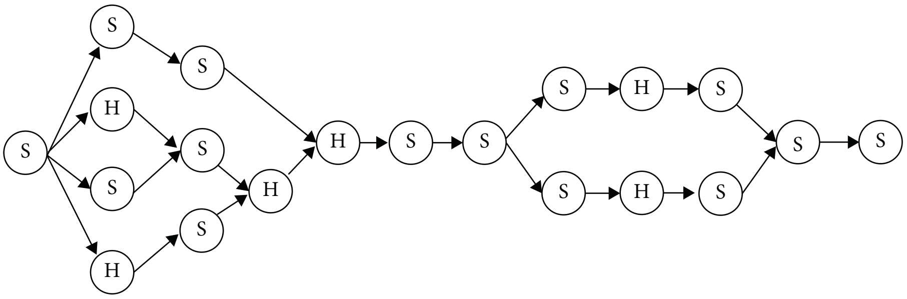


Figure 4: Task type for efficiency verification.

Table 5: Service information.

Software service	$c_{k,v}$	$q_{k,v}$	$r_{k,v}$	$p_{k,v}$
$S_{1,1}$	22	100	1.00	1
$S_{1,2}$	18	99	0.98	2
$S_{1,3}$	14	97	0.97	3
Hardware service				
$S_{2,1}$	40	100	1.00	1
$S_{2,2}$	34	98	0.98	2
$S_{2,3}$	28	97	0.97	3

Table 6: Objective functions, ranks, and fitness values of all possible solutions.

Rank	Value				Rank				Fitness value
	Z_1	Z_2	Z_3	Z_4	Z_1	Z_2	Z_3	Z_4	
1	1	94	0.0000	0.3187	353,454	5,661,611	1	3,899,820	4.034×10^{-7}
2	1	94	0.0000	0.3187	353,454	5,661,611	1	3,899,547	4.034×10^{-7}
3	1	96	0.0000	0.3101	353,454	6,921,456	1	2,932,620	3.918×10^{-7}
4	1	96	0.0000	0.3101	353,454	6,921,456	1	2,934,610	3.917×10^{-7}
5	1	90	0.0000	0.3361	353,454	4,163,149	1	6,170,497	3.742×10^{-7}
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
172	**1**	**96**	**0.1071**	**0.3195**	**272,978**	**6,921,456**	**9,569,507**	**4,248,174**	1.904×10^{-7}
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
34,012,224	7	98	0.1428	0.3727	14,330,889	7,321,564	14,768,747	11,846,972	8.287×10^{-8}

Table 7: Search time according to the numbers of solutions and tasks (unit: seconds).

		The number of solutions to be searched			
		1,000	5,000	10,000	Average scheduling time for each task
The number of tasks to be scheduled	5	140	767	1,551	310
	10	225	1,646	2,050	205
	15	686	1,939	6,442	429
	20	831	4,267	7,964	398
	25	1,068	7,694	10,141	406
	30	1,322	9,012	12,438	415

assumed to follow uniform distributions in [28, 40] and [95, 100], respectively.

Table 7 shows the search times when the numbers of solutions are 1000, 5000, and 10,000 and tasks are 5, 10, 15, 20, 25, and 30. It is obvious that as more solutions are searched it takes longer but is more likely to find the best solution. The rightmost column is the average scheduling time per task when 10,000 solutions are considered.

As seen, the maximum search time is 12,438 seconds (3.455 hours) for 10,000 solutions and 30 tasks. A search time of three and a half hours is a relatively long time, raising the concern that the proposed approach is inadequate to be applied to in real time. However, the average time to schedule a task when we searched 10,000 solutions is 205~420 seconds, approximately 3~7 minutes. In practice, it is not usual for a complicated task to be uploaded every 7 minutes. In addition, the considered task is overly complicated comparing to realistic tasks, implying that the average time will be much lower than the experiment result provided in this study. Thus, once a complete and full schedule has been established, which may take several hours, a couple of tasks can be added to the existing schedule, which can be updated within 10 minutes, to produce an efficient schedule. Therefore, we conclude that the proposed approach is efficient and can be used in practice with a large number of complicated tasks and services. Even though the proposed algorithm is applied to establish schedules under an unreal situation in terms of the number of tasks and services, it can be applied to real tasks and services uploaded in CM such as in [8].

5. Conclusion

CM is a manufacturing model based on the concept of sharing manufacturing resources among manufacturing enterprises to deal with highly customized requests of customers. In CM, each request is regarded a task, and it is assigned to one or more enterprises, called task scheduling. The objective of the task scheduling is to maximize customer satisfaction, and therefore, tardiness, cost, quality, and so forth should be considered. In addition, the task should be assigned as soon as possible for practical usage.

This paper addressed a real-time and multiobjective task-scheduling problem to minimize total tardiness, cost, quality, and reliability penalties in CM. This problem was formalized as a binary integer programming model, and the MOGA was designed to solve the model by focusing on generating feasible solutions. From the experiment, we showed that the schedule based on the proposed approach is near the optimal schedule for small problems. In addition, we verified that the proposed approach establishes effective schedules in a real-time manner for a more realistic large-scale problem.

In future research, we will develop a GA that does not generate infeasible or inferior solutions. This will surely reduce the search time remarkably. In addition, we will conduct practical research to apply our algorithm to the commercialized cloud manufacturing system. Finally, we will improve the proposed multiobjective GA to apply it to train the deep learning model-based CM scheduler.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea Government (MSIT) (2019R1A2C1088255).

References

[1] G. Ahn, Y.-J. Park, and S. Hur, "The dynamic enterprise network composition algorithm for efficient operation in cloud manufacturing," *Sustainability*, vol. 8, no. 12, p. 1239, 2016.

[2] G. Ahn, Y.-J. Park, and S. Hur, "Probabilistic graphical framework for estimating collaboration levels in cloud manufacturing," *Sustainability*, vol. 9, no. 2, p. 277, 2017.

[3] D. Wu, M. J. Greer, D. W. Rosen, and D. Schaefer, "Cloud manufacturing: strategic vision and state-of-the-art," *Journal of Manufacturing Systems*, vol. 32, no. 4, pp. 564–579, 2013.

[4] B. Huang, C. Li, C. Yin, and X. Zhao, "Cloud manufacturing service platform for small-and medium-sized enterprises," *The International Journal of Advanced Manufacturing Technology*, vol. 65, no. 9–12, pp. 1261–1272, 2013.

[5] G. Ahn, Y.-J. Park, and S. Hur, "Performance computation methods for composition of tasks with multiple patterns in cloud manufacturing," *International Journal of Production Research*, vol. 57, no. 2, pp. 517–530, 2019.

[6] L. Zhou, L. Zhang, C. Zhao, Y. Laili, and L. Xu, "Diverse task scheduling for individualized requirements in cloud manufacturing," *Enterprise Information Systems*, vol. 12, no. 3, pp. 1–19, 2018.

[7] Y. Cao, S. Wang, L. Kang, and Y. Gao, "A TQCS-based service selection and scheduling strategy in cloud manufacturing," *The International Journal of Advanced Manufacturing Technology*, vol. 82, no. 1–4, pp. 235–251, 2016.

[8] Y. Liu, X. Xu, L. Zhang, L. Wang, and R. Y. Zhong, "Workload-based multi-task scheduling in cloud manufacturing," *Robotics and Computer-Integrated Manufacturing*, vol. 45, pp. 3–20, 2017.

[9] H. Jiang, J. Yi, S. Chen, and X. Zhu, "A multi-objective algorithm for task scheduling and resource allocation in cloud-based disassembly," *Journal of Manufacturing Systems*, vol. 41, pp. 239–255, 2016.

[10] W. Li, C. Zhu, L. T. Yang, L. Shu, E. C. H. Ngai, and Y. Ma, "Subtask scheduling for distributed robots in cloud manufacturing," *IEEE Systems Journal*, vol. 11, no. 2, pp. 941–950, 2017.

[11] N. D. Hoang, "Image processing-based pitting corrosion detection using metaheuristic optimized multilevel image thresholding and machine-learning approaches," *Mathematical Problems in Engineering*, vol. 2020, Article ID 6765274, 19 pages, 2020.

[12] W. Chen, Z. Li, and J. Guo, "A VNS-EDA algorithm-based feature selection for credit risk classification," *Mathematical Problems in Engineering*, vol. 2020, Article ID 4515480, 14 pages, 2020.

[13] X. Xue, J. Lu, and J. Chen, "Using NSGA-III for optimising biomedical ontology alignment," *CAAI Transactions on Intelligence Technology*, vol. 4, no. 3, pp. 135–141, 2019.

[14] H. S. Pannu, D. Singh, and A. K. Malhi, "Improved particle swarm optimization based adaptive neuro-fuzzy inference system for benzene detection," *CLEAN–Soil, Air, Water*, vol. 46, no. 5, Article ID 1700162, 2018.

[15] L. R. Rodrigues and J. P. P. Gomes, "TLBO with variable weights applied to shop scheduling problems," *CAAI Transactions on Intelligence Technology*, vol. 4, no. 3, pp. 148–158, 2019.

[16] T. Garai and H. Garg, "Multi-objective linear fractional inventory model with possibility and necessity constraints under generalised intuitionistic fuzzy set environment," *CAAI Transactions on Intelligence Technology*, vol. 4, no. 3, pp. 175–181, 2019.

[17] S. Lalwani, H. Sharma, A. Verma, and R. Kumar, "Efficient discrete firefly algorithm for Ctrie based caching of multiple sequence alignment on optimally scheduled parallel

machines," *CAAI Transactions on Intelligence Technology*, vol. 4, no. 2, pp. 92–100, 2019.

[18] S. Jiang, C. Zhang, and S. Chen, "Sequential hybrid particle swarm optimization and gravitational search algorithm with dependent random coefficients," *Mathematical Problems in Engineering*, vol. 2020, Article ID 1957812, 17 pages, 2020.

[19] Y. Peng, K. Lei, X. Yang, and J. Peng, "Improved chaotic quantum-behaved particle swarm optimization algorithm for fuzzy neural network and its application," *Mathematical Problems in Engineering*, vol. 2020, Article ID 9464593, 11 pages, 2020.

[20] R. L. Kadri and F. F. Boctor, "An efficient genetic algorithm to solve the resource-constrained project scheduling problem with transfer times: the single mode case," *European Journal of Operational Research*, vol. 265, no. 2, pp. 454–462, 2018.

[21] F. Lu, H. Bi, M. Huang, and S. Duan, "Simulated annealing genetic algorithm based schedule risk management of IT outsourcing project," *Mathematical Problems in Engineering*, vol. 2017, Article ID 6916575, 17 pages, 2017.

[22] P.-H. Lu, M.-C. Wu, H. Tan, Y.-H. Peng, and C.-F. Chen, "A genetic algorithm embedded with a concise chromosome representation for distributed and flexible job-shop scheduling problems," *Journal of Intelligent Manufacturing*, vol. 29, no. 1, pp. 19–34, 2018.

[23] Y.-B. Woo, S. Jung, and B. S. Kim, "A rule-based genetic algorithm with an improvement heuristic for unrelated parallel machine scheduling problem with time-dependent deterioration and multiple rate-modifying activities," *Computers & Industrial Engineering*, vol. 109, pp. 179–190, 2017.

[24] C. Chamnanlor, K. Sethanan, C.-F. Chien, and M. Gen, "Re-entrant flow shop scheduling problem with time windows using hybrid genetic algorithm based on auto-tuning strategy," *International Journal of Production Research*, vol. 52, no. 9, pp. 2612–2629, 2014.

[25] R. C. Chen, J. Chen, T. S. Chen, C. C. Huang, and L. C. Chen, "Synergy of genetic algorithm with extensive neighborhood search for the permutation flowshop scheduling problem," *Mathematical Problems in Engineering*, vol. 2017, Article ID 3630869, 9 pages, 2017.

[26] L. Tang, L. Zheng, H. Cao, and N. Huang, "An improved multi-objective genetic algorithm for heterogeneous coverage RFID network planning," *International Journal of Production Research*, vol. 54, no. 8, pp. 2227–2240, 2016.

[27] Y. Zhang, X. Hu, and C. Wu, "A modified multi-objective genetic algorithm for two-sided assembly line re-balancing problem of a shovel loader," *International Journal of Production Research*, vol. 56, no. 9, pp. 3043–3063, 2017.

[28] S. Kukkonen and J. Lampinen, "Ranking-dominance and many-objective optimization," in *Proceedings of IEEE Congress on Evolutionary Computation*, pp. 3983–3990, Singapore, September 2007.

[29] G. Syswerda, "Uniform crossover in genetic algorithms," in *Proceedings of the 3rd international Conference on Genetic Algorithms*, pp. 2–9, San Francisco, CA, USA, June 1989.

[30] F. Tao, Y. LaiLi, L. Xu, and L. Zhang, "FC-PACO-RM: a parallel method for service composition optimal-selection in cloud manufacturing system," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 4, pp. 2023–2033, 2013.

Hindawi
Mathematical Problems in Engineering
Volume 2021, Article ID 6903925, 8 pages
https://doi.org/10.1155/2021/6903925

Research Article

Comparative Regression Analysis for Estimating Resonant Frequency of C-Like Patch Antennas

Umut Özkaya,[1] **Enes Yiğit**,[2] **Levent Seyfi**,[1] **Şaban Öztürk**,[3] **and Dilbag Singh**[4]

[1] *Department of Electrical and Electronics Engineering, Konya Technical University, Konya, Turkey*
[2] *Department of Electrical and Electronics Engineering, Karamanoğlu Mehmetbey University, Karaman, Turkey*
[3] *Department of Electrical and Electronics Engineering, Amasya University, Amasya, Turkey*
[4] *Computer Science Engineering, School of Engineering and Applied Sciences, Bennett University, Greater Noida 201310, India*

Correspondence should be addressed to Umut Özkaya; uozkaya@ktun.edu.tr

Received 13 May 2020; Revised 31 July 2020; Accepted 30 September 2020; Published 8 February 2021

Academic Editor: Thomas Hanne

This study provides a comparative analysis of regression techniques to estimate the operating frequency of the C-like microstrip antenna. The performance of well-known regression techniques such as linear regression (LR), regression tree (RT), support vector regression (SVR), Gaussian regression (GR), and artificial neural network (ANN) is tested. For this purpose, 160 C-like microstrip antennas are simulated, of which 145 are used for training of regression techniques and 15 for testing. From the evaluated results, it is found that the pure quadratic Gaussian regression (PQGR) technique has the lowest error rates with 0.0109 mean absolute error (MAE), 0.0087 median error (ME), 0.0002 mean squared error (MSE), 0.0156 root mean squared error (RMSE), and 0.5981 average percentage error (APE). As can be seen in the comparative analysis, the PQGR method outperforms other regression methods on simulation and measurement data. Experimental analysis shows that the resonant frequency of the C-like patch antennas can be calculated very close to measurements.

1. Introduction

Microstrip patch antennas constitute a wide field of research in the literature due to their advantages such as being light, small, easy in production, low cost, and easy integration into a system. Thanks to these advantages, it has been used with different designs in fields such as wireless communication systems, remote sensing systems, and medical and radar applications [1]. The design of microstrip patch antennas is achieved by coating the conductors on the upper and lower surfaces of thin dielectric material. One surface acts as a ground, while the other emits electromagnetic radiation. Microstrip patch antennas have disadvantages such as low gain of receiver and narrow bandwidth. In the literature, patch antennas with square, rectangular, triangular, and circle geometry are so common and easy to analyze theoretically [2–4]. In these types of antennas, different gains can be obtained by using different patch geometry and feeding techniques. Changes in antenna geometry can increase the effective electric field. Patch sizes and ground plane geometry can affect resonant frequency. Besides, the feed point location and feeding technique have high effects on the resonant frequency. The resonant frequency of the antenna is inversely proportional to the patch dimensions. In the literature, there can be found various shaped microstrip antennas such as C, E, H, and L shapes [5–9]. Such antennas are generally symmetrically loaded concerning the edge of the patch, while the slot antennas consist of asymmetric notches on one side of the patch. While symmetric antennas are easy to analyze, asymmetric slot antennas cannot be expressed in basic mathematical equations [10–18]. Therefore, analysis of the C-like slot antenna is performed by choosing different feed types, feed point location, and dielectric materials in this study.

One of the most important parameters for the microstrip patch antenna is the resonant frequency. The fact that antenna has different structure geometry rather than traditional geometry is a factor that makes the analysis difficult. On the other hand, microstrip antennas can be accurately simulated via software technology. The developed method of moments (MoM) uses the Maxwell equations and performs the necessary analyses. Since the cost of purchasing these methods in the software packages is very high, the designers tried to overcome this problem with artificial intelligence methods. Thanks to the intuitive and supervised methods that artificial intelligence offers to the users, they were able to approach the solution of the problems faster and more efficiently. These methods include artificial neural networks (ANNs) [19], regression analysis methods (RAMs) [20], support vector regression (SVR) [21], and regression trees (RT) [22]. In the literature, optimization algorithms and ANN are often preferred to estimate the resonant frequency of microstrip patch antennas. SVR [23], adaptive neurofuzzy interaction system (ANFIS) [24], and artificial bee colony (ABC) optimization are also widely used.

In this study, a detailed comparative analysis is performed to estimate the operating frequency of the C-like microstrip patch antenna. In this context, linear and Gaussian regression analyses, SVR, RT, and ANN are used. The pure quadratic Gaussian regression (PQGR) technique has achieved the highest performance. In PQGR analysis, dielectric material height (h), C-like microstrip patch antenna dimensions (L, W, 1, w, d), and relative dielectric constant (ε_r) are given as input, and operating frequency (f_t) is estimated as output. 160 C-like microstrip antennas are simulated in computer-based software combined with computational electromagnetic (CEM) software [25] for training and testing. While the simulated 145 antennas were used for training, the remaining 15 antennas are used for the test process. For comparison with the literature, the test antennas are selected as in [26]. To test the performance of the PQGR technique, a 6-fold cross-validation technique is applied. As a result of PQGR analysis, resonant frequency is converged with the best values of 0.0109 MAE, 0.0087 ME, 0.0002 MSE, 0.0156 RMSE, and 0.5981 APE. The main contributions of the proposed PQGR method are as follows:

(i) C-shaped microstrip antenna data are analyzed with regression methods

(ii) Proposed PQGR model has higher performance than other regression methods

(iii) As seen in comparative results, the proposed PQGR models have low error metrics

(iv) In the testing phase, simulation and real data were evaluated in the proposed method

The outline of this study is as follows. In the next section, the design parameters of the antennas and the regression analysis methods are presented. In Section 3, comparative analyses are presented. In the last section, the study is summarized and future directions are mentioned.

2. Material and Methods

2.1. Material. As seen in Figure 1, C-like microstrip antenna's length and width are indicated as L and W. The thickness of dielectric material is represented by h, and also its relative permeability is ε_r. In the x-y coordinate system, the coaxial feed point is defined as (x_0, y_0). There is an $l \times w$ slot in the rectangular patch. d indicates the upper distance of the slot. The designed C-like antenna data are simulated with seven variables and form the input of the regression analysis. The resonant frequency is available as output.

The microstrip patch antennas are designed for use in ultra-high frequency (UHF) band applications. It should have a resonant frequency of 1.15–3.335 GHz. 160 different antennas were simulated with CEM software HyperLynx® 3D EM [27]. To provide a homogeneous data distribution, five different group values are created in the antenna dimensions. As given in Table 1, each group has 32 antenna data and outer dimensions of groups were 30, 20; 35, 25; 40, 30; 45, 35; and 50, 40 including different parameters of l, w, d, h, and relative permittivity ε_r. In order to compare with the literature [26], the coaxial feed point was determined as $x_0 = 5$ mm and $y_0 = 5$ mm. 1-volt wave source was used for power supply. A total of 160 different antenna parameters were generated between 1 and 5 GHz in CEM simulation.

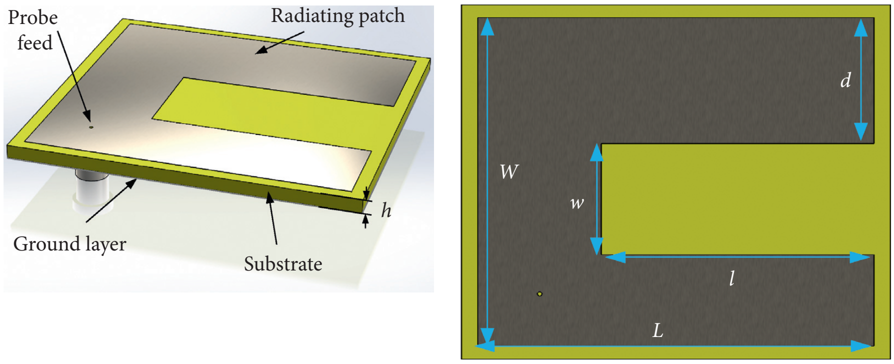
2.2. Techniques

2.2.1. Linear Regression. Dependent variables can be expressed by many independent variables. Multiple linear regression analysis is the method used to explain the relationship between two and more independent variables affecting a variable in a linear model and to determine the effects of these independent variables. Multiple linear regression equation:

$$Y = a + b_1X_1 + b_2X_2 + \ldots b_mX_m + \varepsilon, \tag{1}$$

where Y is a dependent variable, X_1, X_2, ..., X_m are independent variables, b_1, b_2, $b_{3,}$ ..., b_m are regression coefficients, and finally ε represents the error term. a is the regression constant. b values are also called partial regression or partial slope coefficients. In multiple linear regression, there is no correlation between the number of independent variables and predictive accuracy [28]. Therefore, it is vital to determine the number of independent variables in terms of the reliability of estimation. In this study, linear, interaction linear, and stepwise linear regressions were used.

2.2.2. Regression Tree. Regression trees determine output value by integrating the regression method at the end of the model unlike the classification problem. The prediction results of regression trees are lower than those of other regression techniques. Regression trees can perform an efficient calculation by creating segmented linear models. First, a fixed tree model is created and the linear regression analysis is applied to the data in each node [29]. Residuals are calculated by adapting the nodes to the regression model. Since the complexity of the model is shared between the tree



(a) (b)

Figure 1: C-like microstrip patch antenna geometry. (a) Isometric view. (b) Front view.

Table 1: Simulated slot antenna parameters.

Patch dimensions (mm)					h (mm)	ε_r
L	W	l	w	d		
30	20	10; 20	5; 10	3; 6	1.6; 2.5	2.33; 4.4
35	25	15; 25	7; 12	8; 10		
40	30	15; 30	7.5; 15	5; 10		
45	35	25; 30	5; 10	20; 25		
50	40	20; 40	10; 20	7; 14		

structure and the nodes, the complexity of the tree structure is reduced. For this reason, the complexity of the tree structure should be taken into consideration before the design of the model. As the complexity of the model increases, it requires a lot of data for training. Therefore, the size of the tree and the number of nodes should be determined according to the number of data. In this study, fine, medium, and coarse regression tree methods were used.

2.2.3. Support Vector Regression. In regression analysis, SVR is the most widely used type of support vector machine (SVM). The basic idea in SVM includes the determination of the regression function [30]. There are some differences in SVR from standard SVM operations. If the training data are defined as $\{(x_1, y_1), (x_2, y_2), \ldots, (x_l, y_l)\}$, some deviations may occur in objective function $f(x)$. To define the linear objective function,

$$f(x) = (w, x) + b. \tag{2}$$

The solution of the optimization problem in equation (3) can be approached by reducing Euclidean norm $\|w\|_2$.

$$\text{Minimize } \frac{1}{2}\|\omega\|^2 \text{ subject to } \left\{\begin{array}{c} y_i - (\omega, x_i) - b \le \varepsilon \\ (\omega, x_i) + b - y_i \le \varepsilon \end{array}\right\}. \tag{3}$$

For the objective function in equation (3), it is possible to converge at all values (x_i, y_i) with a certain sensitivity (ε). To deal with some constraints of the optimization problem, the objective function needs to be modified with some slack variables.

$$\begin{gathered}\text{Minimize } \frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{l}(\xi_i + \xi_i^*) \\ \text{subject to } \left\{\begin{array}{c} y_i - (\omega, x_i) - b \le \varepsilon + \xi_i \\ (\omega, x_i) + b - y_i \le \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0 \end{array}\right\}.\end{gathered} \tag{4}$$

In this study, linear, quadratic, cubic, fine, medium, and coarse SVR methods were used to analyze these antenna data.

2.2.4. Gaussian Regression. In the Gaussian process, finite subsets are created with multivariate Gaussian distribution with many variables. N number of observations is $y = \{y_1, \ldots, y_N\}$, which can be defined as Gaussian distribution. In general, the mean Gaussian operation is assumed to be 0 in each observation. Covariance function is necessary to establish a relationship between observations [31]. The covariance function for the quadratic exponential function is defined as follows:

$$k(x, x^{\prime}) = \sigma_f^2 \exp\left[\frac{-(x - x^{\prime})^2}{2l^2}\right]. \tag{5}$$

The maximum covariance value that can be obtained as σ^2_f. When x approaches $x^{\prime}$, covariance gets the maximum value. This means that there is a close relationship between $f(x)$ and $f(x^{\prime})$. The covariance value decreases if x value moves away from

x^t. In this study, rational quadratic, squared exponential, Matern 5/2, and exponential Gaussian regression methods were preferred to analyze these antenna data.

2.2.5. Artificial Neural Network. Artificial neural network (ANN) is one of the artificial intelligence techniques. It was developed by imitating the stimulation and information received from organs to the brain via neurons. In these days, it is common to be used in almost all computational sciences. ANNs can also be thought of as a black box that processes input and generates outputs. This system processes information in parallel and learns the principle that connection coefficients between neurons are updated by minimizing the error.

An ANN model includes input, hidden, and output layers. Each layer has neurons that are connected utilizing their weight coefficients. The input and output layer can contain many neurons that are equal to the number of input and output. However, the number of neurons in the hidden layer depends entirely on the input. $(n+1)/2$ or $2n+1$ neurons can be used in the hidden layer, where n is the number of inputs. A different number of neurons may be used depending on the nature of the problem. The less number of neurons will reduce the learning ability of the system [23].

$$Y_i = f\left(\sum_{i=1}^{n} X_h V_{ih}\right), \tag{6}$$

$$Z_j = f\left(\sum_{i=1}^{p} Y_i V_{ji}\right), \tag{7}$$

where inputs are defined as X, the output of hidden layers is Y, and Z is output. V and W, respectively, indicate weights between the input-hidden layer and hidden-output layer in equations (6) and (7) [32, 33].

Learnable parameters in ANN architecture are updated either in the feed-forward or backpropagation phases. In the feed-forward, existing weight coefficients are obtained from neurons and ultimately the outputs of the system. The total error is calculated between predictions and the actual values. The error value is propagated as backward and then weight coefficients in the connections are updated. This process is iterated either for a specific number or until a given error threshold is reached.

2.2.6. Metrics. Some metrics are needed to evaluate the obtained results. Four metrics were used in this study. These are MAE in equation (8), ME in equation (9), MSE in equation (10), RMSE in equation (11), and APE in equation (12):

$$\text{MAE} = \text{mean}_{i=1,n}|e|_i, \tag{8}$$

$$\text{ME} = \text{median}_{i=1,n}|e_i|, \tag{9}$$

$$\text{MSE} = \text{median}_{i=1,n}\left|e_i^2\right|, \tag{10}$$

$$\text{RMSE} = \sqrt{\text{median}_{i=1,n}\left|e_i^2\right|}, \tag{11}$$

$$\text{APE} = \frac{\sum_{i=1}^{N}\left|f_{\text{SIMi}} - f_{\text{COMi}}\right|}{N}. \tag{12}$$

2.2.7. Training and Validation. 160 CEM data were divided into 145 and 15 as training and validation, respectively. All training operations were performed on Intel Core i7-7700 HQ 2.8 GHz processor with 16 GB RAM in Matlab software. The metrics in this study were evaluated for accuracy of obtained resonant frequencies by proposed methods. Figure 2 shows how to generate antenna data from the CEM program.

The K-fold cross-validation method was used to prove the validity of the proposed methods. The cross-validation process divides the existing data set into equal subsets for training and validation processes. This process is divided into K subsets and repeated K times. Each subset must be used only once for validation. In this way, all data are used for both training and validation. Figure 3 illustrates the 6-fold cross-validation process. All data (# 160) were divided into 6 subsets (5×# 29 and # 15). For each iteration, 145 CEM data (CEMs) were used for training and 15 CEMs were used for validation.

2.2.8. Proposed Method. Some statistical metrics were used between simulation and calculated resonant frequency values to compare the performance of proposed regression analysis methods. These metrics are mean absolute error (MAE), median error (ME), mean squared error (RMSE), and root mean squared error (RMSE). In Figure 4, the proposed method is tried to be schematized. f_{rSIM} and f_{rCOM} represent simulated and computed resonant frequencies, respectively. In this study, the dataset was divided into six subsample sets within the scope of 6 cross-validations. Subsample 6 was reserved for testing. The proposed methods in the comparative analysis are LR, interaction LR, stepwise LR, fine RT, medium RT, coarse RT, linear SVR, quadratic SVR, cubic SVR, fine Gaussian SVR, medium Gaussian SVR, coarse Gaussian SVR, rational quadratic GR, squared exponential GR, Matern 5/2 GR, exponential GR, PQGR, and ANN.

3. Results and Discussion

The obtained results by the proposed method are given in Table 2. Primarily, subsample 6 without measurement data was used to test algorithms. It includes 15 CEMs. When metric performances were evaluated, the highest error in MAE belonged to coarse TR with 0.3521. The lowest error in MAE was obtained with 0.0109 in PQGR. Considering ME, LR showed the worst performance with a value of 0.2993. The lowest value of 0.0087 ME was obtained with PQGR. In the context of MSE, the LR method had the worst performance with 0.1274 error value. PQGR was the most successful one with MSE value of 0.0002. As with other error

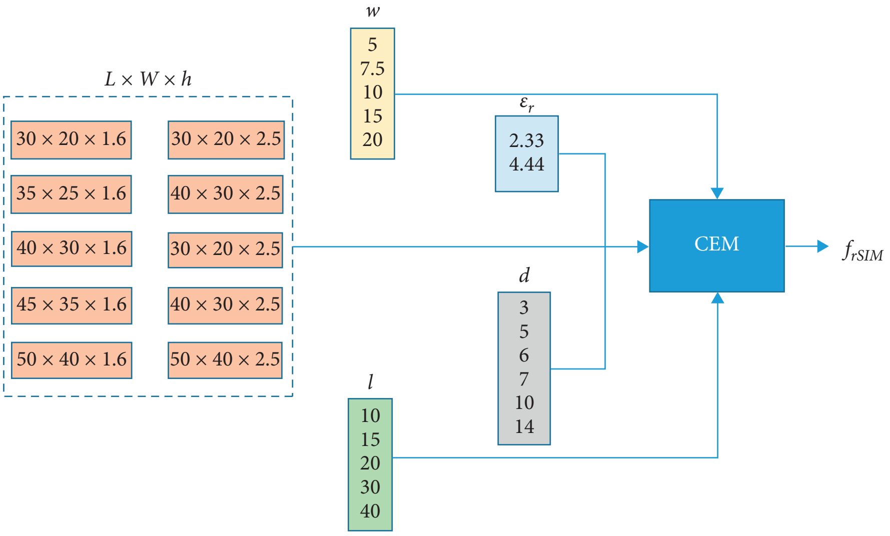
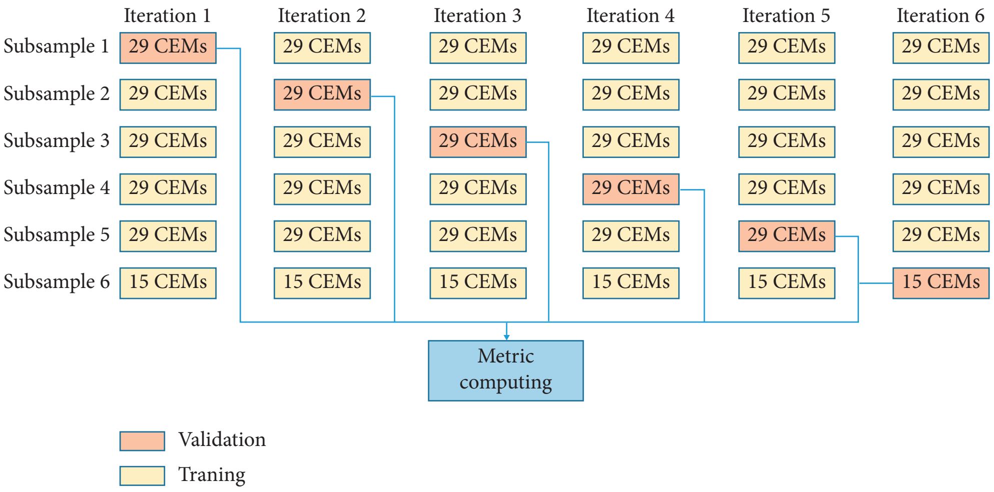


Figure 2: Data generation in CEM.



	Iteration 1	Iteration 2	Iteration 3	Iteration 4	Iteration 5	Iteration 6
Subsample 1	29 CEMs	29 CEMs	29 CEMs	29 CEMs	29 CEMs	29 CEMs
Subsample 2	29 CEMs	29 CEMs	29 CEMs	29 CEMs	29 CEMs	29 CEMs
Subsample 3	29 CEMs	29 CEMs	29 CEMs	29 CEMs	29 CEMs	29 CEMs
Subsample 4	29 CEMs	29 CEMs	29 CEMs	29 CEMs	29 CEMs	29 CEMs
Subsample 5	29 CEMs	29 CEMs	29 CEMs	29 CEMs	29 CEMs	29 CEMs
Subsample 6	15 CEMs	15 CEMs	15 CEMs	15 CEMs	15 CEMs	15 CEMs

Figure 3: 6-fold cross-validation process.

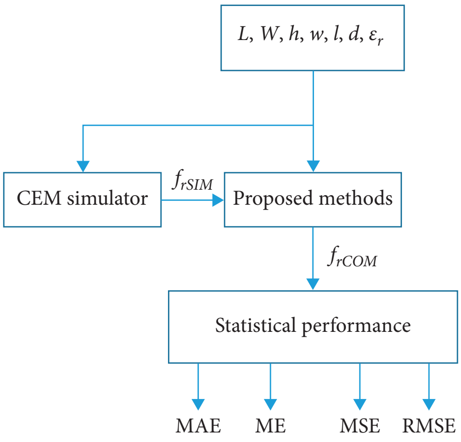


Figure 4: Proposed method.

metrics, LR had the lowest performance with RMSE of 0.3569. PQGR was the best convergence of resonant frequency with 0.0156 RMSE value.

The PQGR method was evaluated within the scope of 6-fold cross-validation as shown in Table 3. The dataset was divided into an equal number of subsamples 1–6. The highest error values were obtained as 0.0537 MAE, 0.0239 ME, 0.0044 MSE, and 0.0662 RMSE. The highest performance is 0.0109 MAE, 0.0087 ME, 0.0002 MSE, and 0.0156 RMSE for the subsample 6 dataset. The mean error values for all subsamples were 0.0292 MAE, 0.0149 ME, 0.0011 MSE, and 0.0278 RMSE. Scatter diagrams about test data are shown in Figure 5.

The trained PQGR model was tested in the subsample with six datasets. This dataset was also evaluated in the concept of MAE, ME, MSE, and RMSE. In this way, the performance of the PQGR model can be verified and

Table 2: Metric performance of proposed methods.

Proposed methods	MAE	ME	MSE	RMSE
LR	0.3246	0.2993	0.1274	0.3569
Interaction LR	0.1960	0.1768	0.1237	0.3517
Stepwise LR	0.1708	0.1680	0.0371	0.1925
Fine TR	0.0729	0.0457	0.0050	0.0705
Medium TR	0.1935	0.1633	0.0361	0.1900
Coarse TR	0.3521	0.2807	0.0494	0.2223
Linear SVR	0.3181	0.2588	0.0446	0.2112
Quadratic SVR	0.1533	0.0866	0.0122	0.1105
Cubic SVR	0.1455	0.1374	0.0289	0.1700
Fine Gaussian SVR	0.3371	0.2854	0.0038	0.0619
Medium Gaussian SVR	0.1896	0.1478	0.0207	0.1437
Coarse Gaussian SVR	0.2990	0.2545	0.0647	0.2545
Rational quadratic GR	0.0801	0.0669	0.0014	0.0368
Squared exponential GR	0.0801	0.0669	0.0014	0.0368
Matern 5/2 GR	0.0828	0.0709	0.0050	0.0709
Exponential GR	0.1250	0.1341	0.0279	0.1670
PQGR	0.0109	0.0087	0.0002	0.0156
ANN	0.3228	0.2369	0.0301	0.1735

Table 3: 6-fold cross-validation for PQGR.

Sample types	MAE	ME	MSE	RMSE
Subsample 1	0.0117	0.0094	0.0003	0.0170
Subsample 2	0.0537	0.0239	0.0006	0.0240
Subsample 3	0.0183	0.0128	0.0044	0.0662
Subsample 4	0.0429	0.0192	0.0003	0.0187
Subsample 5	0.0375	0.0152	0.0006	0.0253
Subsample 6	0.0109	0.0087	0.0002	0.0156
Average	0.0292	0.0149	0.0011	0.0278

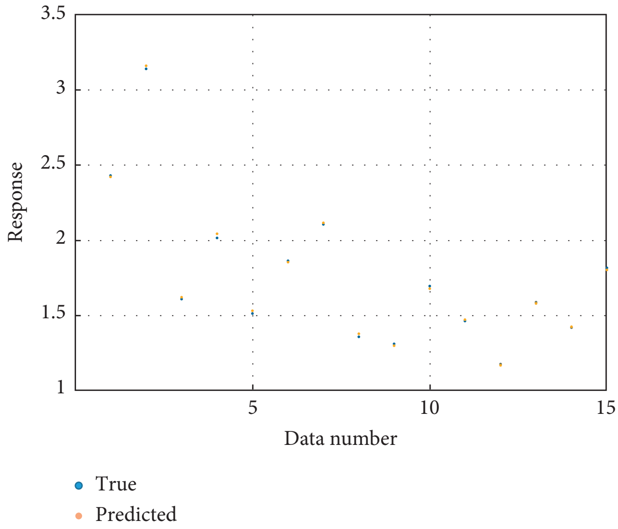


Figure 5: Scatter diagrams of simulated and computed resonant frequency values for test data (subsample 6).

compared with the literature. Comparative simulation and computed resonant frequency values are given in Table 4. In this table, PQGR, fine TR, squared exponential GR, exponential GR, and KNN [26] results are given. Table 4 includes percentage errors for each test data.

Table 5 shows average percentage errors (APEs) of the four best-proposed algorithms. The APE values of PQGR, fine TR, squared exponential GR, and exponential GR are, respectively, 0.5981, 4.1606, 5.3533, and 7.3456. The obtained APE by PQGR is well below the obtained APE by KNN [26]

Table 4: Comparative simulation and computed results.

Sample number	Patch dimension (mm)							Sim.	Mea.	PQGR	Fine TR	Squared exponential GR	Exponential GR	KNN [26]
	L	W	l	w	d	h	ε_r	f_{rSIM}	f_{rMEA}					
1	30	20	10	5	3	2.5	4.4	2.426	—	2.4202	2.3821	2.4588	2.3785	2.4100
2	30	20	10	10	6	2.5	2.33	3.140	—	3.1575	3.1785	3.1402	2.9439	3.1500
3	30	20	20	5	3	1.6	4.4	1.611	—	1.6200	1.5579	1.5293	1.7040	1.6300
4	30	20	20	5	6	1.6	2.33	2.015	—	2.0416	2.0584	2.0345	2.1477	2.0500
5	30	20	20	10	6	1.6	4.4	1.512	—	1.5276	1.5579	1.5384	1.6463	1.5400
6	40	30	15	7.5	5	1.6	4.4	1.860	—	1.8539	1.6208	1.8302	1.8793	1.8700
7	40	30	15	15	5	2.5	2.33	2.107	—	2.1134	2.2321	2.1814	2.0865	2.1000
8	40	30	30	7.5	5	1.6	2.33	1.357	—	1.3736	1.3201	1.4843	1.5863	1.3800
9	40	30	30	7.5	10	2.5	2.33	1.311	—	1.2962	1.3201	1.3085	1.5443	1.2800
10	40	30	30	15	5	2.5	4.4	1.695	—	1.6787	1.7317	1.6202	1.4781	1.6700
11	50	40	20	10	7	1.6	4.4	1.466	—	1.4699	1.2287	1.2366	1.4202	1.4900
12	50	40	20	10	14	2.5	4.4	1.174	—	1.1689	1.2287	1.2406	1.3332	1.1600
13	50	40	20	20	14	1.6	2.33	1.585	—	1.5821	1.5906	1.7255	1.6204	1.6000
14	50	40	40	10	7	2.5	4.4	1.421	—	1.4230	1.3670	1.6788	1.5644	1.4100
15	50	40	40	20	7	2.5	2.33	1.814	—	1.7984	1.8845	1.7772	1.6470	1.7900
16	30	20	10	5	3	3	4	2.426	2.430	2.429	2.3821	2.4654	2.4336	2.4090

Table 5: Percentage errors.

Sample number	Percentage errors (%)				
	PQGR	Fine TR	Squared exponential GR	Exponential GR	KNN [26]
1	0.24	1.81	1.35	1.96	0.65
2	0.55	1.22	0.002	6.25	0.31
3	0.54	3.31	5.09	5.76	1.17
4	1.31	2.14	0.95	6.57	1.73
5	1.02	3.02	1.73	8.87	1.85
6	0.32	12.85	1.59	1.05	0.53
7	0.32	5.95	3.54	0.96	0.33
8	1.24	2.70	9.41	16.92	1.69
9	1.13	0.70	0.19	17.80	2.36
10	0.97	2.16	4.42	12.80	1.47
11	0.23	16.22	15.67	3.16	1.63
12	0.41	4.68	5.70	13.59	1.19
13	0.21	0.33	8.84	2.21	0.94
14	0.16	3.78	18.17	10.11	0.77
15	0.86	3.89	2.03	9.21	1.32
16	0.06	1.81	1.62	0.31	0.70
Average	0.5981	4.1606	5.3533	7.3456	1.1702

in the literature. Therefore, it is seen that the PQGR method is more successful than other methods in calculating the resonant frequency when considering the 6-fold cross-validation results and the results of MAE, ME, MSE, RMSE, and APE in the subsample 6 test.

4. Conclusion

In this study, the resonant frequency for the C-like microstrip patch antenna was tried to be computed in a comparative analysis. Microstrip patch antennas were analyzed by the CEM simulation program, and the dataset was created. There are a total of 160 simulation data with different geometric and electrical properties. The PQGR model was trained with # 145 training data and tested with subsample 6. The regression function for PQGR analysis was pure quadratic. It can analyze challenging datasets accurately. A single measurement data was also used in the test process. MAE, ME, MSE, RMSE, and APE were used to determine the performance of the PQGR model. The resonant frequency for subsample 6 was computed with 0.0109 MAE, 0.0087 ME, 0.0002 MSE, 0.0156 RMSE, and 0.5981 APE. Consequently, the obtained results in PQGR for both training and test data are very close to the simulation results compared to other regression models. Therefore, the PQGR method, which is proposed as an alternative to the highly costly simulation and measurement procedures, can calculate the resonant frequency with high accuracy.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] G. Kumar and K. P. Ray, *Broadband Microstrip Antennas*, Artech House, Massachusetts, MA, USA, 2003.

[2] K.-L. Wong, *Compact and Broadband Microstrip Antennas*, John Wiley and Sons, New York, NY, USA, 2002.

[3] G. Dubost and A. Rabbaa, "Analysis of a slot microstrip antenna," *IEEE Transactions on Antennas and Propagation*, vol. 34, no. 2, pp. 155–163, 1986.

[4] P. Shan-Cheng and W. Kin-Lu, "Dual-frequency triangular microstrip antenna with a shorting pin," *IEEE Trans Antennas Propag*, vol. 45, pp. 1889–1891, 1997.

[5] A. A. Deshmukh and G. Kumar, "Formulation of resonant frequency for compact rectangular microstrip antennas," *Microwave and Optical Technology Letters*, vol. 49, no. 2, pp. 498–501, 2007.

[6] Z. N. Chen, "Radiation pattern of a probe-fed L-shaped plate antenna," *Microwave and Optical Technology Letters*, vol. 27, no. 6, pp. 410–413, 2000.

[7] A. Akdagli, A. Kayabasi, and I. Develi, "Computing resonant frequency of C-shaped compact microstrip antennas by using ANFIS," *International Journal of Electronics*, vol. 102, no. 3, pp. 407–417, 2015.

[8] N. Ojaroudi, M. Ojaroudi, and N. Ghadimi, "UWB omnidirectional square monopole antenna for use in circular cylindrical microwave imaging systems," *IEEE Antennas and Wireless Propagation Letters*, vol. 11, pp. 1350–1353, 2012.

[9] L. Liu, S. W. Cheung, R. Azim, and M. T. Islam, "A compact circular-ring antenna for ultra-wideband applications," *Microwave and Optical Technology Letters*, vol. 53, no. 10, pp. 2283–2288, 2011.

[10] J.-J. Tiang, M. T. Islam, N. Misran, and M. Singh, "Circular microstrip slot antenna for dual-frequency RFID application," *Progress In Electromagnetics Research*, vol. 120, pp. 499–512, 2011.

[11] Y. Ge, K. P. Esselle, and T. S. Bird, "E-shaped patch antennas for high-speed wireless networks," *IEEE Transactions on Antennas and Propagation*, vol. 52, no. 12, pp. 3213–3219, 2004.

[12] D. K. Neog, S. S. Pattnaik, D. C. Panda, S. Devi, M. Dutta, and O. P. Bajpai, "New expression for the resonant frequency of an E-shaped microstrip patch antenna," *Microwave and Optical Technology Letters*, vol. 48, no. 8, pp. 1561–1563, 2006.

[13] B. L. Ooi and Q. Shen, "A novel E-shaped broadband microstrip patch antenna," *Microwave and Optical Technology Letters*, vol. 27, no. 5, pp. 348–352, 2000.

[14] A. K. Bhattacharyya, R. Garg, and R. Garg, "Generalised transmission line model for microstrip patches," *IEE Proceedings H Microwaves, Antennas and Propagation*, vol. 132, no. 2, pp. 93–98, 1985.

[15] A. Taflove, *Computational Electrodynamics: The Finite-Difference Time Domain Method*, Artech House, Boston,USA, 1995.

[16] R. F. Harrington, *Field Computation by Moment Methods*, IEEE Press, Piscataway, NJ, USA, 1993.

[17] M. Paulson, S. O. Kundukulam, C. K. Aanandan, and P. Mohanan, "Resonance frequencies of compact microstrip antenna," *Electronics Letters*, vol. 37, no. 19, pp. 1151–1153, 2001.

[18] D. K. Neog and R. Devi, "Determination of resonant frequency of slot-loaded rectangular microstrip patch antennas," *Microwave and Optical Technology Letters*, vol. 52, pp. 446–448, 2010.

[19] P. Palanisamy, I. Rajendran, and S. Shanmugasundaram, "Prediction of tool wear using regression and ANN models in end-milling operation," *The International Journal of Advanced Manufacturing Technology*, vol. 37, no. 1-2, pp. 29–41, 2008.

[20] S. Wang, L. Zhu, J. Y. H. Fuh, H. Zhang, and W. Yan, "Multiphysics modeling and Gaussian process regression analysis of cladding track geometry for direct energy deposition," *Optics and Lasers in Engineering*, vol. 127, Article ID 105950, 2020.

[21] S. Tong, X. Zhang, Z. Tong, Y. Wu, N. Tang, and W. Zhong, "Online ash fouling prediction for boiler heating surfaces based on wavelet analysis and support vector regression," *Energies*, vol. 13, no. 1, p. 59, 2020.

[22] A. Torres-Barrán, Á. Alonso, and J. R. Dorronsoro, "Regression tree ensembles for wind energy and solar radiation prediction," *Neurocomputing*, vol. 326-327, pp. 151–160, 2019.

[23] S. Ülker, "Support vector regression analysis for the design of feed in a rectangular patch antenna," in *Proceedings of the 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT)*, IEEE, pp. 1–3, Ankara, Turkey, October 2019.

[24] K. V. Rop, *Parameter Optimization in Design of a Microstrip Patch Antenna Using Adaptive Neuro-Fuzzy Inference System Technique*, PhD Thesis, University of Nairobi, Kenya, 2014.

[25] F. Güneş, N. T. Tokan, and F. Gürgen, "A consensual modeling of the expert systems applied to microwave devices," *International Journal of RF and Microwave Computer-Aided Engineering*, vol. 20, pp. 430–440, 2010.

[26] E. Yigit, "Operating frequency estimation of slot antenna by using adapted kNN algorithm," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 6, no. 1, pp. 29–32, 2018.

[27] D. B. Davidson, *Computational Electromagnetics for RF and Microwave Engineering*, Cambridge University Press, Cambridge, UK, 2005.

[28] E. M. HyperLynx® 3D, *Version 15, Mentor Graphics Corporation, 8005 SW Boeckman Road*, Wilsonville, OR 97070, USA.

[29] D. J. Cannon, D. J. Brayshaw, J. Methven, P. J. Coker, and D. Lenaghan, "Using reanalysis data to quantify extreme wind power generation statistics: a 33 year case study in Great Britain," *Renewable Energy*, vol. 75, pp. 767–778, 2015.

[30] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*, MIT Press, Massachusetts, MA, USA, 2006.

[31] A. J. Smola and B. Schölkopf, *A Tutorial on Support Vector Regression*, Technical Report NC-TR-98-030, NeuroCOLT Royal Holloway College, University of London, UK, 1998.

[32] Y. Chen, J. Wang, R. Xia, Q. Zhang, Z. Cao, and K. Yang, "The visual object tracking algorithm research based on adaptive combination kernel," *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, no. 12, pp. 4855–4867, 2019.

[33] Y. Peng, K. Lei, X. Yang, and J. Peng, "Improved chaotic quantum-behaved particle swarm optimization algorithm for fuzzy neural Network and its application," *Mathematical Problems in Engineering*, vol. 2020, Article ID 9464593, 11 pages, 2020.

Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 1479507, 18 pages
https://doi.org/10.1155/2020/1479507

Review Article

Sailing through the COVID-19 Crisis by Using AI for Financial Market Predictions

Gagan Deep Sharma,[1] **Burak Erkut**,[2,3] **Mansi Jain**,[1] **Tuğberk Kaya**,[2,4] **Mandeep Mahendru**,[5,6] **Mrinalini Srivastava**,[1] **Raminder Singh Uppal**,[7] **and Sanjeet Singh**[8]

[1]*University School of Management Studies, Guru Gobind Singh Indraprastha University, Sector 16C, Dwarka, New Delhi 110078, India*
[2]*Institute for Research in Economic and Fiscal Issues, Paris 75017, France*
[3]*Department of Business Administration, Faculty of Economics, Administrative and Social Sciences, Bahcesehir Cyprus University, Nicosia 99010, Northern Cyprus, Turkey*
[4]*Department of Management Information Systems, School of Applied Sciences, Cyprus International University, Nicosia, Northern Cyprus, Turkey*
[5]*State Bank Institute of Credit and Risk Management, Gurugram, Haryana, India*
[6]*State Bank Institute of Leadership, Kolkata, West Bengal, India*
[7]*BBSBE College, Fatehgarh Sahib 140407, Punjab, India*
[8]*Chandigarh University, Gharuan 140413, Punjab, India*

Correspondence should be addressed to Raminder Singh Uppal; rsuppal@gmail.com

Received 22 July 2020; Revised 11 November 2020; Accepted 16 December 2020; Published 30 December 2020

Academic Editor: Manjit Kaur

The outbreak of COVID-19 has brought the world to an unprecedented position where financial and mental resources are drying up. Livelihoods are being lost, and it is becoming tough to save lives. These are the times to think of unprecedented solutions to the financial challenges being faced. Artificial intelligence (AI) has provided a fresh approach to finance through its implementation in the prediction of financial market prices by promising more generalizable results for stock market forecasting. Immense literature has attempted to apply AI and machine learning for predicting stock market returns and volatilities. The research on the applications of AI in finance lacks a consolidated overview of different research directions, findings, methodological approaches, and contributions. Therefore, there is a need to consolidate the extant literature in this upcoming field to consolidate the findings, identify the research gaps in the existing literature, and set a research agenda for future researchers. This paper addresses this need by synthesizing the extant literature in the form of a systematic review for addressing the use of AI in stock market predictions and interpreting the results in a narrative review. The gap formed through this article is the use of a combination of AI as a subject with the neural network as another area and stock market forecasting as another theme, and it will pave the way for future research studies. The analyses help highlight four important gaps in the existing literature on the subject.

1. Introduction

The world is facing an unprecedented situation with the outbreak of COVID-19 across the globe. Locking down the economies has emerged as a prominent measure to contain the spread of the coronavirus but leads to the loss of livelihoods, nevertheless. Whether or not to impose lockdown is the dilemma of saving lives versus livelihoods [1]. Recent empirical evidence shows that fake news associated with COVID-19 leads to panic, distrust, and confusion, whereas isolation leads to confusion in the thoughts of individuals [2]. Policymakers worldwide are choosing from these two choices of imposing lockdown or not, depending upon their demographics, economic situation, medical infrastructure, and social dynamics [3, 4]. The psychological effects of the crisis affect people's capacity to make prudent financial and nonfinancial decisions [5–7]. This unprecedented crisis calls for unusual solutions to help sustain people's well-being. Using techniques such as artificial intelligence for predictions is emerging as one key idea in such a situation.

Complex economic relations and models and high-frequency events in ever-evolving markets make propagating linear relations amongst economic and financial variables outdated [8, 9]. For estimating complex nonlinear relations and predicting stock market returns, computer science and econometric analysis go hand in hand. Advancements in computing technologies and econometric methodologies are changing the face of stock market predictions in recent times. Typically, econometric modeling, such as autoregressive (AR), moving average (MA), autoregressive moving average (ARMA), autoregressive integrated moving average (ARIMA), generalized autoregressive conditional heteroskedasticity (GARCH) model, buying and hold (B and H) strategy, random walk (RW), stochastic volatility (SV) model, has been employed for predicting stock returns [10], which is now being complemented by artificial intelligence and machine learning systems including artificial neural networks (ANNs), multilayer perceptron (MLP), fuzzy inference systems (FIS), adaptive neuro-fuzzy inference systems (ANFIS), and support vector modeling [11, 12].

Artificial neural networks (ANNs) can be considered as neural networks replicated in vitro. A neural network is a meshwork of billions of neurons (cells which are the basic unit of the nervous system) that are involved in processing information. ANNs are analogous to these neural structures as input is required with an optimum amount of threshold to process the inputs and provide a relevant output [13]. Within the neural framework, each neuron carries a weighted combination of many input signals, which is further converted to the activation threshold, and ultimately, an output is generated. Likewise, it is the nature of ANNs where many input signals are combined as a weighted input signal and are transformed as output after the mathematical calculation of all the input signals. The structure of MLP is a three-layered network with an input layer, hidden layer, and output layer [14], which utilizes a backpropagation algorithm for learning. Various algorithms, such as fuzzy logic (FL), probabilistic reasoning (PR), neural networks (NNs), and genetic algorithms (GAs), can be used in symbiotic association in soft computing to solve complex problems of the real world [15].

Artificial intelligence has provided a new and fresh approach to finance through its implementation in the prediction of financial market prices by promising more generalizable results for stock market forecasting. Immense literature has attempted to apply artificial intelligence and machine learning for predicting stock market returns and volatilities [13, 16–45]. Nevertheless, the research on the applications of artificial intelligence in finance lacks a consolidated overview of different research directions, findings, methodological approaches, and contributions. Therefore, there is a need to consolidate the extant literature in this upcoming field to consolidate the findings, identify the research gaps in the existing literature, and set a research agenda for future researchers. This paper addresses this need by synthesizing the extant literature in the form of a systematic review for addressing the use of AI in stock market predictions and interpreting the results in a narrative review.

This paper provides a new outlook on the conceptual synthesis of the extant literature. The researchers can further utilize the existing studies for developing more complex models using machine learning (ML) techniques for forecasting returns, modeling risk, and portfolio construction. The complexity of the nonlinear data can better be served with ML techniques. Multiple-layer perceptron (MLP), support vector machine (SVM), and long short-term memory (LSTM) model are the methods used for quantitative finance. A combination of these methods for deep belief networks and extreme learning machines can predict stocks more accurately [46]. The authors have resolved the conceptual way of explaining the need for more research work in deep learning for advancement in prediction modeling in financial markets. Chatrabgoun et al. [47] explain the use of the Bayesian network with pair-copula construction (BN-PCC) for tackling the issue of heavy tail data in financial applications [48].

Big data in recent times is providing more nonlinear datasets, which can be computed through the use of the ML technique. Furthermore, decision-making involves controlling the variables, which leads to uncertainty for allowing more accurate forecasts in financial markets. Thus, this study serves well by providing an account of variables associated with market prediction and the methodology used to study them, such as hybrid models [49]. However, this becomes pertinent that the models used in the previous studies shall be reinvigorated for their application on nonlinear datasets presented by big data. Techniques such as MLP, SVM, and LSTM can be rekindled with sentiment and emotional analysis for deep learning and prediction accuracy. This paper contributes to the literature in a more unified way by conceptualizing the extant literature and its major findings and methodology implemented for paving the way to future researchers to develop advanced prediction models and frameworks for understanding the variables related to uncertainty in the markets for accuracy in financial market forecasting.

The rest of the paper is organized as follows. In Section 2, the research questions briefly discussed in the introductory section are formulated, and the methodology is introduced. Section 3 presents the findings, where the authors introduce the paradigms associated with artificial intelligence and its use in the predictions of stock behavior under the sections named as descriptive findings. Section 4 involves thematic discussions, whereas Section 5 drives some insights concerning future research and practice.

2. Materials and Methods

2.1. Research Questions. Liberalization of the financial markets, along with the developments in strong communications and trading facilities [50], has led to a variety of investment alternatives. While the financial markets of the developed nations are highly integrated, the conventional capital markets theory has also evolved, and the financial analysis techniques have upgraded considerably [51].

With the important changes taking place over the last two decades in international finance, most of the financial markets are highly embraced now. The authors in [52] use the daily historical price of the North American stock index DJI and Nasdaq (USA), IPC (Mexico), and TSE (Canada) to foresee the stock indices' sign variations by implementing the logic models and fuzzy logic models which provided favorable returns when used as a trading strategy. Additionally, O'Connor and Madden [53] assess the magnitude of the impact of using external indicators such as commodity prices and currency exchange rates in predicting movements in the Dow Jones Industrial Average index, basing the trading decisions on a neural network that provide reliable results. Other papers on the developed financial markets include [20, 45, 54–58]. Qiu et al. [59] apply the artificial neural network involving a hybrid approach based on the genetic algorithm (GA) and simulated annealing (SA) to improve the prediction accuracy of the ANN of the Japanese Nikkei 225 index and proved it to perform better.

Cao et al. [19] use artificial neural networks to anticipate stock prices for the companies listed on the Shanghai stock exchange and opine neural networks as a useful method to predict prices in emerging markets, such as China. Additionally, Chen and Li [60] compare the five models, namely, the linear AR model, the LSTAR and ESTAR smooth transition autoregressive model, and the two ANN models that are MLP and JCN, and find that neural network technique made more accurate predictions. Additionally, other papers with the Chinese stock market as the sample include [55, 57, 61, 62]. Considering the Asian markets, particularly the Nikkei 225 closing index and Shanghai B-share closing index, Dai et al. [22] develop a time series forecast model by integrating nonlinear, independent component analysis and neural networks and prove a better option for the Asian stock markets.

RQ1: –to consolidate the findings of the existing literature related to the impact of artificial intelligence on stock market predictions

In order to generate optimum investment returns in the real-world stock exchange, many studies have employed decision support systems (DSS) techniques, soft computing models, and hybrid structures. However, Pham et al. [58] propose a hybrid Kansei-SOM model, a recent method using Kansei assessment combined with DSS techniques using collective decision-making for investment in the stock market and show that the proposed method performs better than the other prevailing methods. Another paper by Lin et al. [63] proposes a new technique called empirical mode decomposition combined with the *k*-nearest neighbors (EMD-KNN) method in forecasting the stock index, and the results demonstrate it to be more successful than the other methods. Additionally, the authors in [64] develop a new model combining the EMD with stochastic time strength neural network (STNN) and conclude it as a better model in forecasting stock market fluctuations.

Many researchers and financial analysts focus on nonlinear ties in the movement of the financial market, which calls for a new form of financial analysis, i.e., the nonlinear analysis of integrated financial markets. The study by Poddig and Rehkugler [51] employs artificial neural networks (ANNs) and other econometric models to establish the United States, Japan, and Germany's global reserve, bond, and currency business model. A paper by Cao et al. [19] compares the predictive potential of linear models and multivariate models of the neural networks and suggests neural network as an enhanced and effective method for forecasting stock market movements.

Reviewing the stock market's microstructure from a new perspective, Liang [65] uses the neural network technique to learn the complex linking of stock market information sources and determine that experiments illustrate this association statistically. Alternatively, Li et al. [66] study another theme in the concerned field that is the trading volume and the asset price risk using the sentiment analysis and machine learning approaches, including the ANN and support vector machine (SVM). Another topic of interest is captured by Liang [67] with a focus on studying the relationship between stock news on the Internet and stock price movements using neural network.

RQ2: –to understand the potential of different methodological AI approaches to capture the increased complexity and the complex nature of economic and political relations influencing stock price movements and their volatility

The intense research in neural networks produced new techniques for many potential applications, including the ability to forecast future movements of stocks, indices, and currencies, and additionally, Rehkugler and Poddig [68] conclude that, on comparing its results with those of a traditional model based on the multivariate regression analysis, the neural networks produced relevant results. Many papers sustained the implementation of the ANNs for a long time, even after their inception [69–71].

With the aim to outperform the other individual models, approaches involving the integration or the hybridization of different models in a composite model consisting of numerous linear prediction models, nonlinear models, or variations of both are implemented [46]. Kim et al. [72] investigate the efficiency of a hybrid approach with the time-delay neural networks (TDNNs) and the genetic algorithms (GAs) in identifying time-based stock market predictive tasks and find that evaluating the integrated solution is better than the traditional TDNN and the recurrent neural networks (RNNs). Additionally, to predict the settlement price of stock index futures, Li [30] develops a hybrid model involving the decomposition of empirical mode and function of the radial base that results into higher prediction accuracy. Forecasting future trends are one of the key considerations when using a prediction model. Hence, researchers continue to provide upgradations in the time series models. Since the artificial neural networks need a large amount of data, Khashei et al. [73] propose a modern hybrid neural artificial networks and fuzzy regression model for time series forecasting and conclude it to yield better results.

Bohn et al. [74] discuss the new system named MoneyBee that predicts the stock market values, especially from the German stock market, and conclude this information technology product as an innovation due to its cooperation between multiple high-level program groups.

The estimation of stock market movements is one of the key fields, with several approaches used, including the conventional computational techniques to artificial neural networks. However, the traditional approach to statistical economics has difficulties scaling because of noise and nonlinearity in time series [75]. Considering this feature of the traditional approach, Zhang et al. [75] propose the simple wave signal decomposition-based prediction model and the introduction of the α-counting instantaneous frequency of the simple wave and find it a better model.

Clearly, the literature states that whenever a new hybrid model is implemented, or a new approach is adopted to predict the stock market movements, the results produced are positive and favorable from the investment point of view. Hence, this indicates that every new technique and methodology implemented will lead to consistent improvements in the relevant models in artificial intelligence and, in particular, neural networks.

RQ3: to highlight the future research directions that may create value with AI in the context of stock market predictions

Recent research studies have paved the way for the inclusion of neural networks with AI for forecasting stock markets. Rather et al. [46] suggested the use of AI models for optimal portfolio selection through intelligent modeling. Further, risk modeling and return forecasting can be modelled by using the subfield of AI, which is machine learning (ML) [48]. This may enhance the probability to achieve better optimization models in the prediction of stocks in the longer run.

RQ4: to understand the capacity of AI in predicting the financial market behavior during pandemic times as COVID-19

The present review is an attempt in providing a constructive path for forecasting stock markets, and the use of AI models in predicting the financial market behavior during pandemic times seems promising. The correlation that may arise due to sentiment-related volatility in stock markets [76] may be analysed through the machine learning models.

2.2. Methods. The research approach is influenced by the works in [77–80]. It is important to note that a systematic review involves reading several excerpts from an article, considering not only the overview of the keywords and the abstract but a thorough examination of each paper, taking their methods, conclusions, and findings into consideration. This is imperative to categorize their content more meaningfully while examining a particular field of study. Therefore, this review takes into consideration the research works that directly or indirectly discuss the applications of neural networks and artificial intelligence in predicting stock market by carrying out the following activities:

(1) Analyzing the articles previously published

(2) Providing a short description of the contribution of these articles

(3) Categorizing and coding the different parameters

(4) Describing their different contributions

(5) Providing the scope for future research to fill the gaps identified herein

The authors jointly delineated two keywords, namely, “neural network” and “stock market forecasting.” “Fuzzy logic” and “artificial intelligence” were decided as synonymous with “neural network.” In contrast, “stock return,” “share market,” and “share price” were decided to be used as synonymous with the “stock market” for searching the literature. The worldwide accepted online database, Web of Science (WoS), was used for searching the relevant papers. The search was conducted on the Web of Science database on December 24, 2017, using the following BOOLEAN criteria: TS = ((Neural Network* OR Fuzzy Logic* OR Artificial Intelligence) AND (stock market* or stock return* or stock price* or share market* or share return* or share price*)).

This resulted in an initial list of 1040 papers, as shown in Figure 1.

The list of 1040 papers is discussed for designing an objective criterion to select/reject the papers for review. Manual exclusion and inclusion are done by the authors as follows.

Each paper was examined individually by the authors as A (accept), B (reject), and C (doubtful), which was carried out solely based on the abstract and the keywords. The results are showcased in Figure 1, which exhibits that 1040 papers were extracted from the database in the first stage, out of which 250 papers were accepted for review, 605 were rejected, and 185 were put in the doubtful category.

The second stage carried out the reassessment of these papers, which involved going through the entire paper for better understanding and correct categorization. 128 papers were rejected based on an irrelevant field of study. Eventually, 57 were accepted from the list of doubtful papers.

Finally, a total of 307 papers were selected for the systematic review. Table 1 highlights the categories and subcategories for the systematic literature review.

3. Results

3.1. Descriptive Results

3.1.1. Geographical Distribution. The first category deals with the geographical base studied in the existing literature. Table 2 represents the geographical area distribution.

It is evident from Table 2 that the majority of contributions focus on countries from Asia, which is assigned code B. Asia is representative of 39% of research in artificial neural networks (ANNs) for stock market prediction. Code C, representing the USA, shares 15% of its research work in this context. CW countries share 13% of their knowledge base in this background. At the same time, 33% of geographical distribution does not belong to any particular geographical location. It is important to undertake more research in designing prediction models for stock markets. As stock markets play a major role in a country’s economic growth, it is important to have some prediction models which can help

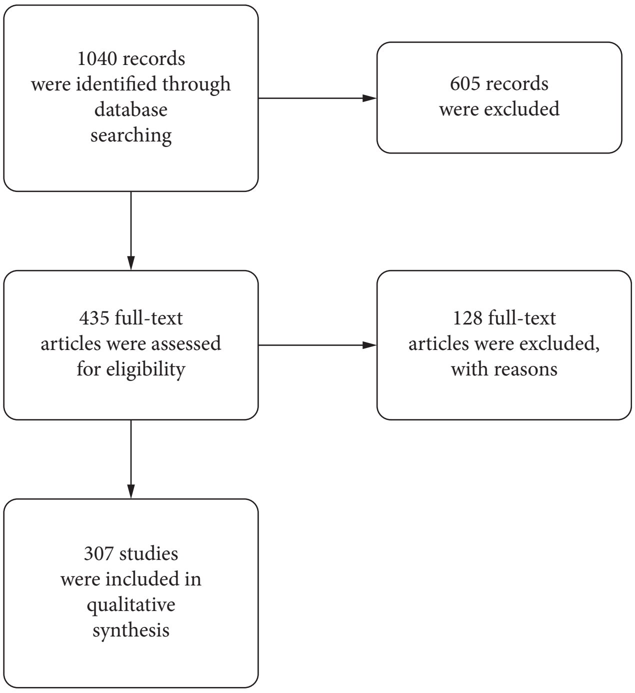


Figure 1: Study flow diagram (PRISMA figure).

in the forecasting of price movements in stock markets for the benefit of investors. Figure 2 represents the geographical area distribution.

3.1.2. Context. The second category relates to the coverage and is divided into three categories: A stands for developed countries, B for developing countries, and C for not applicable. Table 2 depicts that 55% of the contribution is made by developed countries and 41% is by developing countries. The majority of studies have taken place in developed countries. This is suggestive of the fact that researchers in developed nations have started analyzing the issue of prediction of stock prices. However, this is a major concern of investors across the globe, and such studies need to be undertaken in developing nations also (Figure 3).

Gap 1: studies focusing on the future of artificial neural networks in stock market prediction models in developing countries must be taken up.

3.1.3. Method. This category of coding indicates the method applied to the available. Table 2 presents the methodology with seven categories.

Category D presents the maximum number of papers. These papers have used more than one method of analysis. Conceptual papers are denoted as code C, which represents the next highest number of studies. Figure 4 presents the methodology from the existing literature.

It can be concluded that conceptual and theoretical work has been done in this context. The gap which emerges is as follows.

Gap 2: there is much wider scope in the field of artificial intelligence to implement other methodological approaches; for instance, conducting a case study or a survey or an experiment using a hybrid model can further help in the development of more generalized models of stock prediction.

3.1.4. Main Subject. The category of the main subject is classified into five types, as mentioned in Table 2. The maximum number of publications are available in the field of neural network and other computing techniques. The analysis of this category implies that a major part of the literature talks about studies on the combination of artificial intelligence and neural networks [14, 57, 81–83]. Research related to predicting stock indices, such as genetic algorithms and neural networks, and linear and nonlinear models recognizes patterns by estimating coefficients and their statistical significance. However, most agents in the stock market use language that incorporates qualitative aspects such as the price of the asset, but the authors in [52] opine that predictive models of fuzzy logic achieve statistically significant and positive returns when used in trading strategy [84]. Model predictive control (MPC) is regarded as the most widely applied control technique capable of handling constraints and incorporates other economic considerations [85]. In the context of portfolio optimization, where the challenge also lies in finding the optimal trade-off between risk and return over a fixed time horizon, Herzog et al. [86] propose to use the model predictive model (MPC) to optimize the portfolio by the probabilities and parameters of the implied regime including transaction costs, risk aversion, and other restraints. Nystrup et al. [84] employ the model predictive control (MPC) to optimize a portfolio based on forecasts of the mean and variance of financial returns from a hidden Markov model and conclude that MPC yields better returns and relatively lesser risk than investment in other stock markets. On the other hand, Yamada and Primbs [87] employ MPC in the context of the hedge funds incorporating the issues of gross exposure and transaction costs, while Dombrovskii and Obedko [88] adopt MPC to develop feedback portfolio optimization strategies, indicating extensive reliability on this technique in financial applications, especially in cases of portfolio optimization and dynamic hedging. Concerning our main topic, extensive work could be taken up, which will ultimately lead to algorithms and prediction models in this background (Figure 5).

3.1.5. Themes. The themes are divided into ten different categories, as depicted in Table 2. It can be drawn that a major part of the studies available is in the field of computer sciences and business economics (Figure 6).

There arises a need for more studies to be taken up under themes of material sciences, soft computing, optimization techniques, automation, and control system. This brings us to the following gap.

Gap 3: the scope of research lies in the trade-off between different optimization algorithms and the adoption of the best algorithm as the solution. This can be achieved by filling the gaps under other themes of science as they can provide

Table 1: Categories and subcategories of the systematic literature review.

Category	Meaning	Codes for alternatives
1	Geographical distribution	Commonwealth countries Asia USA Not applicable
2	Context	Developed countries Developing/emerging countries Not applicable
3	Methodology	Quantitative Qualitative Conceptual Quantitative/qualitative or qualitative/quantitative Survey Case study Not applicable
4	Main subject	Stock return prediction Other financial returns prediction Artificial intelligence Neural networks Other computing techniques
5	Themes	Neuroscience Optimization techniques (science tech.) Business economics Soft computing (engineering) Material science Automation and control system Operational research and management science Physics and mathematics Computer science Other topics
6	Contribution	New perspectives Consistent with previous literature Previous model with different datasets/time periods Comparative study Not applicable
7	Analysis period	Less than 3 years Between 3 and 5 years Between 5 and 10 years More than 10 years Not applicable

Table 2: Codes and categories of selected papers.

Code	Geographical distribution	Context	Method	Main subject	Themes	Contribution	Analysis period
A	40 (13%)	169 (55%)	3 (1%)	23 (7.49%)	6 (2%)	40 (13.02%)	25 (8.14%)
B	120 (39%)	126 (41%)	9 (3%)	43 (14%)	4 (1%)	34 (11.07%)	14 (4.56%)
C	46 (15%)	12 (4%)	52 (17%)	24 (7.87%)	54 (18%)	35 (11.40%)	124 (40.39%)
D	101 (33%)	NA	126 (41%)	51 (16.61%)	15 (5%)	45 (14.65%)	144 (46.90%)
E	NA	NA	31 (10%)	52 (16.93%)	2 (1%)	43 (14.33%)	NA
F	NA	NA	37 (12%)	NA	4 (1%)	NA	NA
G	NA	NA	49 (16%)	NA	4 (1%)	NA	NA
H	NA	NA	NA	NA	19 (6%)	NA	NA
I	NA	NA	NA	NA	192 (63%)	NA	NA
J	NA	NA	NA	NA	7 (2%)	NA	NA
Multiple	NA	NA	NA	114 (37.13%)	NA	109 (35.50%)	NA

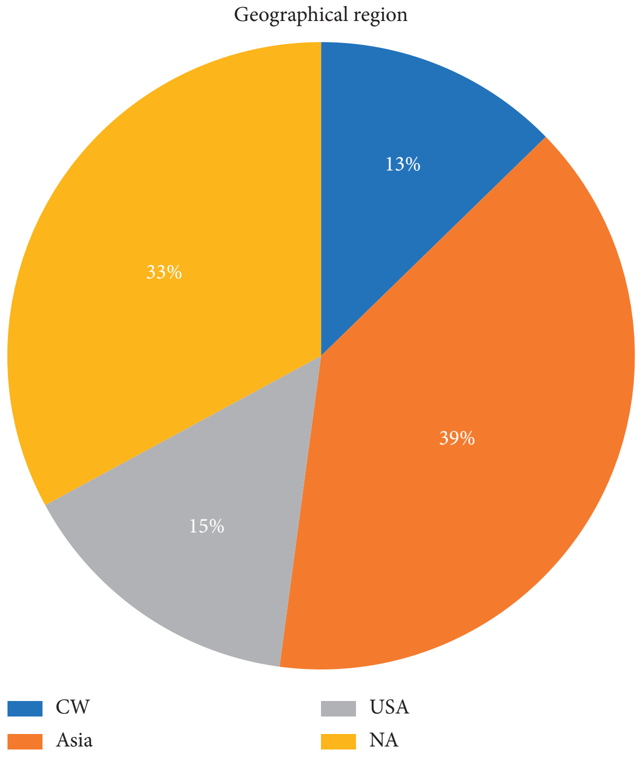


Figure 2: Distribution of the geographic region in the existing literature.

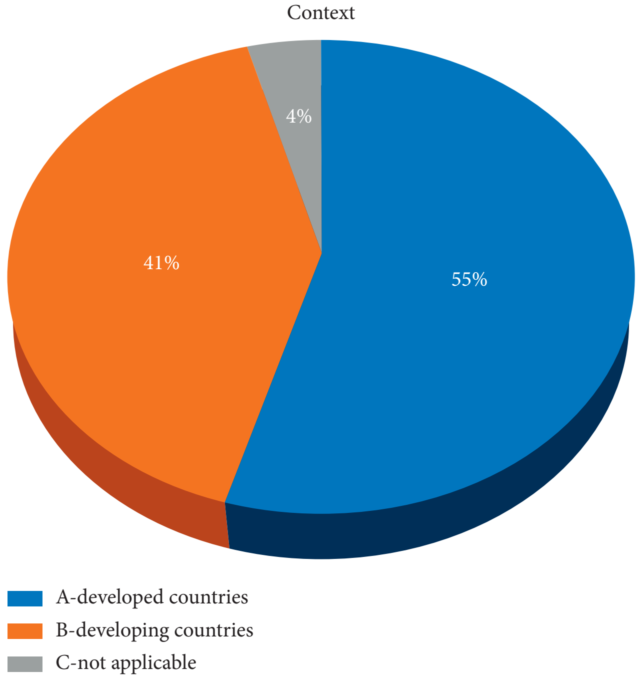


Figure 3: Context of existing research papers.

better insights into AI for the development of stock market prediction models.

3.1.6. Contribution. This classification presents the contribution of the articles understudy, presented in Table 2. The contribution made by category D is the maximum number of 45 papers followed by category "E" with 44 and further followed by "A" with 40. Hence, it seems to be an ideal spread of the previous literature with a fine number of papers concentrating on either implementing previously developed models with a new dataset or drawing a comparison between different techniques or an extension of findings from the previous literature (Figure 7).

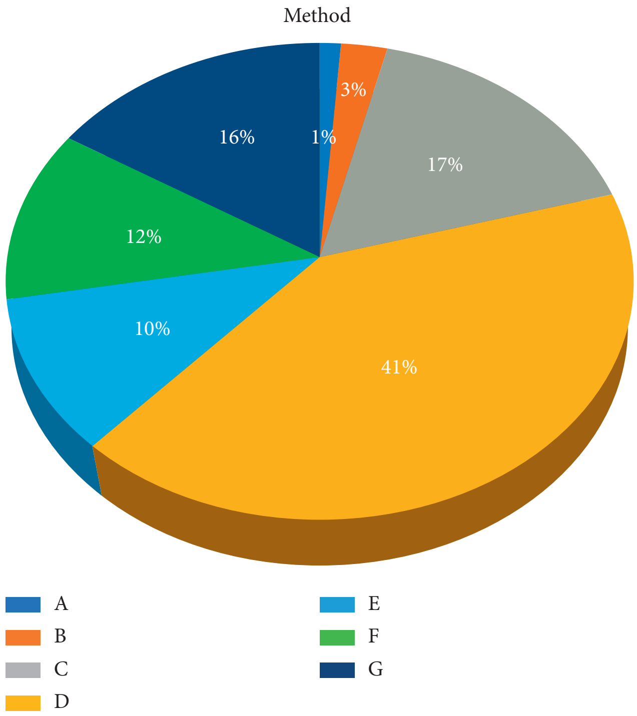


Figure 4: Methodology employed in previous papers.

3.1.7. Analysis Period. The analysis period indicates that, in the last three years, only 25 papers are published. It is also less for a period of 3 to 5 years, accounting for only 14 papers. However, AI as a concept emerged long back in the 1960s and the concept of the neural network is indeed an old established concept in neurology. The merging of these two concepts is still in a quandary. There is a dire need to uptake more work in this field and reap benefits out of AI and neural networks for the prediction of stock markets (Figure 8).

Gap 4: more studies are required to see the consistency throughout the time period since the inception of these three topics, viz., AI, neural networks, and stock market forecasting.

4. Discussion

Using the method suggested in [77–80], an analysis of the relevant literature available on the Web of Sciences (WoS) was conducted to identify the major themes in studying the application of artificial intelligence and related techniques in predicting stock behavior. The methodological details are presented in Section 2 of this paper. The characteristics and parameters studied by the existing literature are segregated and presented in this section.

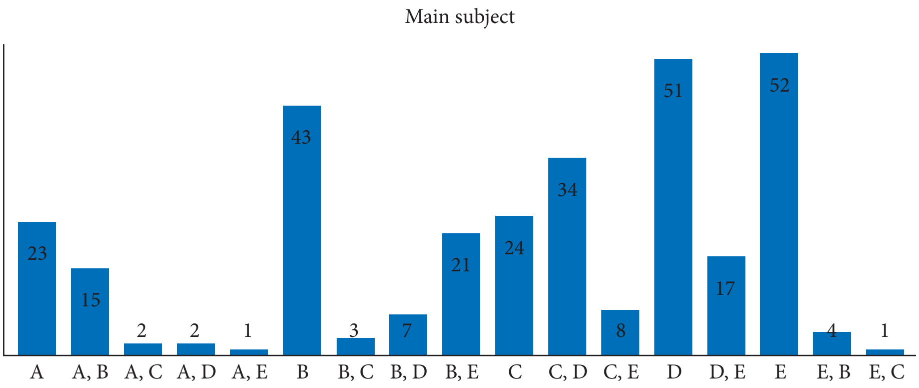


FIGURE 5: Distribution of main subject in the existing literature.

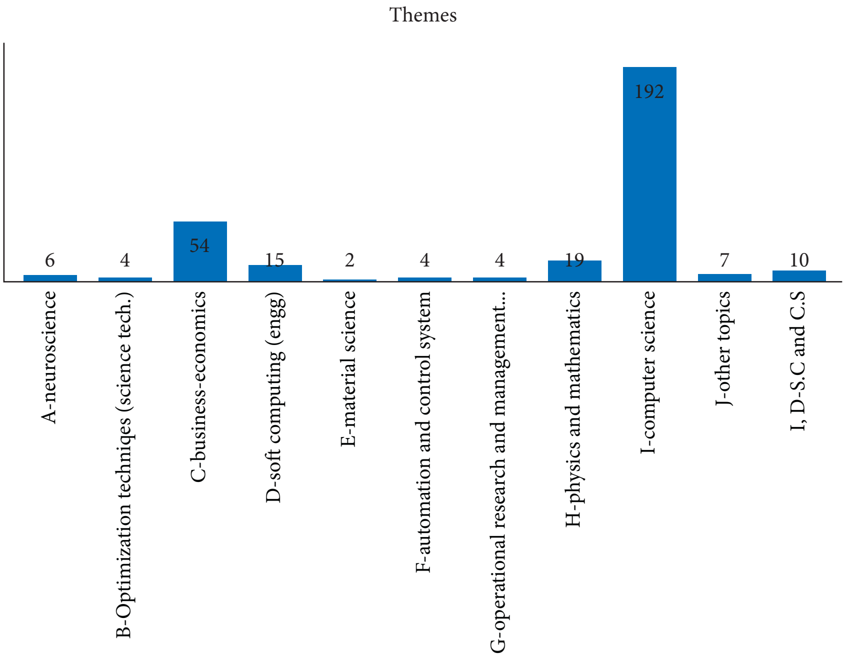
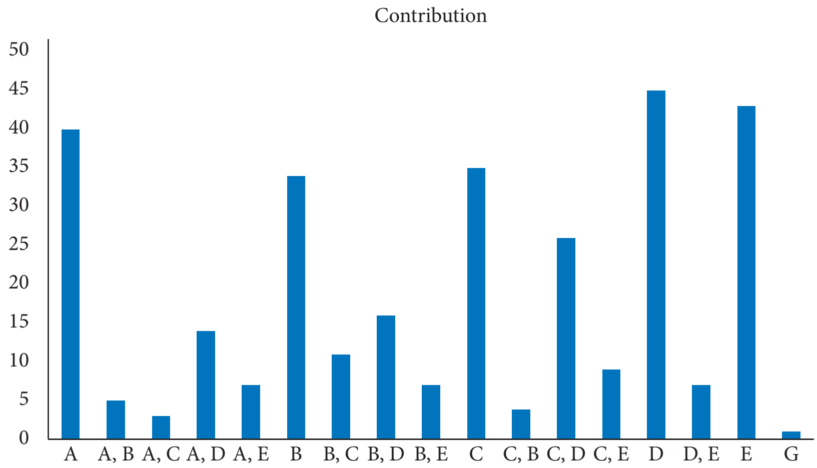


FIGURE 6: Themes of papers.



	A	A, B	A, C	A, D	A, E	B	B, C	B, D	B, E	C	C, B	C, D	C, E	D	D, E	E	G
▬ Contribution	40	5	3	14	7	34	11	16	7	35	4	26	9	45	7	43	1

FIGURE 7: Histogram showing the contribution to the paper.

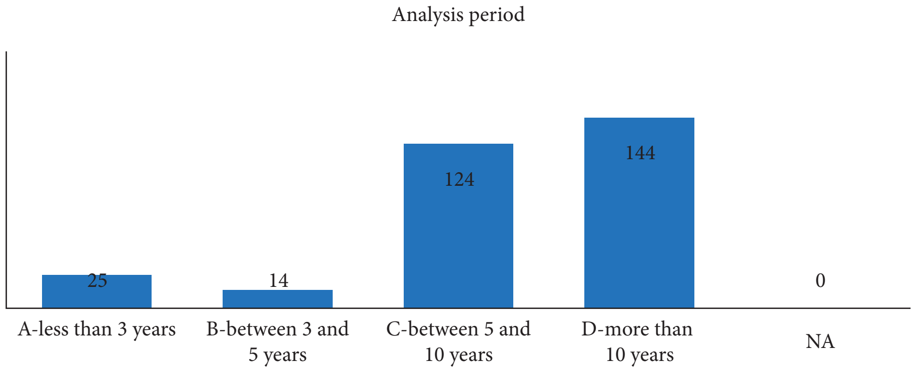


Figure 8: Analysis period from the existing literature.

4.1. Soft Computing. In the financial world, soft computing is gathering traction. A variety of real and potential soft computing technologies are employed in the financial sector, including commodity and retail market estimates, trade, portfolio management, credit scoring, or projections of financial distress [89]. Soft computing includes a variety of techniques that replicate the ability of the human imagination that applies reasoning approaches that are subjective and not precise. The term “soft computing” was coined by Zadeh in the early nineties [90], concluding fuzzy logic, neural network theory, and probabilistic reasoning as the main components of soft computing. Securities and foreign exchange forecasting is one classical field of soft computation in which forecasts of the conduct of hybrid currencies, currency exchange rates, and stock prices are determined [89]. A significant example of soft computation techniques used in stock market forecasting is given byBoyacioglu and Avci [11], where the authors utilize an adaptive network-based fuzzy inference system, with which they have a prediction accuracy of 98.3% for the Istanbul Stock Exchange. The authors use a combination of macroeconomic variables and indices with a set of if-then rules and layers based on the principles of ANN and fuzzy logic. Current research on predictive models in financial economics and econometrics distinguishes between parametric models and nonparametric models. In the case of option pricing models, the famous Black–Scholes model is an important primer for parametric models estimating option prices.

On the other hand, methods such as extreme learning machines and support vector regressions are mainly nonparametric models, with which financial predictions can be done. Das and Padhy [91] suggest a combination of parametric and nonparametric methods mentioned here in order to increase the predictive power of option pricing models. Here, the hybrid mode superiority is due to return distributions being nonnormal and the need for adaptive learning, which arises from the extreme learning machines method.

4.2. Neural Networks. As a branch of computational science, the neural network encompasses a broad range of techniques of function modeling focused on interconnected processing components, known as neurons, which work together to generate a particular output. Among the important approaches are the multilayer perceptrons, radial basis networks, or self-organizing maps [89]. Jayne et al. [92] investigate the applicability of neural networks by forecasting the values of each share, taking into account the general index value, and state that neural networks learn the complex mapping between ideal attributes and the particular domain parameters. Results from the multilayer perceptron method (MPL) and radial basis function (RBF) tests are good and acceptable considering the type of inversion problem addressed. Corresponding to this, Kim and Lee [93] compare the genetic algorithm (GA) with the linear transformational model (LTM) and fuzzy transformational model (FTM) for the artificial neural networks to function better in predicting the stock market patterns and report reduction in the irrelevant factors for stock market prediction. On the other hand, stock opening price forecasting has also gained momentum in recent years. To improve the accuracy of this forecasting, Qun et al. [94] propose a model incorporating the emotional data along with the actual behavior data and report improvement in the prediction accuracy. Accordingly, Nayak et al. [95] study the neural network of artificial chemical reaction to prepare multilayer perceptrons for stock market index forecasting and conclude significant improvement in the prediction accuracy. It cannot be concluded that neural networks always outperform traditional models in financial markets, as pointed out by Bahrammirzaee [96], who finds out that the opposite can also be true. Nevertheless, the use and performance of NNs are encouraging due to their numeric nature, their distribution-free approach, and the ability of NNs to update data. Nevertheless, due to some limitations in technical aspects of NN, NN itself can sometimes be problematic to use—in that case, hybrid models combining NN approaches with other techniques can be used [96]. One example of such hybridization of NN will be mentioned in Section 4.4.

4.3. Artificial Intelligence. AI is viewed as an alternative to predictive modeling. This computer science, or systems engineering, field was originally introduced in the 1950s to develop machinery intelligence [54]. Since then, a variety of AI methods have developed, including the Bayesian networks, artificial neural networks (ANNs), expert systems, support vector machines, and fuzzy-logic-based techniques.

In comparison with statistical techniques, AI does not make the same data assumptions as regression analysis. It performs well within datasets contaminated by variable noise and is effective in the impartial and semicontrolled analysis of data [54]. However, many real financial applications have uncertain behavior that changes over time, and this problem causes an increased interest in artificial intelligence applicability. Bahrammirzaee [96] reviews three AI techniques, namely, artificial neural networks, expert systems, and hybrid intelligence systems in the financial market and proves the AI techniques are superior to the traditional statistical methods in terms of their accuracy, majorly regarding the nonlinear patterns.

Alternatively, Li et al. [97] propose a modern blend prediction model based on AI techniques and integrated forecast concept, which may direct the investor's businesses in the real stock market and prove to be an effective instrument for decision making investments. Additionally, the authors in [59, 98] apply the ANN and genetic algorithm (GA) and find that this hybrid approach improves the stock prediction accuracy significantly and outperforms the traditional backpropagation (BP) learning algorithm. Wu and Duan [83] compare network structures of different neural networks in predicting the price trend of the Chinese stock market and conclude that the dynamic relationship between investors and market volatility can be thoroughly illustrated. Bebarta et al. [99] propose a model that selects optimum nonlinear combinations of Indian stock forecasting. The model proposed by the authors makes use of an evolutionary algorithm to optimize the weight parameters of different functional expansions to improve forecast accuracy.

Management of stock portfolios is a challenging task, and to solve such problems, artificial intelligence models are applied, where Rather et al. [46] conduct a survey highlighting the traditional mathematical models to artificial intelligence-based models available in recent articles. One problem that they identify is that, in many cases, linear and mathematical models for portfolio optimization work fairly well so that there may be no need for AI models. In addition to that, one common limitation of these AI models is the slow convergence and the fact that there is no guarantee for an optimal solution. One example of an unsuccessful attempt to make use of ANN in combination with genetic algorithms is given in [98], where even the improved version of their earlier model has a prediction accuracy of 63%. This also indicates that the complex nature of ANN cannot immediately deliver efficient and concrete predictions even though some scholars found methods to fasten the learning process (see, e.g., [12]).

Nevertheless, the evolutionary computation can deliver evidence from another perspective than optimization, which is known for being a primer in ergodic economic theory, as recently pointed out by Menger-Anderson [100]. Ghandar et al. [101] show that, using evolutionary computing methods, the machine itself can develop evolutionary trading rules with which an excellent performance in ever-evolving market conditions can be reached. On the other hand, Hamed et al. [12] combine ANN and MLP approaches with a method from signal processing called blind source separation technique. The signal processing technique utilized by the authors not only optimizes the learning process of the neural network but also fastens this process. Through this improvement, accurate and efficient results have been obtained for Microsoft stock movements, among others.

4.4. Fuzzy Logic. After being considered one of the main components of soft computing, fuzzy logic is derived from fuzzy set theory dealing with theoretical reasoning rather than precisely derived rationale. Unlike the traditional logic, it arrives at conclusions based on vague, ambiguous, noisy, or incomplete information [89]. Employing a fuzzy logic-based approach simplifies the design problem and describes the ambiguity and vagueness of pattern functions. It includes uncertainty in the trading decision system that advises the investor on how much and where to invest. Naranjo et al. [102] illustrate this methodology by proposing a fuzzy trading system that reveals improvement in pattern recognition and provides more benefits in a less volatile fashion than most trading structures. However, previous literature finds certain drawbacks in the forecasting models. To refine these, Wei et al. [103] propose a hybrid model employing a fuzzy inference system (FIS) and argue that the model is superior to the previous traditional models due to improvement in the forecasts generated. Since neural networks can be considered as having a black box nature sometimes [104], the problem is that these cannot be used to determine a causality relation between dependent and independent variables. To overcome this problem, instead of rejecting neural networks completely [104], make use of a hybridization technique to generate a fuzzy neural network model. This model combines the reasoning style from fuzzy models and learning styles of neural networks to yield interpretable results and high profits for stock trading decisions. Through this hybrid model, the authors show an improvement of neural network-based decision rules for stock trading. Alternatively, Bekiros et al. [105] propose a volatility-based neuro-fuzzy model to predict FTSE100 and New York Stock Exchange returns. This contribution shows that considering volatility changes in a neuro-fuzzy model can improve the accuracy of stock market forecasts and it outperforms other approaches such as a Markov switching model, a feedforward neural network model, and a buy and hold strategy when transaction costs are also taken into account. The managerial implication of this model is that a neuro-fuzzy model incorporating volatility changes can bring higher returns than the approach for passive fund management.

4.5. Stock Returns. The economic and social well-being of developing countries with reasonably privatized economies depends heavily on the financial sector in a region, which is seen as an important factor for growth and development. Over the years, researchers are trying to promote financing by offering credit and other financial goods, predict market patterns, model market, and consumer conduct, manage the portfolio, open stock price allocation, predict defaults and bankruptcy, etc. In respect of these, several methods are used that can be classified as parametric (e.g., logistic regression and discriminatory analysis), nonparametric (e.g., decision trees) mathematical techniques, and soft computing (e.g., artificial intelligence algorithms) techniques [96].

Investors, industrialists, and researchers remained focused on stock estimation and investment in suitable stocks. In the years, several models and techniques of artificial intelligence have been developed to solve such problems. However, Box and Jenkins's ARIMA models have gained popularity in the time series prediction field, while linear prediction models are still used in practice [106]. Complementary to this, Rather et al. [46] survey articles on stock prediction and stock selection for portfolios and opine more precise demand forecasts by AI-based models. Karathanasopoulos [107] introduces a new short-term adaptive model, i.e., partial swarm optimizer combined with linear and nonlinear models to forecast the daily closing returns of FTSE100 exchange-traded funds (ETFs) and find the model to perform well in terms of correct directional change.

On the other hand, Li et al. [108] forecast the REITs and stock indices by Group Method of Data Handling (GMDH) neural network method with Hurst and confirm the model to perform better than the conventional forecasting algorithms, including ARIMA, the single and double exponential smooth, and the neural network backpropagation. Additionally, a paper by Zhong and Enke [109] present a systematic data mining method using artificial neural networks and logistic regression models, to predict the S&P 500 Index ETF (SPY) daily return, and conclude that the risk-adjusted returns of the trading strategies are higher than the comparable benchmark. Bildirici and Ersin [10] focus on an M-GARCH type model that makes use of ANNs to improve forecasting accuracy. The novelty of this model in comparison with similar GARCH models is the increased predictive power of the model involving the regime-switching structure augmented with neural networks. The approach proposed by Chang et al. [110] makes use of the simplest form of business cycle theory, namely, the theory that tells us that there are ups and downs which we call turning points regarding stock prices, but also the economy in general. The novelty of the approach is to decompose historical data into segments, through which the authors identify temporary turning points by using piecewise linear representation. The model therefore gives investors clues about buying, selling, or holding stocks, which is a more preferred approach than forecasting the stock price only.

Additionally, the model predictive control (MPC) technique is developed by Herzog et al. [86] to solve restricted stochastic problems in terms of portfolio optimization, including inherent nonlinearity, constraints, and uncertainty [111]. Dombrovskii and Obedko [88] consider optimum portfolio selection issues subject to investment, trade, and various borrowing and lending rates limitations, use MPC for designing optimization feedback portfolio strategies, and further test the approach on real data of the Russian Stock Exchange, Moscow Interbank Currency Exchange, and New York Stock Exchange. Conversely, to gain insights into the stock market and allocation of capital, Trimborn et al. [112] employ MPC to approximate and solve the optimization of the utility function. To predict shifts in the term interest rate structure, Zantedeschi et al. [113] propose a dynamic product partition model (PPM) that relates macrovariables to term structures and indicate major economic disruptions, including recessions.

4.6. AI and Knowledge Management.

Artificial intelligence and knowledge management developed historically from different roots, but they share a common ground due to the subjective, tacit, dispersed nature of knowledge [114]. Sanzogni et al. [114] propose that collective tacit knowledge available in society and transferred to an individual via his or her interactions with the society, tacit relational knowledge based on contingencies concerning human interactions, and tacit somatic understanding lying within the mind of an individual are building the common grounds for these two concepts. The authors point out the fact that AI can be used to support human-driven knowledge processes, which are necessarily subjective. Still, the state-of-the-art research and practices show that AI is leading the way to autonomous intelligent machines which will rather substitute human-driven knowledge processes. The authors note that AI and KM evolved in parallel ways but need to be integrated for further research since AI-based technologies can be a key to the implementation of new knowledge strategies and KM visions for decision-making.

Furthermore, it is also noticed by the authors that AI technologies may be unaware of their actions and the political consequences of these actions, and this challenge can be managed by KM strategies. The latter point seems to be especially relevant for financial markets since disseminating fake or real news has a significant effect on these. In contrast, these build the macroeconomic implications of AI and its complex relationship with KM. Some microeconomic implications can be relevant for individual behavior, especially regarding individuals who directly or indirectly influence stock market prices. A primer in the interaction of these fields had been written by Dong and Zhou [115]. Using ANNs as their point of departure, the authors deliver empirical evidence showing the relation between stock price response and firm size, which differs across different firms. The novelty of this approach is to show that using the whole dividend event data to conclude may lack this differentiation across different types of firms. Hence, managing knowledge on stock price response can be improved by managing knowledge on firm types.

Big data, as an emerging field of application of AI, can be useful for personal knowledge management, as stated by Liu et al. [116]. The authors consider the fields where big data can be useful, such as time management, machine performance monitoring, activity monitoring of mobile devices, healthcare, and web navigation. Concerning financial decision-making, these areas can be improved both for security and for more efficient investment decisions. Nevertheless, this is a challenging issue, since real-world phenomena can be very complex, and often, there can be some omitted variables which may result in false causalities and conclusions [117].

Table 3: Research gaps and future agenda.

Gaps	Questions	Related studies
Gap 1–; there is a need to focus on the future of artificial neural networks in stock market prediction models in developing countries	RQ1: to integrate the findings of the previous research in the field of artificial intelligence in stock market prediction models in different parts of the world, with more focus on the developing economies	[10, 19, 22, 26, 60, 118–131]
Gap 2: there is a much wider scope in the field of artificial intelligence to implement other methodological approaches; for instance, conducting a case study or a survey or an experiment using a hybrid model can further help in the development of more generalized models of stock prediction	RQ2: to adopt other methodological approaches involving a case study, an experiment, or a survey	Experiments: [11, 29, 75, 91, 97] Survey: [96, 114] Case study: [91, 132–136]
Gap 3: the scope of research lies in choosing the optimal algorithm as the solution. A combination of other themes may provide better insights and models for stock market prediction.	RQ3: to conduct more research using a combination of other themes for stock market prediction, also because the existing artificial neural networks have not been able to provide effective results [137] and this calls for more advanced techniques such as neuro-fuzzy methods, genetic algorithm, option pricing models, machine learning techniques, and component analysis models	Neuro-fuzzy systems: [104, 105, 138–141] Genetic algorithm and linear representation methods: [28, 93, 142–148] Linear regression models: [21, 116, 122, 149] Option pricing model, machine learning techniques and hybrid combinations with neural networks: [34, 91, 150–153] Component analysis model: [22, 46, 62, 137, 154]
Gap4: more studies are required to maintain consistency throughout the time period since the inception of these three topics, viz., AI, neural networks, and stock market forecasting	RQ4: the majority of the study is concentrated on longer periods. Very few studies have conducted an experiment or a case study on the trading prices spread over the previous 6 months or less	[30, 38, 50, 75, 81, 91, 95, 98, 103, 146, 155–159]

5. Conclusions

The paper discussed the relevance of artificial neural networks (ANNs) in stock market prediction to cope up with the challenge of low financial well-being during the crisis of COVID-19. The existing literature in the field relates that AI and neural networks have a promising future together though the extent to which it can be applied should not be neglected. AI can provide multiple algorithms in solving the problems. The combination of neural networks with AI can provide for the development of human neurons and the functioning of the same in the human body. ANNs can help analyze nonlinear problems and more generalization of the solutions provided through this combined approach. Thus, stock prediction tools can be developed through the use of ANNs.

The existing literature on this subject has largely used the conceptual methodology, though a combination of quantitative and qualitative approaches has also been employed in some cases. The theory generalization aspect has not been studied extensively by the literature. It can also be seen that developing countries are not doing so well when confronted with the development of stock prediction models than their counterparts. Witnessing the current era of scientific developments brings us to a state where AI can play a major role in developing models of stock market prediction, and more research studies are needed. The gap formed through this article is the use of a combination of AI as a subject with the neural network as another area and stock market forecasting as another theme, and it will pave the way for future research works. The analyses help highlight four important gaps in the existing literature on the subject.

We focused on the 4 research questions, as outlined in Section 2, and found four important gaps in the existing literature. From these gaps, we formulate the future research agenda to serve as a guideline for future researchers interested in these topics (Table 3).

The limitations of this study can provide a precursor for further studies. The papers have been analysed from the Web of Science only. Other databases can also be referred for addressing new gaps. This study acknowledges the relationship between AI, neural networks, and stock market prediction. Therefore, their use in predicting sentiment-related volatility in stock markets especially during pandemic times of COVID-19 needs attention. Hence, new perspectives can be brought to light through the use of other databases and running more ML models for forecasting during COVID-19. The exploration of newer ideas will result in more application-based studies.

Disclosure

The funders had no role in the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Conflicts of Interest

The authors declare no conflicts of interest.

Authors' Contributions

G. D. S. and B. E. conceptualized the study; G. D. S. and M. M. were responsible for methodology; G. D. S., B. E., M. M., M. S., and M. J. performed formal analysis; R. S. U. and T. K. investigated the study; S. S. was responsible for resources; G. D. S., B. E., M. J., T. K., M. M., M. S., R. S. U., and S. S. prepared the original draft; M. S., M. J., and B. E. reviewed and edited the manuscript; M. M. visualized the study; R. S. U. supervised the study; G. D. S. was involved in project administration; B. E. was responsible for funding acquisition. All authors have read and agreed to the published version of the manuscript.

Acknowledgments

The authors wish to thank Bettina Beer from the Graduiertenakademie of TU Dresden for issues associated with project management. This paper has been written as a result of the project IPID4ALL-2017_76 as a collaboration between TU Dresden, GGS Indraprastha University, and All India Institute of Medical Sciences (AIIMS), New Delhi, supported by DAAD and DFG. The authors are thankful to DAAD and DFG for support. Also, Burak Erkut is thankful to Deutsche Bundesbank, which sponsored his second research visit at GGS Indraprastha University and AIIMS.

References

[1] G. D. Sharma and M. Mahendru, "Lives or livelihood: insights from locked-down India due to Covid-19," *Social Sciences & Humanities Open*, vol. 2, no. 1, Article ID 100036, 2020.

[2] G. D. Sharma, A. S. Ghura, M. Mahendru, B. Erkut, T. Kaur, and D. Bedi, "Panic during Covid-19 pandemic! A qualitative investigation into the psychosocial experiences of a sample of Indian people," *Frontiers in Psychology*, 2020.

[3] G. D. Sharma, G. Talan, and M. Jain, "Policy response to the economic challenge from Covid-19 in India : a qualitative enquiry," *Journal of Public Affairs*, pp. 1–16, 2020.

[4] G. D. Sharma, G. Talan, M. Srivastava, A. Yadav, and R. Chopra, "A qualitative enquiry into strategic and operational responses to Covid-19 challenges in South Asia," *Journal of Public Affairs*, Article ID e2195, 2020.

[5] J. G. Júnior, J. P. de Sales, M. M. Moreira, W. R. Pinheiro, C. K. T. Lima, and M. L. R. Neto, "A crisis within the crisis: the mental health situation of refugees in the world during the 2019 coronavirus (2019-nCoV) outbreak," *Psychiatry Research*, vol. 288, 2020.

[6] The Lancet, "Avoiding panic in a pandemic," *The Lancet*, vol. 373, no. 9681, p. 2084, 2009.

[7] World Health Organization, *Mental Health and Psychosocial Considerations during COVID-19 Outbreak*, pp. 1–6, World Health Organization, Geneva, Switzerland, 2020.

[8] B. Erkut, T. Kaya, M. Lehmann-Waffenschmidt et al., "A fresh look on financial decision-making from the plasticity perspective," *International Journal of Ethics and Systems*, vol. 34, no. 4, pp. 426–441, 2018.

[9] M. Göçken, M. Özçalıcı, A. Boru, and A. T. Dosdoğru, "Integrating metaheuristics and artificial neural networks for improved stock price prediction," *Expert Systems with Applications*, vol. 44, pp. 320–331, 2016.

[10] M. Bildirici and O. Ersin, "Modeling Markov switching ARMA-GARCH neural networks models and an application to forecasting stock returns," *Scientific World Journal*, 2014.

[11] M. A. Boyacioglu and D. Avci, "An adaptive network-based fuzzy inference system (ANFIS) for the prediction of stock market return: the case of the Istanbul stock exchange," *Expert Systems with Applications*, vol. 37, no. 12, pp. 7908–7912, 2010.

[12] I. M. Hamed, A. S. Hussein, and M. F. Tolba, "An intelligent model for stock market prediction," *International Journal of Computational Intelligence Systems*, vol. 5, no. 4, pp. 639–652, 2012.

[13] G. Ruxanda and L. M. Badea, "Configuring artificial neural networks for stock market predictions," *Technological and Economic Development of Economy*, vol. 20, no. 1, pp. 116–132, 2014.

[14] Z. Yudong and W. Lenan, "Stock market prediction of S&P 500 via combination of improved BCO approach and BP neural network," *Expert Systems with Applications*, vol. 36, no. 5, pp. 8849–8854, 2009.

[15] P. P. Bonissone, "Soft computing: the convergence of emerging reasoning technologies," *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, vol. 1, no. 1, pp. 6–18, 1997.

[16] O. Akbilgic, H. Bozdogan, and M. E. Balaban, "A novel Hybrid RBF neural networks model as a forecaster," *Statistics and Computing*, vol. 24, no. 3, pp. 365–375, 2014.

[17] A. Arango and J. D. Velasquez, "Forecasting the Colombian exchange market index (IGBC) using neural networks," *IEEE Latin America Transactions*, vol. 12, no. 4, pp. 718–724, 2014.

[18] G. Armano, A. Murru, and F. Roli, "Stock market prediction by a mixture of genetic-neural experts," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 16, no. 05, pp. 501–526, 2002.

[19] Q. Cao, K. B. Leggio, and M. J. Schniederjans, "A comparison between Fama and French's model and artificial neural networks in predicting the Chinese stock market," *Computers & Operations Research*, vol. 32, no. 10, pp. 2499–2512, 2005.

[20] S. Chakravarty and P. K. Dash, "A PSO based integrated functional link net and interval type-2 fuzzy logic system for predicting stock market indices," *Applied Soft Computing*, vol. 12, no. 2, pp. 931–941, 2012.

[21] J. F. Chang, L. Y. Wei, and C. H. Cheng, "Anfis-based adaptive expectation model for forecasting stock index," *International Journal of Innovative Computing Information and Control*, vol. 5, no. 7, pp. 1949–1958, 2009.

[22] W. Dai, J.-Y. Wu, and C.-J. Lu, "Combining nonlinear independent component analysis and neural network for the prediction of Asian stock market indexes," *Expert Systems with Applications*, vol. 39, no. 4, pp. 4444–4452, 2012.

[23] Á. Dávila, N. Sanchez-Choez, and J. L. Román-Vásquez, "Pronóstico del índice bursátil ecuatoriano (ecuindex) mediante redes neuronales autorregresivas," *3C Empresa :Investigación y Pensamiento Crítico*, vol. 6, no. 3, pp. 16–32, 2017.

[24] W.-Q. Duan and H. E. Stanley, "Cross-correlation and the predictability of financial return series," *Physica A: Statistical Mechanics and Its Applications*, vol. 390, no. 2, pp. 290–296, 2011.

[25] K. E. Fish, J. D. Johnson, R. E. Dorsey, and J. G. Blodgett, "Using an artificial neural network trained with a genetic algorithm to model brand share," *Journal of Business Research*, vol. 57, no. 1, pp. 79–85, 2004.

[26] V. Georgescu, "Robustly forecasting the bucharest stock exchange bet index through a novel computational intelligence approach," *Economic Computation and Economic Cybernetics Studies and Research*, vol. 44, no. 3, pp. 23–42, 2010.

[27] C. M. Hsu, "A hybrid procedure for stock price prediction by integrating self-organizing map and genetic programming," *Expert Systems with Applications*, vol. 38, no. 11, pp. 14026–14036, 2011.

[28] H.-j. Kim and K.-s. Shin, "A hybrid approach based on neural networks and genetic algorithms for detecting temporal patterns in stock markets," *Applied Soft Computing*, vol. 7, no. 2, pp. 569–576, 2007.

[29] K.-j. Kim and H. Ahn, "Simultaneous optimization of artificial neural networks for financial forecasting," *Applied Intelligence*, vol. 36, no. 4, pp. 887–898, 2012.

[30] H. F. Li, "Price forecasting of stock index futures based on a new hybrid EMD-RBF neural network model," *Agro Food Industry Hi-Tech*, vol. 28, no. 1, pp. 1744–1747, 2017.

[31] C.-J. Lu, "Integrating independent component analysis-based denoising scheme with neural network for stock price prediction," *Expert Systems with Applications*, vol. 37, no. 10, pp. 7056–7064, 2010.

[32] D. Ormoneit, "A regularization approach to continuous learning with an application to financial derivatives pricing," *Neural Networks*, vol. 12, no. 10, pp. 1405–1412, 1999.

[33] A. Parisi, F. Parisi, and J. L. Guerrero, "Neural network models for forecasting stock market indices," *Trimestre Economico*, vol. 70, no. 280, pp. 721–744, 2003.

[34] S. Pyo, J. Lee, M. Cha, and H. Jang, "Predictability of machine learning techniques to forecast the trends of market index prices: hypothesis testing for the Korean stock markets," *PLoS One*, vol. 12, no. 11, 2017.

[35] C. O. Ribeiro and S. M. Oliveira, "A hybrid commodity price-forecasting model applied to the sugar-alcohol sector," *Australian Journal of Agricultural and Resource Economics*, vol. 55, no. 2, pp. 180–198, 2011.

[36] S. Safi and A. White, "Short and long-term forecasting using artificial neural networks for stock prices in Palestine: a comparative study," *Electronic Journal of Applied Statistical Analysis*, vol. 10, no. 1, pp. 14–28, 2017.

[37] C.-H. Su, T.-L. Chen, C.-H. Cheng, and Y.-C. Chen, "Forecasting the stock market with linguistic rules generated from the minimize entropy principle and the cumulative probability distribution approaches," *Entropy*, vol. 12, no. 12, pp. 2397–2417, 2010.

[38] A. Tan, C. Quek, and K. C. Yow, "Maximizing winning trades using a novel RSPOP fuzzy neural network intelligent stock trading system," *Applied Intelligence*, vol. 29, no. 2, pp. 116–128, 2008.

[39] J. L. Ticknor, "A Bayesian regularized artificial neural network for stock market forecasting," *Expert Systems with Applications*, vol. 40, no. 14, pp. 5501–5506, 2013.

[40] C.-F. Tsai and Y.-J. Chiou, "Earnings management prediction: a pilot study of combining neural networks and decision trees," *Expert Systems with Applications*, vol. 36, no. 3, pp. 7183–7191, 2009.

[41] X. D. Wu, M. Fung, and A. Flitman, "Forecasting stock market performance using hybrid intelligent system," in *Computational Science -- Iccs 200, Proceedings Pt 2*, vol. 2074, pp. 447–456, Springer-Verlag Berlin, Berlin, Germany, 2001.

[42] S. X. Xu and M. Zhang, "An adaptive activation function for higher order neural networks," in *Al 2002: Advances in Artificial Intelligence*, vol. 2557, pp. 356–362, Springer-Verlag Berlin, Berlin, Germany, 2002.

[43] C.-H. Yeh and C.-Y. Yang, "Examining the effectiveness of price limits in an artificial stock market," *Journal of Economic Dynamics and Control*, vol. 34, no. 10, pp. 2089–2108, 2010.

[44] J. Yim, "A comparison of neural networks with time series models for forecasting returns on a stock market index," in *Developments in Applied Artificail Intelligence, Proceedings*, vol. 2358, pp. 25–35, Springer-Verlag Berlin, Berlin, Germany, 2002.

[45] X. Zhu, H. Wang, L. Xu, and H. Li, "Predicting stock index increments by neural networks: the role of trading volume under different horizons," *Expert Systems with Applications*, vol. 34, no. 4, pp. 3043–3054, 2008.

[46] A. M. Rather, V. N. Sastry, and A. Agarwal, "Stock market prediction and Portfolio selection models: a survey," *Opsearch*, vol. 54, no. 3, pp. 558–579, 2017.

[47] O. Chatrabgoun, A. Hosseinian-Far, V. Chang, N. G. Stocks, and A. Daneshkhah, "Approximating non-Gaussian Bayesian networks using minimum information vine model with applications in financial modelling," *Journal of Computational Science*, vol. 24, pp. 266–276, 2018.

[48] S. Emerson, R. Kennedy, L. O. Shea, and J. O. Brien, "Trends and applications of machine learning in quantitative finance," in *Proceedings of the 8th International Conference on Economics and Finance Research*, Lyon, France, June 2019.

[49] C. H. Fajardo-Toro, J. Mula, and R. Poler, "Adaptive and hybrid forecasting models—a review," in *Engineering Digital Transformation*, pp. 315–322, Springer, Berlin, Germany, 2018.

[50] T. Kaya and B. Erkut, "Tacit knowledge for strategic advantage: social media use of employees in the financial sector," in *Proceedings of the 18th European Conference on Knowledge Management (ECKM 2017)*, pp. 516–523, Barcelona, Spain, September 2017.

[51] T. Poddig and H. Rehkugler, "A "world" model of integrated financial markets using artificial neural networks," *Neurocomputing*, vol. 10, no. 3, pp. 251–273, 1996.

[52] A. Parisi and F. Parisi, "Predictive models of logic and fuzzy logic in stock market index of the United States," *Trimestre Economico*, vol. 73, no. 290, pp. 265–288, 2006.

[53] N. O'Connor and M. G. Madden, "A neural network approach to predicting stock exchange movements using external factors," *Knowledge-Based Systems*, vol. 19, no. 5, pp. 371–378, 2006.

[54] M. F. Abbod, F. C. Hamdy, D. A. Linkens, and J. W. Catto, "Predictive modeling in cancer: where systems biology meets the stock market," *Expert Review of Anticancer Therapy*, vol. 9, no. 7, pp. 867–870, 2009.

[55] Z. Liao and J. Wang, "Forecasting model of global stock index by stochastic time effective neural network," *Expert Systems with Applications*, vol. 37, no. 1, pp. 834–841, 2010.

[56] F. Liu and J. Wang, "Fluctuation prediction of stock market index by Legendre neural network with random time strength function," *Neuro Computing*, vol. 83, pp. 12–21, 2012.

[57] H. Y. Mo and J. Wang, "Volatility degree forecasting of stock market by stochastic time strength neural network," *Mathematical Problems in Engineering*, vol. 2013, Article ID 436795, 2013.

[58] H. V. Pham, T. Cao, I. Nakaoka, E. W. Cooper, and K. Kamei, "A proposal of hybrid kansei-som model for stock market investment," *International Journal of Innovative Computing Information and Control*, vol. 7, no. 5B, pp. 2863–2880, 2011.

[59] M. Qiu, Y. Song, and F. Akagi, "Application of artificial neural network for the prediction of stock market returns: the case of the Japanese stock market," *Chaos, Solitons & Fractals*, vol. 85, pp. 1–7, 2016.

[60] Q. A. Chen and C. D. Li, "Comparison of forecasting performance of AR, STAR and ANN models on the Chinese stock market index," in *Advances in Neural Networks - Isnn 2006, Pt 3, Proceedings*, vol. 3973, pp. 464–470, Springer-Verlag Berlin, Berlin, Germany, 2006.

[61] Y. K. Bao, Y. S. Lu, and J. L. Zhang, "Forecasting stock price by SVMs regression," in *Artificial Intelligence: Methodology, Systems, and Applications, Proceedings*, vol. 3192, pp. 295–303, Springer-Verlag Berlin, Berlin, Germany, 2004.

[62] H. F. Liu and J. Wang, "Integrating independent component analysis and principal component analysis with neural network to predict Chinese stock market," *Mathematical Problems in Engineering*, vol. 2019, 2011.

[63] A. J. Lin, P. J. Shang, G. C. Feng, and B. Zhong, "Application of empirical mode decomposition combined with k-nearest neighbors approach in financial time series forecasting," *Fluctuation and Noise Letters*, vol. 11, no. 2, 2012.

[64] J. Wang and J. Wang, "Forecasting stochastic neural network based on financial empirical mode decomposition," *Neural Networks*, vol. 90, pp. 8–20, 2017.

[65] X. Liang, "Neural network method to predict stock price movement based on stock information entropy," in *Advances in Neural Networks-Isnn 2006, Pt 3, Proceedings*, vol. 3973, pp. 442–451, Springer-Verlag Berlin, Berlin, Germany, 2006.

[66] N. Li, X. Liang, X. Li, C. Wang, and D. D. Wu, "Network environment and financial risk using machine learning and sentiment analysis," *Human and Ecological Risk Assessment: An International Journal*, vol. 15, no. 2, pp. 227–252, 2009.

[67] X. Liang, "Impacts of Internet stock news on stock markets based on neural networks," in *Advances in Neural Networks - Isnn 2005, Pt 2, Proceedings*, vol. 3497, pp. 897–903, Springer-Verlag Berlin, Berlin, Germany, 2005.

[68] H. Rehkugler and T. Poddig, "Artificial neural networks in financial analysis-A new approach to forecast movements of stocks, indexes and currencies," *Wirtschaftsinformatik*, vol. 33, no. 5, pp. 365–374, 1991.

[69] B. Freisleben, "Stock-market prediction with backpropagation networks," *Lecture Notes in Artificial Intelligence*, vol. 604, pp. 451–460, 1992.

[70] D. W. Salt, N. Yildiz, D. J. Livingstone, and C. J. Tinsley, "The use of artificial neural networks in Qsar," *Pesticide Science*, vol. 36, no. 2, pp. 161–170, 1992.

[71] Y. Yoon, G. Swales, and T. M. Margavio, "A comparison of discriminant analysis versus artificial neural networks," *The Journal of the Operational Research Society*, vol. 44, no. 1, pp. 51–60, 1993.

[72] H. J. Kim, K. S. Shin, and K. Park, "Time delay neural networks and genetic algorithms for detecting temporal patterns in stock markets," in *Advances in Natural Computation, Pt 1, Proceedings*, vol. 3610, pp. 1247–1255, Springer-Verlag Berlin, Berlin, Germany, 2005.

[73] M. Khashei, S. Reza Hejazi, and M. Bijari, "A new hybrid artificial neural networks and fuzzy regression model for time series forecasting," *Fuzzy Sets and Systems*, vol. 159, no. 7, pp. 769–786, 2008.

[74] A. Bohn, T. Güting, T. Mansmann, and S. Selle, "MoneyBee: aktienkursprognose mit künstlicher intelligenz bei hoher rechenleistung," *Wirtschaftsinformatik*, vol. 45, no. 3, pp. 325–333, 2003.

[75] L. Zhang, N. Liu, and P. Yu, "A novel instantaneous frequency algorithm and its application in stock index movement prediction," *IEEE Journal of Selected Topics in Signal Processing*, vol. 6, no. 4, pp. 311–318, 2012.

[76] O. Haroon and S. A. R. Rizvi, "COVID-19: media coverage and financial markets behavior-A sectoral inquiry," *Journal of Behavioral and Experimental Finance*, vol. 27, Article ID 100343, 2020.

[77] M. C. R. de C. Ferreira, V. A. Sobreiro, H. Kimura, and F. L. de M. Barboza, "A systematic review of literature about finance and sustainability," *Journal of Sustainable Finance & Investment ISSN*, vol. 6, no. 2, pp. 112–147, 2016.

[78] C. J. C. Jabbour, "Environmental training in organisations: from a literature review to a framework for future research," *Resources, Conservation and Recycling*, vol. 74, pp. 144–155, 2013.

[79] M. Lage Junior and M. Godinho Filho, "Variations of the kanban system: literature review and classification," *International Journal of Production Economics*, vol. 125, no. 1, pp. 13–21, 2010.

[80] S. Seuring, "A review of modeling approaches for sustainable supply chain management," *Decision Support Systems*, vol. 54, no. 4, pp. 1513–1520, 2013.

[81] M. J. Rezaee, M. Jozmaleki, and M. Valipour, "Integrating dynamic fuzzy C-means, data envelopment analysis and artificial neural network to online prediction performance of companies in stock exchange," *Physica A-Statistical Mechanics and Its Applications*, vol. 489, pp. 78–93, 2018.

[82] A. K. Rout, P. K. Dash, R. Dash, and R. Bisoi, "Forecasting financial time series using a low complexity recurrent neural network and evolutionary learning approach," *Journal of King Saud University-Computer and Information Sciences*, vol. 29, no. 4, pp. 536–552, 2017.

[83] B. Wu and T. Duan, "A performance comparison of neural networks in forecasting stock price trend," *International Journal of Computational Intelligence Systems*, vol. 10, no. 1, pp. 336–346, 2017.

[84] P. Nystrup, H. Madsen, and E. Lindström, "Dynamic portfolio optimization across hidden market regimes," *Quantitative Finance*, vol. 18, pp. 1–13, 2017.

[85] Z. Tracking, "Economic model predictive control with zone tracking," *Mathematics*, vol. 6, no. 65, pp. 1–19, 2018.

[86] F. Herzog, G. Dondi, and H. P. Geering, "Stochastic model predictive control and portfolio optimization," *International Journal of Theoretical and Applied Finance*, vol. 10, no. 02, pp. 203–233, 2007.

[87] Y. Yamada and J. A. Primbs, “Model predictive control for optimal pairs trading portfolio with gross exposure and transaction cost constraints,” *Asia-Pacific Financial Markets*, 2017.

[88] V. Dombrovskii and T. Obedko, “Feedback predictive control strategies for investment in the financial market with serially correlated returns subject to constraints and trading costs,” *Optimal Control Applied Methodology*, pp. 1–14, 2017.

[89] A. Mochón, D. Quintana, Y. Sáez, and P. Isasi, “Soft computing techniques applied to finance,” *Applied Intelligence*, vol. 29, no. 2, pp. 111–115, 2008.

[90] L. A. Zadeh, “Fuzzy logic, neural networks, and soft computing,” *Communications of the ACM*, vol. 37, no. 3, pp. 77–84, 1994.

[91] S. P. Das and S. Padhy, “A new hybrid parametric and machine learning model with homogeneity hint for European-style index option pricing,” *Neural Computing and Applications*, vol. 28, no. 12, pp. 4061–4077, 2017.

[92] C. Jayne, A. Lanitis, and C. Christodoulou, “Neural network methods for one-to-many multi-valued mapping problems,” *Neural Computing and Applications*, vol. 20, no. 6, pp. 775–785, 2011.

[93] K.-j. Kim and W. B. Lee, “Stock market prediction using artificial neural networks with optimal feature transformation,” *Neural Computing and Applications*, vol. 13, no. 3, pp. 255–260, 2004.

[94] Z. G. Qun, L. Y. Xu, and G. W. Zhang, “LSTM neural network with emotional analysis for prediction of stock price,” *Engineering Letters*, vol. 25, no. 2, pp. 167–175, 2017.

[95] S. C. Nayak, B. B. Misra, and H. S. Behera, “Artificial chemical reaction optimization of neural networks for efficient prediction of stock market indices,” *Ain Shams Engineering Journal*, vol. 8, no. 3, pp. 371–390, 2017.

[96] A. Bahrammirzaee, “A comparative survey of artificial intelligence applications in finance: artificial neural networks, expert system and hybrid intelligent systems,” *Neural Computing and Applications*, vol. 19, pp. 1165–1196, 2010.

[97] Y. Li, C. Wu, J. Liu, and P. Luo, “A combination prediction model of stock composite index based on artificial intelligent methods and multi-agent simulation,” *International Journal of Computational Intelligence Systems*, vol. 7, no. 5, pp. 853–864, 2014.

[98] M. Inthachot, V. Boonjing, and S. Intakosum, “Artificial Neural Network and genetic algorithm hybrid intelligence for predicting Thai stock price index trend.pdf,” *Computational Intelligence and Neuroscience*, 2016.

[99] D. K. Bebarta, B. Biswal, and P. K. Dash, “Polynomial based functional link artificial recurrent neural network adaptive system for predicting Indian stocks,” *International Journal of Computational Intelligence Systems*, vol. 8, no. 6, pp. 1004–1016, 2015.

[100] K. Menger-Anderson, “The ethics of digitally networked beings–towards data science,” 2018.

[101] A. Ghandar, Z. Michalewicz, M. Schmidt, T.-D. To, and R. Zurbrugg, “Computational intelligence for evolving trading rules,” *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 1, pp. 71–86, 2009.

[102] R. Naranjo, J. Arroyo, and M. Santos, “Fuzzy modeling of stock trading with fuzzy candlesticks,” *Expert Systems with Applications*, vol. 93, pp. 15–27, 2018.

[103] L. Y. Wei, T. L. Chen, and T. H. Ho, “A hybrid model based on adaptive-network-based fuzzy inference system to forecast Taiwan stock market,” *Expert Systems with Applications*, vol. 38, no. 11, pp. 13625–13631, 2011.

[104] K. K. Ang and C. Quek, “Stock trading using RSPOP: a novel rough set-based neuro-fuzzy approach,” *IEEE Transactions on Neural Networks*, vol. 17, no. 5, pp. 1301–1315, 2006.

[105] S. D. Bekiros, “Sign prediction and volatility dynamics with hybrid neurofuzzy approaches,” *IEEE Transactions on Neural Networks*, vol. 22, no. 12, pp. 2353–2362, 2011.

[106] J. Komlos and B. Suessmuth, *Empirische Ökonomie-Eine Einführung in Methoden und Anwendungen*, Springer, Berlin, Germany, 2010, https://www.springer.com/de/book/9783642017049.

[107] A. Karathanasopoulos, “Modelling and trading the English stock market with novelty optimization techniques,” *Economics and Business Letters*, vol. 5, no. 2, pp. 50–57, 2016.

[108] R. Y. M. Li, S. Fong, and K. W. S. Chong, “Forecasting the REITs and stock indices: group method of data handling neural network approach,” *Pacific Rim Property Research Journal*, vol. 23, no. 2, pp. 123–160, 2017.

[109] X. Zhong and D. Enke, “A comprehensive cluster and classification mining procedure for daily stock market return forecasting,” *Neurocomputing*, vol. 267, pp. 152–168, 2017.

[110] P. C. Chang, C. Y. Fan, and C. H. Liu, “Integrating a piecewise linear representation method and a neural network model for stock trading points prediction,” *IEEE Transactions on Systems Man and Cybernetics Part C-Applications and Reviews*, vol. 39, no. 1, pp. 80–92, 2009.

[111] M. Kheradmandi and P. Mhaskar, “Data driven economic model predictive control,” *Mathematics*, vol. 6, no. 51, pp. 1–17, 2018.

[112] T. Trimborn, L. Pareschi, and M. Frank, “Portfolio optimization and model predictive control: a kinetic approach,” *Discrete & Continuous Dynamical Systems - B*, vol. 24, no. 11, pp. 6209–6238, 2019.

[113] D. Zantedeschi, P. Damien, and N. G. Polson, “Predictive macro-finance with dynamic partition models,” *Journal of the American Statistical Association*, vol. 106, no. 494, pp. 427–439, 2011.

[114] L. Sanzogni, G. Guzman, and P. Busch, “Artificial intelligence and knowledge management: questioning the tacit dimension,” *Prometheus*, vol. 35, no. 1, pp. 37–56, 2017.

[115] M. Dong and X.-S. Zhou, “Knowledge discovery in corporate events by neural network rule extraction,” *Applied Intelligence*, vol. 29, no. 2, pp. 129–137, 2008.

[116] C.-H. Liu, J. S. Wang, and C.-W. Lin, “The concepts of big data applied in personal knowledge management,” *Journal of Knowledge Management*, vol. 21, no. 1, pp. 213–230, 2017.

[117] B. Erkut, “From digital government to digital governance: are we there yet?” *Sustainability*, vol. 12, no. 3, 2020.

[118] S. I. Ao, “Automating stock prediction with neural network and evolutionary computation,” in *Intelligent Data Engineering and Automated Learning*, vol. 2690, pp. 203–210, Springer-Verlag Berlin, Berlin, Germany, 2003.

[119] Q. Cao, M. E. Parry, and K. B. Leggio, “The three-factor model and artificial neural networks: predicting stock price movement in China,” *Annals of Operations Research*, vol. 185, no. 1, pp. 25–44, 2011.

[120] A.-S. Chen, M. T. Leung, and H. Daouk, “Application of neural networks to an emerging financial market: forecasting and trading the Taiwan Stock Index,” *Computers & Operations Research*, vol. 30, no. 6, pp. 901–923, 2003.

[121] E. Constantinou, R. Georgiades, A. Kazandjian, and G. P. Kouretas, “Regime switching and artificial neural network forecasting of the cyprus stock exchange daily returns,” *International Journal of Finance & Economics*, vol. 11, no. 4, pp. 371–383, 2006.

[122] J. H. Ma and L. X. Liu, "Multivariate nonlinear analysis and prediction of Shanghai stock market," *Discrete Dynamics in Nature and Society*, vol. 2008, Article ID 526734, 2008.

[123] M. M. Mostafa, "Forecasting stock exchange movements using neural networks: empirical evidence from Kuwait," *Expert Systems with Applications*, vol. 37, no. 9, pp. 6302–6309, 2010.

[124] K. J. Oh, T. Y. Kim, H. Y. Lee, and H. Lee, "Using neural networks to support early warning system for financial crisis forecasting," in *Ai 2005: Advances in Artificial Intelligence*, vol. 3809, pp. 284–296, Springer-Verlag Berlin, Berlin, Germany, 2005.

[125] H. S. Basavegowda, G. Dagnew, and G. Dagnew, "Deep learning approach for microarray cancer data classification," *CAAI Transactions on Intelligence Technology*, vol. 5, no. 1, pp. 22–33, 2020.

[126] N. N. Das, K. Naresh, K. Manjit, K. Vijay, and S. Dilbag, "Automated deep transfer learning-based approach for detection of COVID-19 infection in chest X-rays," *Irbm*, 2020.

[127] A. Jaiswal, N. Gianchandani, D. Singh, V. Kumar, and M. Kaur, "Classification of the COVID-19 infected patients using DenseNet201 based deep transfer learning," *Journal of Biomolecular Structure and Dynamics*, pp. 1–8, 2020.

[128] S. Osterland and J. Weber, "Analytical analysis of single-stage pressure relief valves," *International Journal of Hydromechatronics*, vol. 2, no. 1, pp. 32–53, 2019.

[129] R. Wang, H. Yu, G. Wang, G. Zhang, and W. Wang, "Study on the dynamic and static characteristics of gas static thrust bearing with micro-hole restrictors," *International Journal of Hydromechatronics*, vol. 2, no. 3, pp. 189–202, 2019.

[130] T. Wiens, "Engine speed reduction for hydraulic machinery using predictive algorithms," *International Journal of Hydromechatronics*, vol. 2, no. 1, pp. 16–31, 2019.

[131] M. H. F. Zarandi, B. Rezaee, I. B. Turksen, and E. Neshat, "A type-2 fuzzy rule-based expert system model for stock price analysis," *Expert Systems with Applications*, vol. 36, no. 1, pp. 139–154, 2009.

[132] A. Azadeh, M. Aryaee, M. Zarrin, and M. Saberi, "A novel performance measurement approach based on trust context using fuzzy T-norm and S-norm operators: the case study of energy consumption," *Energy Exploration & Exploitation*, vol. 34, no. 4, pp. 561–585, 2016.

[133] D. Singh, V. Kumar, Vaishali, and M. Kaur, "Classification of COVID-19 patients from chest CT images using multi-objective differential evolution-based convolutional neural networks," *European Journal of Clinical Microbiology & Infectious Diseases*, vol. 39, no. 7, pp. 1379–1389, 2020.

[134] S. Ghosh, P. Shivakumara, P. Roy, U. Pal, and T. Lu, "Graphology based handwritten character analysis for human behaviour identification," *CAAI Transactions on Intelligence Technology*, vol. 5, no. 1, pp. 55–65, 2020.

[135] A. Girdhar, H. Kapur, V. Kumar, M. Kaur, D. Singh, and R. Damasevicius, "Effect of COVID-19 outbreak on urban health and environment," *Air Quality, Atmosphere & Health*, pp. 1–9, 2020.

[136] B. Gupta, M. Tiwari, and S. Singh Lamba, "Visibility improvement and mass segmentation of mammogram images using quantile separated histogram equalisation with local contrast enhancement," *CAAI Transactions on Intelligence Technology*, vol. 4, no. 2, pp. 73–79, 2019.

[137] R. Singh and S. Srivastava, "Stock prediction using deep learning," *Multimedia Tools and Applications*, vol. 76, no. 18, pp. 18569–18584, 2017.

[138] M. Alizadeh, R. Rada, F. Jolai, and E. Fotoohi, "An adaptive neuro-fuzzy system for stock portfolio analysis," *International Journal of Intelligent Systems*, vol. 26, no. 2, pp. 99–114, 2011.

[139] G. S. Atsalakis, E. M. Dimitrakakis, and C. D. Zopounidis, "Elliott Wave Theory and neuro-fuzzy systems, in stock market prediction: the WASP system," *Expert Systems with Applications*, vol. 38, no. 8, pp. 9196–9206, 2011.

[140] A. Keles, M. Kolcak, and A. Keles, "The adaptive neuro-fuzzy model for forecasting the domestic debt," *Knowledge-Based Systems*, vol. 21, no. 8, pp. 951–957, 2008.

[141] L. Y. Wei, "An expanded Adaptive Neuro-Fuzzy Inference System (ANFIS) model based on AR and causality of multi-nation stock market volatility for TAIEX forecasting," *African Journal of Business Management*, vol. 5, no. 15, pp. 6377–6387, 2011.

[142] G. Armano, M. Marchesi, and A. Murru, "A hybrid genetic-neural architecture for stock indexes forecasting," *Information Sciences*, vol. 170, no. 1, pp. 3–33, 2005.

[143] Q. Cao and M. E. Parry, "Neural network earnings per share forecasting models: a comparison of backward propagation and the genetic algorithm," *Decision Support Systems*, vol. 47, no. 1, pp. 32–41, 2009.

[144] C.-H. Cheng, T.-L. Chen, and L.-Y. Wei, "A hybrid model based on rough sets theory and genetic algorithms for stock price forecasting," *Information Sciences*, vol. 180, no. 9, pp. 1610–1629, 2010.

[145] A. Esfahanipour and S. Mousavi, "A genetic programming model to generate risk-adjusted technical trading rules in stock markets," *Expert Systems with Applications*, vol. 38, no. 7, pp. 8438–8445, 2011.

[146] Y. K. Kwon and B. R. Moon, "A hybrid neurogenetic approach for stock forecasting," *IEEE Transactions on Neural Networks*, vol. 18, no. 3, pp. 851–864, 2007.

[147] J. Mandziuk and M. Jaruszewicz, "Neuro-genetic system for stock index prediction," *Journal of Intelligent & Fuzzy Systems*, vol. 22, no. 2–3, pp. 93–123, 2011.

[148] T. Z. Tan, C. Quek, and G. S. Ng, "Biological brain-inspired genetic complementary learning for stock market and bank failure prediction," *Computational Intelligence*, vol. 23, no. 2, pp. 236–261, 2007.

[149] V. S. Desai and R. Bharati, "The efficacy of neural networks in predicting returns on stock and bond indices," *Decision Sciences*, vol. 29, no. 2, pp. 405–423, 1998.

[150] S. Amornwattana, D. Enke, and C. H. Dagli, "A hybrid option pricing model using a neural network for estimating volatility," *International Journal of General Systems*, vol. 36, no. 5, pp. 558–573, 2007.

[151] W. Huang, S. Y. Wang, L. Yu, Y. K. Bao, and L. Wang, "A new computational method of input selection for stock market forecasting with neural networks," in *Computational Science - Iccs 2006, Pt 4, Proceedings*, vol. 3994, pp. 308–315, Springer-Verlag Berlin, Berlin, Germany, 2006.

[152] M. Kohler and A. Krzyżak, "Pricing of American options in discrete time using least squares estimates with complexity penalties," *Journal of Statistical Planning and Inference*, vol. 142, no. 8, pp. 2289–2307, 2012.

[153] P. R. Lajbcygier and J. T. Connor, "Improved option pricing using artificial neural networks and bootstrap methods," *International Journal of Neural Systems*, vol. 08, no. 04, pp. 457–471, 1997.

[154] H. Ince and T. B. Trafalis, "Kernel principal component analysis and support vector machines for stock price prediction," *Iie Transactions*, vol. 39, no. 6, pp. 629–637, 2007.

[155] D. Olson and C. Mossman, "Neural network forecasts of Canadian stock returns using accounting ratios," *International Journal of Forecasting*, vol. 19, no. 3, pp. 453–465, 2003.

[156] J. Patel, S. Shah, P. Thakkar, and K. Kotecha, "Predicting stock and stock price index movement using Trend Deterministic Data Preparation and machine learning techniques," *Expert Systems with Applications*, vol. 42, no. 1, pp. 259–268, 2015.

[157] A. M. Safer, "The application of neural networks to predict abnormal stock returns using insider trading data," *Applied Stochastic Models in Business and Industry*, vol. 18, no. 4, pp. 381–389, 2002.

[158] I. Yildirim, S. Ozsahin, and K. C. Akyuz, "Prediction of the financial return of the paper sector with artificial neural networks," *Bioresources*, vol. 6, no. 4, pp. 4076–4091, 2011.

[159] S. Yümlü, F. S. Gürgen, and N. Okay, "A comparison of global, recurrent and smoothed-piecewise neural models for Istanbul stock exchange (ISE) prediction," *Pattern Recognition Letters*, vol. 26, no. 13, pp. 2093–2103, 2005.

Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 8503454, 14 pages
https://doi.org/10.1155/2020/8503454

Research Article

Multipopulation Genetic Algorithms with Different Interaction Structures to Solve Flexible Job-Shop Scheduling Problems: A Network Science Perspective

Ding-Shan Deng,[1] **Wei Long**,[1] **Yan-Yan Li**,[1] **and Xiao-Qiu Shi**[2]

[1]*School of Mechanical Engineering, Sichuan University, Chengdu 610065, China*
[2]*School of Manufacturing Science and Engineering, Southwest University of Science and Technology, Mianyang 621000, China*

Correspondence should be addressed to Yan-Yan Li; yyl_scu@163.com

Received 22 April 2020; Revised 1 July 2020; Accepted 24 September 2020; Published 27 November 2020

Academic Editor: Manjit Kaur

Populations of multipopulation genetic algorithms (MPGAs) parallely evolve with some interaction mechanisms. Previous studies have shown that the interaction structures can impact on the performance of MPGAs to some extent. This paper introduces the concept of complex networks such as ring-shaped networks and small-world networks to study how interaction structures and their parameters influence the MPGAs, where subpopulations are regarded as nodes and their interaction or migration of elites between subpopulations as edges. After solving the flexible job-shop scheduling problem (FJSP) by MPGAs with different parameters of interaction structures, simulation results were measured by criteria, such as success rate and average optimal value. The analysis reveals that (1) the smaller the average path length (APL) of the network is, the higher the propagation rate will be; (2) the performance of MPGAs increased first and then decreased along with the decrease of APL, indicating that, for better performance, the networks should have a proper APL, which can be adjusted by changing the structural parameters of networks; and (3) because the edge number of small-world networks remains unchanged with different rewiring possibilities of edges, the change in performance indicates that the MPGA can be improved by a more proper interaction structure of subpopulations as other conditions remain unchanged.

1. Introduction

Genetic algorithm (GA) [1], an original metaheuristic, is easy to fall into local optima when employed to solve resource-constrained project scheduling problems [2] such as flexible job-shop schedule problems (FJSPs) due to the complexity of searched space and high dimensions [3].

To maintain population diversity (enhancing the search diversity) and avoid premature convergence, a multipopulation genetic algorithm (MPGA) [4] is one feasible method where subpopulations are generated and individuals migrate periodically among them. In the evolutionary process, when intra-subpopulation evolution pushes individuals towards different local optima, migration can introduce new genes into the subpopulations [5]. As in [5, 6], the migration can be implemented in different topologies that define how subpopulations are connected, which can influence the outcomes of MPGAs to some extent.

To more effectively research on the performance with different interaction structures between subpopulations, we can consider subpopulations as nodes and their interaction or migration of elites between subpopulations as edges so that MPGA can be regarded as complex networks. Similarly, but not identically, Du et al. [7] proposed the networked evolutionary algorithm where nodes represent information process units, i.e., individuals, and connections denote information transmission links. Payne et al. [8] shed light on dynamic population structures, wherein edges are dynamically rewired according to the genotypic or phenotypic properties of individuals or according to the success of prior interindividual interaction. Chao et al. [9, 10] proposed a multiobjective cellular grey wolf optimizer with a topological structure for hybrid flow-shop scheduling, in which each wolf or individual can be regarded as a grid of a lattice structure, and the interaction among them is restricted to the neighborhood. In the aforementioned three papers, the nodes of networks were regarded as individuals; however, this paper mainly discusses the interactions of subpopulations. For example, Leitao et al. [5] conducted simulations to study how the topology of island models (subpopulations) impacts the effectiveness and diversity of evolutionary algorithms with three types of networks, i.e., the ring and torus, as well as the fully connected networks. Besides, Zhang and Li [11] proposed a multiobjective evolutionary algorithm based on decomposition (MOEA/D). It can also be regarded as a multipopulation algorithm, and each subpopulation has only one individual. Fu et al. [12] extended and used the idea of MOEA/D to realistic problems, e.g., flow-shop scheduling problem, and each subpopulation contains multiple individuals. MOEA/D decomposes a multiobjective optimization problem into a number of scalar optimization subproblems, and the offspring individual to one subproblem can be generated upon the parent individuals from its neighboring subproblems. This neighborhood relation can also be treated as a ring-shaped network. Unlike MPGA discussed in this paper, the subpopulations of MOEA/D are heterogeneous, representing different subproblems, and cannot evolve independently to get an optimal solution. Additionally, other networks such as 4D hypercubes [13] and a 4×4 toroidal mesh are also utilized [4].

When it comes to scheduling problems [14, 15], MPGAs are widely used to get the optimal solution. Kimms et al. [16] introduced an MPGA as a procedure to solve the synchronized and integrated two-level lot sizing and scheduling problem; in the migration process, a copy of each best individual is inserted into the next population replacing a random individual. Therefore, the interaction of subpopulations can be abstracted as a ring-shaped network. Zandieh and Karimi [17] proposed an MPGA to search the Pareto optimal solution for a multiobjective group scheduling problem in hybrid flexible flow-shop with sequence-dependent setup times, where the interaction of subpopulations can be abstracted as fully connected networks.

We specifically choose the FJSP [18–21] as a measurement to test the performance of the MPGA as interaction structures of subpopulations vary. Research on the FJSP is essential for enterprises, especially for small-medium enterprises, to carry out production planning and scheduling in order to meet the delivery dates under a complex market [22]. For instance, Zhang et al. [23] put forward the MPGA based on the multiobjective scheduling of flexible job-shop, in which the abstracted topology of migration is a centralized network.

As summarized in Table 1, complex networks, such as scale-free networks and small-world networks, have been employed in cellular evolutionary algorithms to control interaction behaviors between individuals. However, for MPGAs, the focus of interaction structures among subpopulations is mainly rings, tori, hypercubes, and so on, and complex networks have rarely been utilized to represent subpopulations and their interaction purposefully.

Therefore, in our previous studies [24, 25], we addressed how seven different network structures, including the ring-shaped network and the small-world network, influence the propagation rate of advantageous genes and thus affect the performance of MPGA for solving the FJSP. However, only the scale of networks is discussed, and the influence of other structural parameters is not researched. Thus, in this paper, we mainly take advantage of ring-shaped networks and small-world networks [26] to study the structural influence on the performance of MPGAs to solve the FJSP because ring-shaped networks are elementary topology widely used in MPGAs intentionally or unintentionally, and small-world networks with a definite node number will possess the same edge number as the rewiring possibility of edges varies, which would make them convenient and easy to study the influence of interaction structures as other conditions remain unchanged.

Concretely, we change the inherent structural parameter of the corresponding network constantly and record the simulation results of MPGAs with the network. The success rate (SR) and average optimal value (AOV) are utilized to measure the results of MPGAs. Besides, the Hamming distance index (HDI) is introduced to evaluate the difference between elite individuals and characterize the propagation rate of advantageous genes, which provide an insight into how different interaction structures of subpopulations affect the performance of MPGAs.

The sections below are organized as follows. In Section 2, basic knowledge of complex networks and MPGA with networks for solving the FJSP, as well as the evaluation index, are introduced. Then, experiments and results analysis are reported in Section 3. Conclusions follow in Section 4.

2. Preliminaries

2.1. Network Models. A network consists of some nodes connected by some edges with a certain topology (structure) [27]. Some basic concepts utilized to characterize the network are introduced in the following. On top of that, typic models of the ring-shaped network and small-world network are elaborated.

2.1.1. Degree and Average Path Length. The concept of degree is the most fundamental character and measure of a node in a network (in this paper, networks are undirected networks), and the degree of a node is defined to be the number k_i of its existing edges. The average degree of a network is the average value of all such node degrees k_i over the entire network, denoted by ¯k.

The average distance or average path length (APL) of a network is defined to be the average value of all distances over the network:

$$\mathrm{APL} = \frac{2}{N \times (N-1)} \sum_{i<j} d_{ij}, \tag{1}$$

Table 1: Summary of background studies.

References	Networks or topologies	Denotation of nodes
[4]	Toroidal mesh networks; hypercubes	Subpopulations
[5]	Fully connected networks; rings; tori	Subpopulations
[7]	BA networks	Individuals
[8]	Scale-free networks; small-world networks	Individuals
[9, 10]	Lattices	Individuals
[11, 12]	Rings	Subpopulations
[13]	4D hypercubes	Subpopulations
[16]	Ring-shaped networks	Subpopulations
[17]	Fully connected networks	Subpopulations
[23]	Centralized networks	Subpopulations

where N is the total number of nodes in the network and d_{ij} is the distance between node i and node j, i.e., the total number of edges connecting them through the shortest route [27].

2.1.2. Ring-Shaped Networks. A network in which every node has the same degree, i.e., the same number of connecting edges, is called a regular network. A typical sparse regular network is a nearest-neighbor coupled network, where every node is connected to $2K$-nearest neighbors, K nodes on each side. Particularly, as shown in Figure 1, such a network with a periodic boundary connectivity condition is a ring-shaped network [27].

2.1.3. Small-World Networks. The so-called WS small-world networks [28] were first proposed by Watts and Strogatz in 1998, which have both features of large clustering coefficients and short average path lengths. This model is generated by rewiring the edges of a ring-shaped network one by one with probability p, in which the case of $p = 0$ corresponds to a regular network, and $p = 1$ corresponds to a kind of ER random-graph networks.

A WS small-world network can be generated by the following algorithm:

(1) Start from a ring-shaped network with N nodes, where each node is connected to its $2K\,(K > 0)$ neighbors, K nodes on each side

(2) For every pair of connected nodes in the ring-shaped network, reconnect the edge with possibility p in such a way that the beginning end of the edge is kept, but the other end is disconnected, and then rewire it to a node randomly

This operation is conducted edge by edge on the original ring-shaped network, once and once only, either clockwise or counterclockwise. Additionally, self-loop and multiple edges are avoided. Figure 2 shows an example of WS small-world networks. Since the WS small-world network is the original and most widely used topology, the term "small-world network" mentioned in this paper is always the WS small-world network.

Additionally, compared to WS small-world networks, the algorithm can be modified by replacing "random rewiring edges" with "random adding edges," resulting in the NW small-world network model [29]. However, the edge number of NW small-world networks is changed with the rewiring process, so we discard the use of NW small-world networks.

2.2. Flexible Job-Shop Scheduling Problem. FJSP, which is also introduced in [24, 30], can be divided into two subproblems: machine subproblem and operation subproblem, namely, selecting a specific machine for each operation and arranging a proper processing order of all operations. Usually, there exist a job set within n jobs, labeled as $J = \{J_i\}_{i=1}^{n}$, and a machine set within m machines, labeled as $M = \{M_k\}_{k=1}^{m}$. For each job J_i, there is an operation set within l_i operations, labeled as $O_i = \{O_{ij}\}_{j=1}^{l_i}$. Each operation O_{ij} can only be processed on a specific group of machines, labeled as $S_{ij} \subseteq M$. The goal of the scheduling problem is to assign operations with proper orders to proper machines at the proper time, to pursue some given objectives such as minimizing maximum machine workload and minimizing makespan. For simplicity without losing generality, we choose to minimize makespan as the objective. The mathematical model is as follows [25]:

$$\min F_{\max} = \max\left(F_{ij}\right), \quad \forall i, j, \tag{2}$$

s.t.

$$F_{ijk} \geq 0, \quad \forall i, j, k, \tag{3}$$

$$P_{ijk} \geq 0, \quad \forall i, j, k, \tag{4}$$

$$B_{ijk} \geq 0, \quad \forall i, j, k, \tag{5}$$

$$X_{ijk} \in \{0, 1\}, \quad \forall i, j, k, \tag{6}$$

$$i, j, l, i', j' \in \{1, 2, 3, \ldots,\}, \tag{7}$$

$$\sum_{k \in S_{ij}} X_{ijk} = 1 \wedge F_{ijk} - B_{ijk} = P_{ijk}, \quad \forall i, j, \tag{8}$$

$$F_{ij} - F_{i(j-1)} \geq P_{ijk} \cdot X_{ijk}, \quad \forall i, j, k, \tag{9}$$

$$F_{i'j'k} \leq B_{ijk} \vee F_{ijk} \leq B_{i'j'k}, \quad \forall i', j' \neq i, j. \tag{10}$$

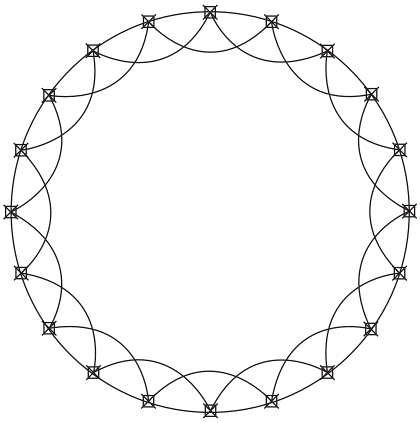


Figure 1: Ring-shaped networks.

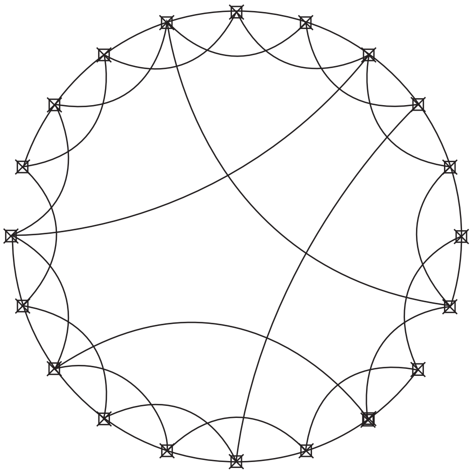


Figure 2: WS small-world networks.

Table 2 lists the notations used in the FJSP. For equation (6), as O_{ij} is arranged to be processed on M_k, $X_{ijk} = 1$; otherwise, 0. Equation (7) denotes the domains of variables. Equation (8) guarantees that each operation can only be processed at one machine without disruption, and '$\wedge$' represents logical AND. Therefore, $F_{ij} = F_{ijk}$, $\forall i, j, \exists k, k \in S_{ij}$. Equation (9) ensures each job will be processed in the correct order. Equation (10) ensures that one machine can only process one operation at the same time, and '$\vee$' represents logical OR.

Besides, Table 3 lists a four-step process of the FJSP for 2 jobs with a total of 4 operations (4×4), where '—' means the corresponding operation cannot be processed in that machine. The fourth number "19" in the first line, for example, represents that O_{11} can be processed in M_4 within time horizon 19.

2.3. MPGA with Networks for Solving the FJSP. For using an MPGA with networks to solve the FJSP, multiple operations are included, such as encoding, decoding, crossing, mutation, and migration.

2.3.1. Encoding. In this algorithm, we employ an integer encoding method [31] to generate individuals, each of which can represent a feasible solution of the FJSP. The process of encoding can be divided into two stages, to wit, encoding for machine subproblem and operation subproblem. For the former, an encoded solution is denoted by a string of integers whose length is equal to the total operation number of all jobs, in which an integer in each position represents an operation, and the value represents the ordinal number of machines in the candidate machine set (S_{ij}) of this operation. For the FJSP mentioned in Table 3, suppose there is an encoded solution (1 2 3 1) within four integers which reflect that there are a total of four operations in the two jobs. The first integer 1 means that O_{11} is arranged with M_1. For the second integer, because the candidate machine set of O_{12} is $S_{12} = \{M_2, M_4\}$, integer 2 denotes that O_{12} will be processed on the fourth machine, namely, M_4, rather than M_2. Similarly, O_{21} is arranged to be processed on M_4 and O_{22} on M_3.

For the latter one, a string of integer represents an encoded solution, and the number of integers equals the total operation number of all jobs too. The value of an integer denotes the serial number of jobs, and if a job has l_i operations, the job number will appear l_i times. The sequence of integers represents the processing order of the corresponding jobs. For example, there is an encoded solution (2 1 1 2) for the FJSP as shown in Table 3, in which integers "1" and "2" appear twice, respectively, due to both J_1 and J_2 including two operations. In detail, the fourth integer is "2," and it is the second occurrence of "2" in this string, indicating the processing of the second operation of the second job, namely, O_{22}. Therefore, the processing order of operations indicated by this solution is O_{21}, O_{11}, O_{12}, and O_{22}.

Therefore, we combine (1 2 3 1) and (2 1 1 2) together, getting an individual (1 2 3 1 2 1 1 2), which means O_{21} will be processed on M_4, O_{11} on M_1, O_{12} on M_4, and O_{22} on M_3 sequentially.

2.3.2. Decoding. An encoded individual must be decoded into an original solution of the FJSP to calculate the fitness. The decoding algorithm proposed by Shi et al. [30] was adopted in this paper as follows:

> At first, we convert an individual into a matrix, labeled as H_d. For instance, [3 1 2 1 2 1 1 2], a given individual for the FJSP in Table 3, can be converted into a matrix as shown in Figure 3.
>
> In matrix H_d, the first column stores the sequence numbers of jobs; the second column stores the sequence numbers of operations of the corresponding job; the third column stores the sequence numbers of machines in S_{ij}; and the fourth column stores the time horizon as the corresponding operation processed in the specific machine. Besides, the last two columns are reserved to store the start time and completion time, respectively. For instance, the first row of H_d represents that O_{21} is processed on M_3 within time horizon 19.

Table 2: Notations related to the FJSP.

Symbol	Meaning
n	The number of jobs
J	The job set
J_i	The ith job
m	The number of machines
M	The machine set
M_k	The kth machine
l_i	The number of operations of the ith job
O_i	The operation set of the ith job
O_{ij}	The jth operation of the ith job
S_{ij}	The candidate machine set O_{ij} can be processed on
$F_{\max}$	Makespan
F_{ij}	Completion time of O_{ij}
F_{ijk}	Finish time of O_{ij} on M_k
P_{ijk}	Processing time of O_{ij} on M_k
B_{ijk}	Start time of O_{ij} on M_k
X_{ijk}	Indicate if O_{ij} is processed on M_k

Table 3: One instance of the FJSP.

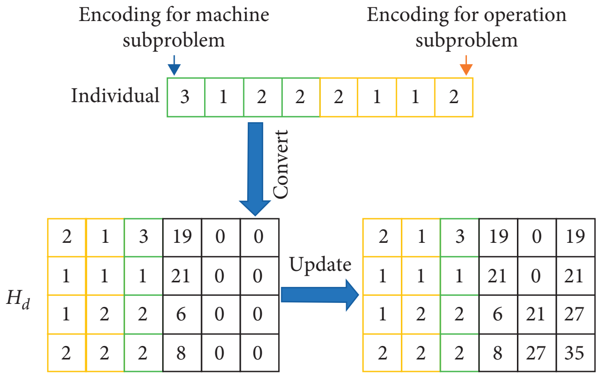
Jobs	Operations	Machines			
		M_1	M_2	M_3	M_4
J_1	O_{11}	24	—	16	19
	O_{12}	—	21	—	13
J_2	O_{21}	9	—	6	8
	O_{22}	—	—	8	6

Next, we update the start time and completion time row by row. If O_{ij} represented by the corresponding row is the first operation of J_i and M_k selected to process O_{ij} has not processed any other operation, then $B_{ijk} = 0$, and $F_{ijk} = B_{ijk} + P_{ijk}$. Assign B_{ijk} and F_{ijk} into columns 5 and 6 of H_d.

If O_{ij} is the first operation of J_i and M_k has processed other operations, then find all idle-time intervals of M_k denoted by $[S_q E_q]$ $(q = 1, 2, \ldots,)$, check all idle intervals one by one to find the first one whose duration is longer than P_{ijk}, and set $B_{ijk} = S_q$ and $F_{ijk} = B_{ijk} + P_{ijk}$.

If O_{ij} is not the first operation of J_i, the former operation $O_{i(j-1)}$ is finished on $M_{k'}$, and M_k has not been assigned to any operation, then $B_{ijk} = F_{i(j-1)k'}$ and $F_{ijk} = B_{ijk} + P_{ijk}$.

If O_{ij} is not the first operation of J_i and M_k has been assigned to other operations, then search for and check all idle-time intervals of M_k from left to right on the timeline. If $E_q - S_q \ge P_{ijk}$ and $F_{i(j-1)k'} \le S_q$, then $B_{ijk} = S_q$ and $F_{ijk} = B_{ijk} + P_{ijk}$; if $E_q - S_q \ge P_{ijk}$, $F_{i(j-1)k'} \ge S_q$, and $E_q - F_{i(j-1)k} \ge P_{ijk}$, then $B_{ijk} = F_{i(j-1)k'}$ and $F_{ijk} = B_{ijk} + P_{ijk}$. Since E_q of the last interval is positive infinity, an idle-time interval can always be found to satisfy one of these conditions. In this way, there is no earlier idle-time interval on M_k for J_i to insert without delaying or interrupting other operations, indicating that an active schedule is achieved [32]. The optimal schedule for any regular measure of performance, including makespan, will be a member of the active schedule set [33]. Therefore, the optimal value of makespan can be found by MPGAs with this decoding method.


Encoding for machine subproblem
Encoding for operation subproblem
Individual
Convert
Update
H_d

3	1	2	2	2	1	1	2

2	1	3	19	0	0
1	1	1	21	0	0
1	2	2	6	0	0
2	2	2	8	0	0

2	1	3	19	0	19
1	1	1	21	0	21
1	2	2	6	21	27
2	2	2	8	27	35

Figure 3: Illustration of decoding.

At last, after updating columns 5 and 6 of all rows, the maximum value of column 6 can be found as the makespan of the FJSP, and our goal is to minimize makespan through MPGA. As shown in Figure 3, the makespan of our example is 35.

2.3.3. Crossing. The processing of crossing is also divided into two stages. For the machine subproblem, we implement the standard two-point crossing process [30]. In this process, we choose two individuals randomly, called parent 1 and parent 2, both of which will be divided into three parts with the same size at random common breakpoints.

We partially select parts from parent 1 and parent 2 to shape offspring 1, and remaining parts compose offspring 2, which are indicated in Figure 4.

For the code of the operation subproblem [34], all jobs are randomly divided into two groups: group 1 and group 2. Offspring 1 inherits the integers of parent 1 belonging to group 1 and those of parent 2 belonging to group 2; in the same method, offspring 2 inherits the integers of parent 1 belonging to group 2 and those of parent 2 belonging to group 1, respectively, meanwhile preserving the sequence of these integers. For example, as indicated in Figure 5, jobs 1 and 2 belong to group 1, and jobs 3 and 4 belong to group 2. To get offspring 1, we first preserve the integers belonging to group 1, i.e., job 1 and job 3. Next, we insert integers of group 2, i.e., job 2 and job 4, from parent 2 into parent 1. Meanwhile, the original sequence of integers representing jobs 2 and 4 is also preserved. In the same way, we get offspring 2.

2.3.4. Mutation. For MPGAs, mutation can be a supplementary strategy to maintain diversity [35]. Based on the characteristic of integer encoding, it is divided into two stages. In the process of the machine subproblem, we randomly choose several individuals based on the mutation probability. Then, in several random positions of these selected individuals, the original integers can be replaced by alternative integers, which should be smaller than the total

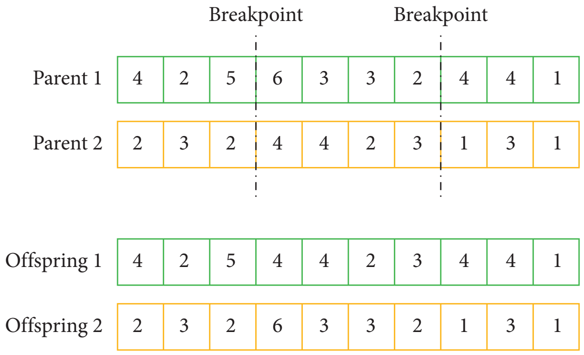

Breakpoint
Breakpoint
Parent 1 4 2 5 6 3 3 2 4 4 1
Parent 2 2 3 2 4 4 2 3 1 3 1
Offspring 1 4 2 5 4 4 2 3 4 4 1
Offspring 2 2 3 2 6 3 3 2 1 3 1

Figure 4: Crossing of the machine subproblem.

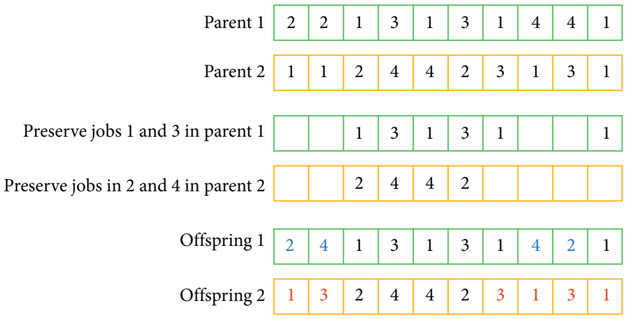

Parent 1 2 2 1 3 1 3 1 4 4 1
Parent 2 1 1 2 4 4 2 3 1 3 1
Preserve jobs 1 and 3 in parent 1 1 3 1 3 1 1
Preserve jobs in 2 and 4 in parent 2 2 4 4 2
Offspring 1 2 4 1 3 1 3 1 4 2 1
Offspring 2 1 3 2 4 4 2 3 1 3 1

Figure 5: Crossing of the operation subproblem.

number of the corresponding candidate machines. For the operation subproblem, in the same way, some individuals are randomly selected. Within each selected individual, several pairs of integers are randomly selected, and the positions of two integers, within each pair, are swapped.

2.3.5. Migration. The migration strategy is often utilized to mitigate the premature convergence of evolutionary algorithms [36]. To obtain an MPGA with networks, we divide the population of standard genetic algorithms into subpopulations, denoted by nodes. Communications between nodes, denoted by edges, occur when certain individuals in one node migrate to another periodically based on different networks. Figure 6 displays an MPGA with a certain network.

In the process of migration, we select a subpopulation randomly in an MPGA, denoted by one node of the network, and find all neighbor nodes (subpopulations). Next, we find the best elite of these subpopulations which will replace a random individual in each of these selected subpopulations.

2.3.6. Implementation of the MPGA with Networks to Solve the FJSP. As shown in Algorithm 1, we initialize N_s subpopulations, and each of them can be run parallel [37]. Within each subpopulation, there are S individuals, namely, $N_s \times S$ individuals, in total, and we first decode the individuals to get fitness, i.e., makespan. Then, tournament selection strategy is employed to generate the next subpopulation and maintain the subpopulation size unchanged. After that, crossing and mutation operators are implemented to generate new individuals. After all of the subpopulations are updated, we calculate the fitness again and select the elites of each subpopulation into the elite set. At last, our proposed migration strategy is utilized to propagate advantageous genes. As iterating to the maximum generation, the best fitness and individual are output.

2.4. Evaluation Indexes. An index to evaluate the propagation rate of advantageous genes among subpopulations under different network structures is needed. All the best individuals of each subpopulation are selected into an elite set. With the accumulation of advantageous genes through the operators of MPGA, the difference between individuals becomes small. Therefore, if the propagation rate of advantageous genes is larger, the difference between these elites will be smaller. Thus, we introduce the HDI to evaluate the difference between elite individuals. The HDI is calculated as follows [24]:

$$\mathrm{HDI} = \sum_{i=1}^{100} \sum_{j=1}^{2\times J_T} \frac{\left(1-\delta\left(h_{ij}^1, h_{ij}^2\right)\right)}{\left(100 \times 2 \times J_T\right)}, \tag{11}$$

where we only sample 100 pairs of individuals to limit time consumption and $\delta(\ldots)$ is the Kronecker function. As the two independent variables are unequal, the value of $\delta(\ldots)$ is 0; otherwise, 1. J_T represents the total operation number of all jobs, and h_{ij}^1 and h_{ij}^2 denote the ith pair's two jth integers in this sample. Therefore, smaller HDI indicates less difference between elites, suggesting the propagation among subpopulations is faster.

Besides, we use the SR to assess the performance of MPGA, better performance with higher SR. This measure is defined as follows:

$$\mathrm{SR} = \frac{N_s}{N_t}, \tag{12}$$

where N_s denotes the number of times the algorithm finds the optimal value and N_t denotes the total number of times it runs. The optimal value, mentioned in this paper, is defined as the best value found by all current algorithms by now, rather than the best value found by the currently running algorithm.

Also, due to the stochastic nature of MPGA, we have to run the proposed algorithm repeatedly, and statistical features are often utilized to analyze simulation results [38–40]. Therefore, the AOV is introduced and calculated as follows:

$$\mathrm{AOV} = \frac{1}{N_t} \sum_{i}^{N_t} \mathrm{OV}_i, \tag{13}$$

in which OV_i is the optimal value obtained by the ith run of the algorithm. The smaller the AOV is, the better the performance will be.

3. Experiments and Results' Analysis

In this section, we design experiments to evaluate how the parameters of interaction structures of subpopulations, i.e.,

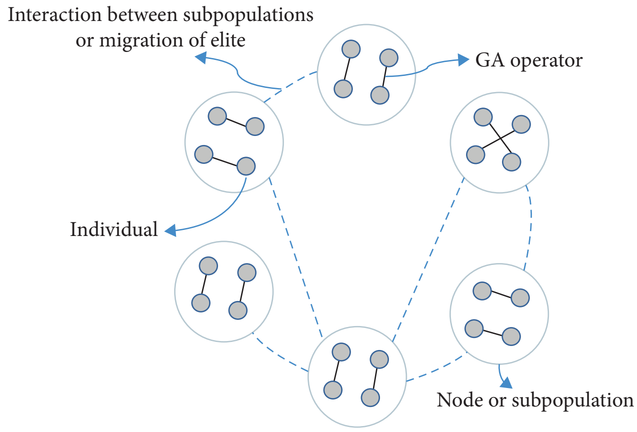


Figure 6: Schematic diagram of the MPGA with networks.

ring-shaped networks and small-world networks, affect the performance of MPGAs.

We take two benchmark problems proposed by Kacem et al. [41] as examples. One possesses 8 jobs and 8 machines, labeled as P1; the other possesses 10 jobs and 10 machines, labeled as P2. For the basic parameters of MPGA, we set the number of subpopulations N_s as 100, the size of a subpopulation S as 40, the mutation possibility P_m as 0.08, and the total number of iterations $\text{iter}_{\max}$ as 400.

3.1. MPGAs with Ring-Shaped Networks. For ring-shaped networks, the parameter K is changed constantly. The value of K slides from 1 to 24, and the interval is 1. Due to the stochastic nature of evolutionary algorithms, for each value of K, the MPGA with a certain network is run 50 times.

Figure 7 illustrates the curves of the HDI over iterations for different K. The X-axis represents the values of the HDI, and the Y-axis represents the times of iteration. As shown in the following, the HDI decreases faster as K becomes larger, which indicates the propagation rate of advantageous genes becomes larger as K increases.

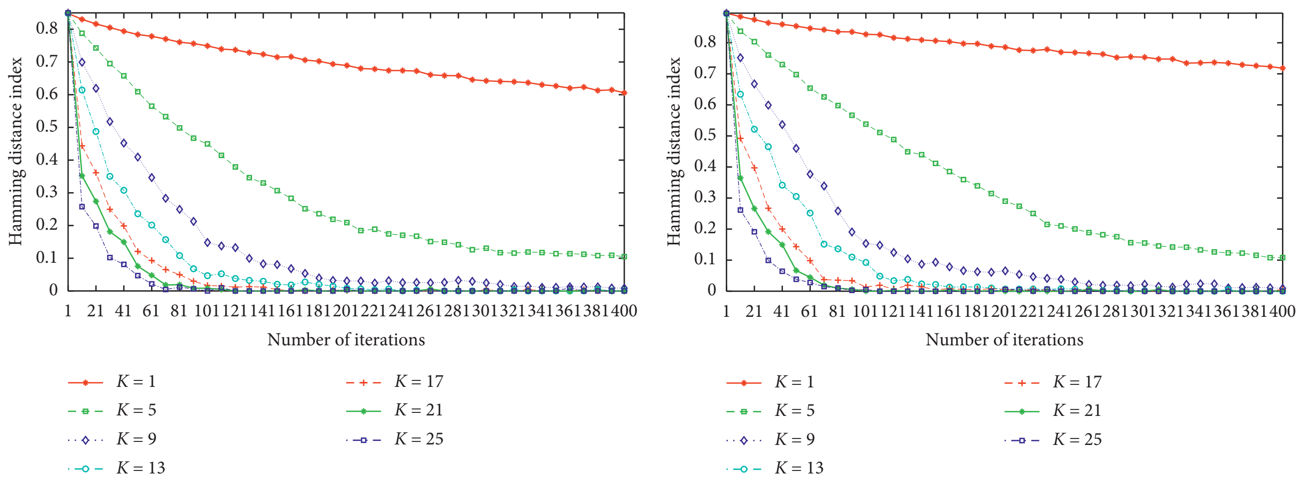
To further reveal the relationship between the propagation rate and the structural parameter of networks, Figure 8 shows the curves of the HDI at 100th iteration, 200th iteration, and 300th iteration along with APL for different K, in which the X-axis represents the value of K, and the Y-axis (left) and Y-axis (right) represent the value of HDI and APL, respectively. It can be seen that, as K becomes larger, the HDI at the same iteration decreases in accordance with the trend of APL shown by cyan lines, namely, the smaller the APL is, the faster the propagation of advantageous genes will be. Also, the variation of parameter K can change the APL to affect the propagation rate of advantageous genes. In this way, the performance of MPGAs can be influenced by different values of K.

To directly show the influence of variation of K on the performance of MPGAs, curves of SR and AOV over K are shown in Figure 9. For both benchmark problems, when K increases from 1 to 25, the performance of MPGAs increases rapidly at first and then decreases with fluctuation. Concretely, in Figure 9(a), the SR of P1 begins with 24% and increases to the peak, i.e., 90%, when $K = 4$. Then, SR decreases slowly with fluctuation. In contrast, the AOV of P1 begins with 14.9 and decreases to 14.12. Then, AOV rises slowly with fluctuation. For P2, shown in Figure 9(b), the trends of SR and AOV are similar to those in P1. The peak of the SR or the lowest value of AOV can also be found at $K = 4$. For the worst performance of P2, the lowest SR is 4% at $K = 24$, and the highest AOV is 8.42 at $K = 23$.

Therefore, the variation of K can influence the propagation rate of advantageous genes, consequently affecting the performance of algorithms. In detail, as K is very small, the APL is large, and the advantageous genes can hardly propagate to distant subpopulation in the network. Each subpopulation evolves with little communication and cannot benefit from elites of other subpopulations. However, as K is too large, the APL is very small, and the advantageous genes can be propagated rapidly. Therefore, MPGA is more likely to fall into local optimal, leading to premature convergence. For our benchmark problems, the moderate value of K is 4, so the corresponding propagation rate is neither too large nor too small, and the MPGA with networks can get the best performance.

Additionally, because of the stochastic nature of metaheuristics, to show significant differences between MPGAs with different structural parameters K of the network, the nonparametric Wilcoxon signed-rank test of the 50 independent runs is conducted [42]. The null hypothesis, termed as H_0, is set as "there exists no difference between MPGAs with different structural parameters of networks." Accordingly, the alternative hypothesis H_1 is "the MPGAs with two different values of the structural parameter are statistically different." A significance level 0.05 is employed, and a p value less than 0.05 indicates that there exist significant differences between the two samples [43]. As shown in Table 4, we take $K = 4$ vs. other values of K as examples, and the row titled with "total" indicates how many times $h = 1$ within our two benchmark problems. Basically, the larger the difference between two values of K is, more likely $h = 1$, namely, there exist differences between MPGAs with different K, especially, as the difference of K is large. The results further indicate parameter K can influence the performance of MPGAs.

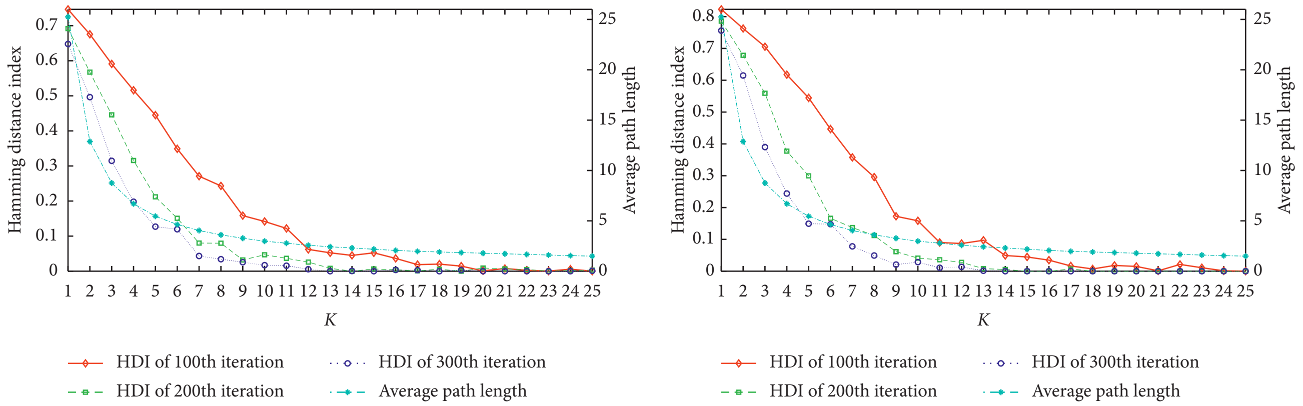
3.2. MPGAs with Small-World Networks. For small-world networks, the structural parameters are K and p as mentioned in Section 2.1.3. Since the number of subpopulations (100) is very small, causing the scale of the corresponding network to be small, the rewiring possibility p can bring significant uncertainty, namely, the number of rewired edges fluctuates with the same p. Thus, we directly change the number of rewired edges P as the structural parameter to implement simulation. We suppose the value of P slides from 0 to 120 and the interval becomes larger as P increases. When $P = 0$, the corresponding network is a ring-shaped network. It is noted that, with the same $P\,(P \neq 0)$, there still exists uncertainty because different edges can be selected to rewire to different locations. To limit this, we randomly generate three different small-world networks with the same



(a)

(b)

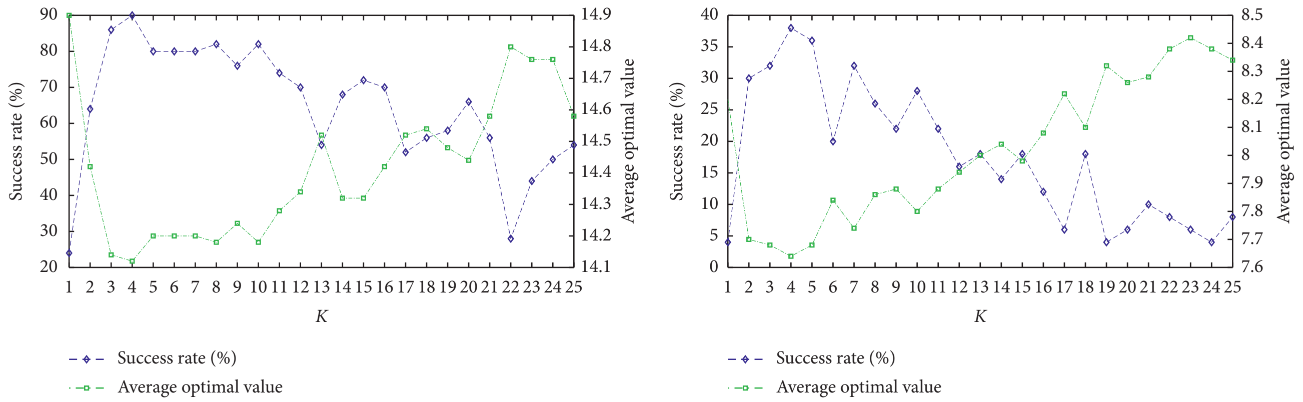
Figure 7: HDI over iterations with ring-shaped networks. (a) P1. (b) P2.



(a)

(b)

Figure 8: HDI over K with ring-shaped networks. (a) P1. (b) P2.



(a)

(b)

Figure 9: SR and AOV over K with ring-shaped networks. (a) P1. (b) P2.

```
Input:
  FJSP: the FJSP to solve
  $N_s$: the number of subpopulations;
  $S$: the size of each subpopulation;
  $P_m$: mutation probability;
  $\text{iter}_{\max}$: the maximum generation to run;
Output:
  $R^*$: the best fitness or makespan;
  $g^*$: the best individual;
$G = \{G^1, G^2, \ldots, G^M\} \leftarrow \text{encoding}(\text{FJSP}, N_s, S)$;
While $(\text{genration} \le \text{iter}_{\max})$ do
  For i_sub = 1: $N_s$ do
  $R \leftarrow \text{decoding}(G^{i\text{–sub}})$;
  $G_s^{i\text{–sub}} \leftarrow \text{selection}(G^{i\text{–sub}})$ based on $R$;
  $G_c^{i\text{–sub}} \leftarrow \text{crossing}\ (G_s^{i\text{–sub}})$;
  $G^{i\text{–sub}} \leftarrow \text{mutation}\ (G_c^{i\text{–sub}}, P_m)$;
End for
  $R \leftarrow \text{decoding}(G)$;
  Select the elites of each subpopulation to compose elite set $E$;
  $G \leftarrow \text{migration}(G, E)$;
End while
Find the best individual $g^*$ from $E$ and the corresponding fitness $R^*$.
```

Algorithm 1: MPGA with networks.

P, and MPGA is run 20 times with each network, namely, a total of 60 times for each value of P. At last, for P1, the value of K is set as 2. For P2, the value of $K$is set as 4.

Figure 10 illustrates curves of the HDI over iterations for different P. It can be noted that the HDI decreases faster as P increases, indicating the propagation rate of advantageous genes becomes larger with the increase of P.

To intuitively show the influence of variation of P on the performance of MPGAs, curves of the HDI at 100th, 200th, and 300th iterations over P are shown in Figure 11, along with APL. The X-axis represents the value of P, and the Y-axis (left) and Y-axis (right) represent the value of HDI and APL, respectively. As shown by red, green, and blue lines, with the increase of P, HDI of the same iteration decreases in accordance with the trend of APL shown by cyan lines. It can be concluded that the variation of P can change the APL, thus affecting the propagation rate of advantageous genes, by which the performance of MPGAs is influenced. Compared with the ring-shaped network in Figure 8, the decrease of APL caused by the increase of P is gentler than that caused by the increase of K, resulting in a more gradual change of HDI in Figure 11.

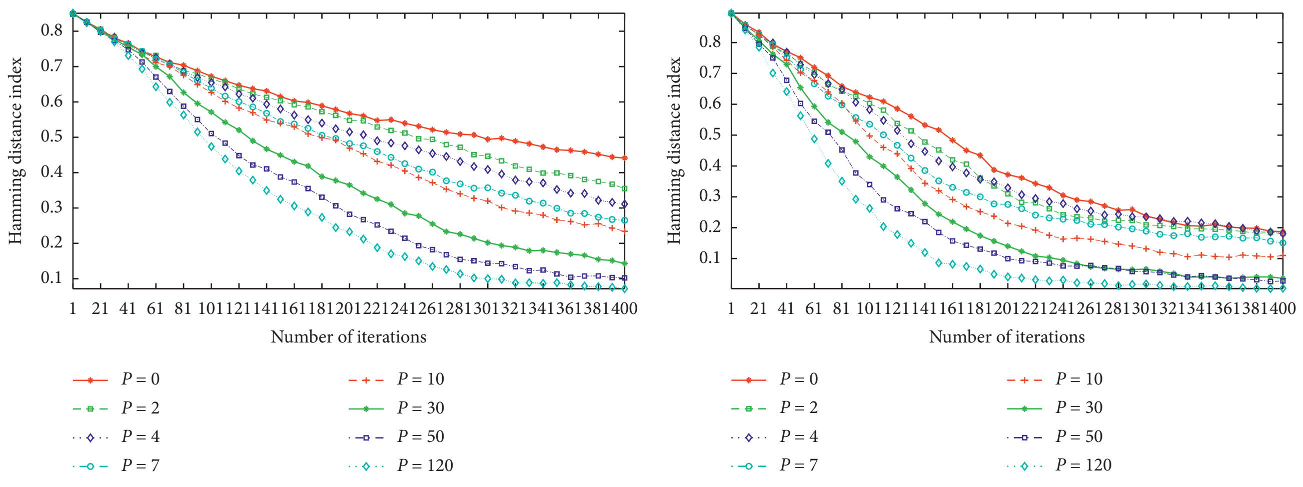
Figure 12 directly shows how the performance, measured by SR and AOV, is affected by P, in which the X-axis represents the value of P, and the Y-axis (left) and Y-axis (right) represent the value of SR and AOV, respectively. For P1, shown in Figure 12(a), the SR of P1 starting from 64.44% increases to the peak of 86.67% as $P = 4$ and then decreases with fluctuation. Besides, the AOV starting from 14.41 decreases to the lowest value 14.12 as $P = 5$ and then increases with fluctuation.

In Figure 12(b), because P2 is more sophisticated than P1, the success rate is already low. We choose to set K as 4, at which, as shown in Figure 9, MPGA with the ring-shaped network gets the best performance, i.e., SR = 38% and AOV = 7.64. Therefore, the improvement of performance is limited and subject to fluctuation due to the stochastic nature of evolutionary algorithms. However, the best performance can be found at $P = 6$ at which the peak of the SR is 48.33%, and the lowest value of AOV is 7.5. Besides, for the worst performance, the SR is 18.33%, and AOV is 7.86.

Therefore, a similar conclusion can be dawn that the variation of structural parameter P can influence the propagation rate of advantageous genes, thus affecting the performance of MPGAs with small-world networks. Besides, it is noteworthy that the edge number and node number are not changed as P varies, so the change in performance indicates that the MPGA can be improved by choosing a more proper interaction structure of subpopulations as other conditions remain unchanged.

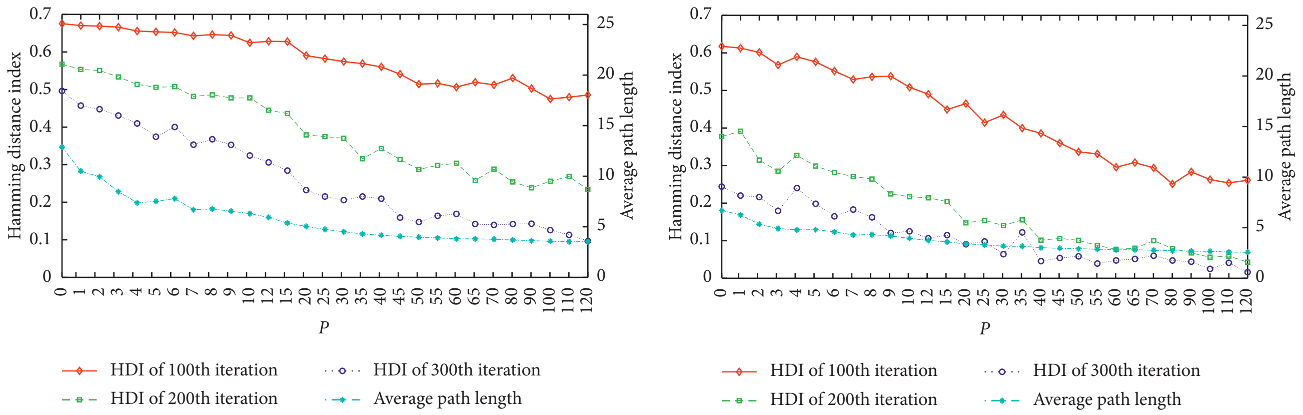
Besides, the Wilcoxon signed-rank test of the 60 independent runs is implemented again. As shown in Table 5, we take $P = 6$ vs. other values of P as examples. Although there are fewer times $h = 1$, the results still indicate that parameter P can influence the performance of MPGAs with small-world networks.

3.3. Comparison between Ring-Shaped Networks and Small-World Networks. We can conclude that both parameter K of ring-shaped networks and unique parameter P of small-world networks can influence the performance of MPGAs, but the degree of influence is different. This is because due to the variation of K and P, the APL is changed to a different extent. For small-world networks, as $K = 2$ and P slides from 0 to 120, APL varies from 12.88 to 3.53; as $K = 4$ and P slides from 0 to 120, APL varies from 7.00 to 2.57. However, for ring-shaped networks, as K slides from 1 to 25, APL varies



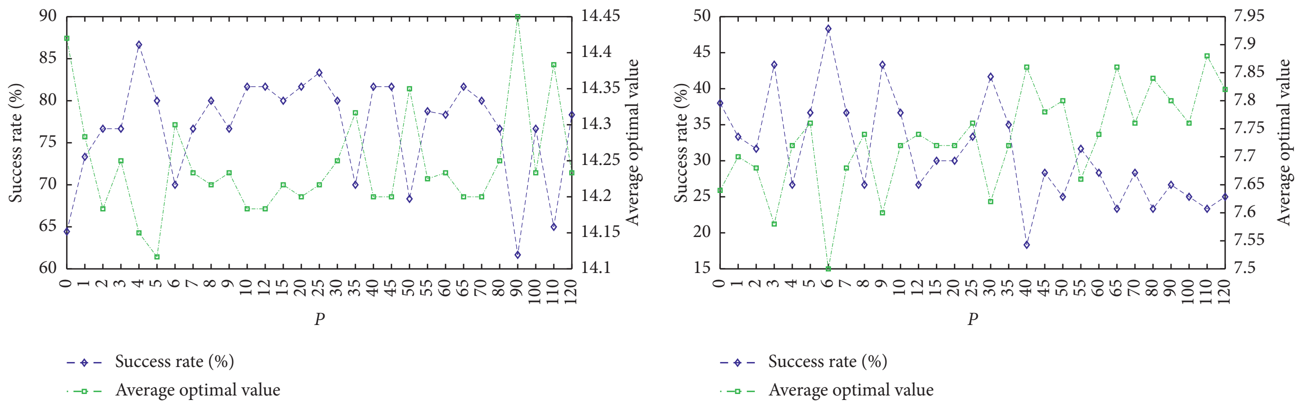
(a) (b)

Figure 10: HDI over iterations with small-world networks. (a) P1. (b) P2.



(a) (b)

Figure 11: HDI over P with small-world networks. (a) P1. (b) P2.



(a) (b)

Figure 12: SR and AOV over P with small-world networks. (a) P1. (b) P2.

Table 4: Wilcoxon signed-rank test results for ring-shaped networks.

Parameter pairs		$K = 4$ vs. $K = 1$	$K = 4$ vs. $K = 2$	$K = 4$ vs. $K = 6$	$K = 4$ vs. $K = 9$	$K = 4$ vs. $K = 11$	$K = 4$ vs. $K = 13$	$K = 4$ vs. $K = 15$	$K = 4$ vs. $K = 18$	$K = 4$ vs. $K = 21$	$K = 4$ vs. $K = 25$
P_1	p value	$1.2e-07$	0.0104	0.4850	0.2407	0.0920	$7.2e-04$	0.0540	0.0017	0.0011	$5.4e-04$
	h	1	1	0	0	0	1	0	1	1	1
P_2	p value	$1.2e-05$	0.5316	0.0709	0.0339	0.0282	0.0051	0.0076	$3.3e-04$	$3.9e-05$	$1.5e-05$
	h	1	0	0	1	1	1	1	1	1	1
Total		2	1	0	1	1	2	1	2	2	2

Table 5: Wilcoxon signed-rank test results for small-world networks.

Parameter pairs		$P = 6$ vs. $P = 1$	$P = 6$ vs. $P = 2$	$P = 6$ vs. $P = 3$	$P = 6$ vs. $P = 4$	$P = 6$ vs. $P = 5$	$P = 6$ vs. $P = 8$	$P = 6$ vs. $P = 12$	$P = 6$ vs. $P = 20$	$P = 6$ vs. $P = 55$	$P = 6$ vs. $P = 90$
P_1	p value	0.8348	0.1444	0.5485	0.0389	0.0164	0.2513	0.1444	0.0495	0.3359	0.0762
	h	0	0	0	1	1	0	0	1	0	0
P_2	p value	0.048	0.0771	0.4191	0.0133	0.0946	0.0106	0.0133	0.0285	0.041	0.0077
	h	1	0	0	1	0	1	1	1	1	1
Total		1	0	0	2	1	1	1	2	1	1

Table 6: Comparison with other algorithms.

Problems	Optimal value						
	AIA	HHS	M2	M4	MILP	HA	This article
SFJS01	66	66	66	66	66	66	66
SFJS02	107	107	107	107	107	107	107
SFJS03	221	221	221	221	221	221	221
SFJS04	355	355	355	355	355	355	355
SFJS05	119	119	119	119	119	119	119
SFJS06	320	320	320	320	320	320	320
SFJS07	397	397	397	397	397	397	397
SFJS08	253	253	253	253	253	253	253
SFJS09	210	210	210	210	210	210	210
SFJS10	516	516	516	516	516	516	516
MFJS01	468	468	468	468	468	468	468
MFJS02	448	446	446	466	446	446	446
MFJS03	468	466	466	466	446	466	466
MFJS04	554	554	564	590	554	554	554
MFJS05	527	514	514	546	514	514	514
MFJS06	635	634	634	666	634	634	634
MFJS07	879	879	928	1990	879	879	879
MFJS08	884	884	—	—	—	884	884

from 25.25 to 1.50. It can be seen that the decrease of APL is dramatic as K increases, bringing significant influence on the performance, while for P of small-world networks, the variation of P is gentle, causing the influence to be slighter. Therefore, to adjust the APL and the propagation rate of advantageous genes, we can first change parameter K, and the variation of P can be a supplementary method to adjust APL in a smaller step. In this way, we can get a more proper interaction structure to improve the performance of MPGA.

3.4. *Efficacy of the MPGA with Networks to Solve the FJSP.*

To further demonstrate the efficacy of MPGA with networks to solve the FJSP, our algorithm is employed to solve small-size FJSPs (SFJ01–10), and larger-size FJSPs (MFJS01–08) [44]. Based on the above analysis, we set $K = 4$ and $P = 3$. Other parameters for MPGA are the same as above. Previous studies, such as AIA [45], HHS [46], M2 and M4 [47], MILP [48], and HA [49], are also experimented on the aforementioned instances, and their results are cited to compare with our algorithm.

As shown in Table 6, regarding MFJS02 and MFJS03, the optimal value of our algorithm is 446 and 466 (better than 448 and 468 found by AIA). Regarding MFJS04, the optimal value is 554 (better than 564 found by M2 and 590 found by M4). Regarding MFJS05, the optimal value is 514 (better than 527 and 546 found by AIA and M4, respectively). Regarding MFJS06, the optimal value is 634 (better than 635 and 666 found by AIA and M4, respectively). Regarding MFJS07, the optimal value is 879 (better than M2 and M4). At last, for MFJS08, the optimal value is 884, which is not given by M2, M4, and MILP.

4. Conclusion and Future Scope

For further analyzing the influence of interaction structures on the performance of MPGAs, MPGA based on complex networks, such as ring-shaped networks and small-world networks, has been presented in this work. Then, how structural parameters K and $P$of these two types of networks affect the performance of MPGAs is discussed by solving the FJSP.

First, the HDI is utilized to quantitatively measure the differences of elites between subpopulations. The smaller the HDI is, the higher the propagation rate of advantageous genes will be. The simulation results indicate that the HDI has a positive correlation with the APL. This means the smaller the APL is, the higher the propagation rate is.

Next, as the structural parameters K and P increase, the curves of the HDI over iterations decrease faster, namely, the propagation rate becomes faster with the increases of K and P, resulting in the variation of the performance of MPGAs captured by SR and AOV. For instance, the highest and lowest SR of P2 are 48.33% and 4.00%. Since the propagation rate of advantageous genes can neither be too high nor too low, we should choose proper K and P for MPGAs with networks. Based on experiments, the proper intervals for K can be [2, 5], and the choice of P is dependent on the value of K because the variation of K can bring a significant change of APL, and the variation of P can be a supplementary method to adjust APL in a smaller step.

At last, the edge number and node number of small-world networks are not changed as P varies, so the change in performance indicates that MPGA can be improved by choosing a more proper interaction structure of subpopulations as other conditions remain unchanged.

Because of the limitation of computing capability, the number of nodes is not big enough compared with classic network science, but this study can still illustrate the importance of interaction structures and the meaning of introducing complex networks as a study tool. In the future, more complex or compound networks, such as multilayer networks, can be introduced into the research.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (Grant no. 51805059) and the National Green Manufacturing System Integration Support Project [2017] 327.

References

[1] D. Whitley, "A genetic algorithm tutorial," *Statistics and Computing*, vol. 4, no. 2, 1994.

[2] X.-L. Zheng and L. Wang, "A multi-agent optimization algorithm for resource constrained project scheduling problem," *Expert Systems with Applications*, vol. 42, no. 15-16, p. 6039, 2015.

[3] Y. Li and X. Zeng, "Multi-population co-genetic algorithm with double chain-like agents structure for parallel global numerical optimization," *Applied Intelligence*, vol. 32, no. 3, pp. 292–310, 2010.

[4] E. Cantú-Paz, "A survey of parallel genetic algorithms," *Calcualtion of Paralleles Reseaux System Repart*, vol. 7, no. 1, 1998.

[5] A. Leitão, F. B. Pereira, and P. Machado, "Island models for cluster geometry optimization: how design options impact effectiveness and diversity," *Journal of Global Optimization*, vol. 63, no. 4, pp. 677–707, 2015.

[6] Y.-J. Gong, W.-N. Chen, Z.-H. Zhan et al., "Distributed evolutionary algorithms and their models: a survey of the state-of-the-art," *Applied Soft Computing*, vol. 34, p. 286, 2015.

[7] W. Du, M. Zhang, W. Ying et al., "The networked evolutionary algorithm: a network science perspective," *Applied Mathematics and Computation*, vol. 338, pp. 33–43, 2018.

[8] J. L. Payne, M. Giacobini, and J. H. Moore, "Complex and dynamic population structures: synthesis, open questions, and future directions," *Soft Computing*, vol. 17, no. 7, pp. 1109–1120, 2013.

[9] C. Lu, L. Gao, and J. Yi, "Grey wolf optimizer with cellular topological structure," *Expert Systems with Applications*, vol. 107, pp. 89–114, 2018.

[10] C. Lu, L. Gao, Q. Pan, X. Li, and J. Zheng, "A multi-objective cellular grey wolf optimizer for hybrid flowshop scheduling problem considering noise pollution," *Applied Soft Computing*, vol. 75, pp. 728–749, 2019.

[11] Q. Zhang and H. Li, "MOEA/D: a multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, 2007.

[12] Y. Fu, H. Wang, M. Huang, and J. Wang, "A decomposition based multiobjective genetic algorithm with adaptive multi-population strategy for flowshop scheduling problem," *Natural Computing*, vol. 18, no. 4, p. 757, 2019.

[13] D. S. Knysh and V. M. Kureichik, "Parallel genetic algorithms: a survey and problem state of the art," *Journal of Computer and Systems Sciences International*, vol. 49, no. 4, p. 579, 2010.

[14] F. Yang, K. Gao, I. W. Simon, Y. Zhu, and R. Su, "Decomposition methods for manufacturing system scheduling: a survey," *IEEE/CAA Journal of Automatica Sinica*, vol. 5, no. 2, p. 389, 2018.

[15] Y. Hou, N. Wu, M. Zhou, and Z. Li, "Pareto-optimization for scheduling of crude oil operations in refinery via genetic algorithm," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 3, p. 517, 2017.

[16] A. Kimms, C. F. M. Toledo, P. M. França, R. Morabito, and A. Kimms, "Multi-population genetic algorithm to solve the synchronized and integrated two-level lot sizing and scheduling problem," *International Journal of Production Research*, vol. 47, no. 11, 2009.

[17] M. Zandieh and N. Karimi, "An adaptive multi-population genetic algorithm to solve the multi-objective group scheduling problem in hybrid flexible flowshop with sequence-dependent setup times," *Journal of Intelligent Manufacturing*, vol. 22, no. 6, pp. 979–989, 2011.

[18] P. Brucker and R. Schlie, "Job-shop scheduling with multi-purpose machines," *Computing*, vol. 45, no. 4, pp. 369–375, 1990.

[19] P. Pongchairerks and V. Kachitvichyanukul, "A particle swarm optimization algorithm on job-shop scheduling problems with multi-purpose machines," *Asia-Pacific Journal of Operational Research*, vol. 26, no. 2, pp. 161–184, 2009.

[20] J. Xiong, X. Tan, K.-W. Yang, L.-N. Xing, and Y.-W. Chen, "A hybrid multiobjective evolutionary approach for flexible job-shop scheduling problems," *Mathematical Problems in Engineering*, vol. 2012, p. 1, Article ID 478981, 2012.

[21] K. Gao, Z. Cao, L. Zhang, Z. Chen, Y. Han, and Q. Pan, "A review on swarm intelligence and evolutionary algorithms for solving flexible job shop scheduling problems," *IEEE/CAA Journal of Automatica Sinica*, vol. 6, no. 4, p. 904, 2019.

[22] A. A. R. Hosseinabadi, H. Siar, S. Shamshirband, M. Shojafar, and M. H. N. M. Nasir, "Using the gravitational emulation local search algorithm to solve the multi-objective flexible dynamic job shop scheduling problem in small and medium enterprises," *Annals of Operations Research*, vol. 229, no. 1, pp. 451–474, 2015.

[23] W. Zhang, J. B. Wen, Y. C. Zhu, and Y. Hu, "Multi-objective scheduling simulation of flexible job-shop based on multi-population genetic algorithm," *International Journal of Simulation Modelling*, vol. 16, no. 2, pp. 313–321, 2017.

[24] X. Shi, W. Long, Y. Li, D. Deng, and Y. Wei, "Research on the performance of multi-population genetic algorithms with different complex network structures," *Soft Computing*, vol. 24, no. 17, p. 13441, 2020.

[25] X. Shi, W. Long, Y. Li, and D. Deng, "Multi-population genetic algorithm with ER network for solving flexible job shop scheduling problems," *PLoS One*, vol. 15, no. 5, Article ID e0233759, 2020.

[26] J. W. Rivkin and N. Siggelkow, "Patterned interactions in complex systems: implications for exploration," *Management Science*, vol. 53, no. 7, p. 1068, 2007.

[27] C. Guan-Rong, W. Xiao-Fan, and L. Xiang, "Introduction to complex networks: models structures and dynamics," *Nature*, 2012.

[28] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[29] M. E. J. Newman and D. J. Watts, "Renormalization group analysis of the small-world network model," *Physics Letters, Section A: General, Atomic and Solid State*, vol. 14, 1999.

[30] X.-Q. Shi, W. Long, Y.-Y. Li, Y.-L. Wei, and D.-S. Deng, "Different performances of different intelligent algorithms for solving FJSP: a perspective of structure," *Computational Intelligence and Neuroscience*, vol. 2018, Article ID 4617816, 1 page, 2018.

[31] C. Zhang, "Bilevel genetic algorithm for the flexible job-shop scheduling problem," *Chinese Journal of Mechanical Engineering*, vol. 43, no. 4, pp. 119–124, 2007.

[32] W. Changjun and X. Yugeng, "Performance analysis of active schedules in identical parallel machine," *Journal of Control Theory and Applications*, vol. 5, no. 3, pp. 239–243, 2007.

[33] J. Hutchison and Y.-L. Chang, "Optimal nondelay job shop schedules," *International Journal of Production Research*, vol. 28, no. 2, pp. 245–257, 1990.

[34] H.-C. Chang and T.-K. Liu, "Optimisation of distributed manufacturing flexible job shop scheduling by using hybrid genetic algorithms," *Journal of Intelligent Manufacturing*, vol. 28, no. 8, p. 1973, 2017.

[35] A. Lara-Caballero, S. G. De-Los-Cobos-Silva, R. A. Mora-Gutiérrez, E. A. Rincón-García, M. Á. Gutiérrez-Andrade, and P. Lara-Velázquez, "Multiobjective genetic algorithms for reinforcing equal population in congressional districts," *Mathematical Problems in Engineering*, vol. 2019, p. 1, 2019.

[36] T. Qiu, J. Liu, W. Si, and D. O. Wu, "Robustness optimization scheme with multi-population co-evolution for scale-free wireless sensor networks," *IEEE/ACM Transactions on Networking*, vol. 27, no. 3, p. 1028, 2019.

[37] S. Tiwari, K. Kaur, Y. Pathak, S. Shivani, and K. Kaur, "Computed tomography reconstruction on distributed storage using hybrid regularization approach," *Modern Physics Letters B*, vol. 33, no. 6, Article ID 1950063, 2019.

[38] L. Li and R. Mo, "A comprehensive decision-making approach based on hierarchical attribute model for information fusion algorithms' performance evaluation," *Mathematical Problems in Engineering*, vol. 2014, p. 1, Article ID 124156, 2014.

[39] V. P. Singh, R. Srivastava, Y. Pathak, S. Tiwari, and K. Kaur, "Content-based image retrieval based on supervised learning and statistical-based moments," *Modern Physics Letters B*, vol. 33, no. 19, Article ID 1950213, 2019.

[40] Y. Pathak, K. V. Arya, and S. Tiwari, "An efficient low-dose CT reconstruction technique using partial derivatives based guided image filter," *Multimedia Tools and Applications*, vol. 78, no. 11, Article ID 14733, 2019.

[41] I. Kacem, S. Hammadi, and P. Borne, "Approach by localization and multiobjective evolutionary optimization for flexible job-shop scheduling problems," *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*, vol. 32, no. 1, p. 1, 2002.

[42] E. L. Yu and P. N. Suganthan, "Ensemble of niching algorithms," *Information Sciences*, vol. 180, no. 15, p. 2815, 2010.

[43] F. Kang, J. Li, and Z. Ma, "Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions," *Information Sciences*, vol. 181, no. 16, p. 3508, 2011.

[44] P. Fattahi, M. Saidi Mehrabad, and F. Jolai, "Mathematical modeling and heuristic approaches to flexible job shop scheduling problems," *Journal of Intelligent Manufacturing*, vol. 18, no. 3, p. 331, 2007.

[45] A. Bagheri, M. Zandieh, I. Mahdavi, and M. Yazdani, "An artificial immune algorithm for the flexible job-shop

scheduling problem," *Future Generation Computer Systems*, vol. 26, no. 4, p. 533, 2010.

[46] Y. Yuan, H. Xu, and J. Yang, "A hybrid harmony search algorithm for the flexible job shop scheduling problem," *Applied Soft Computing*, vol. 13, no. 7, p. 3259, 2013.

[47] Y. Demir and S. Kürşat İşleyen, "Evaluation of mathematical models for flexible job-shop scheduling problems," *Applied Mathematical Modelling*, vol. 37, no. 3, p. 977, 2013.

[48] E. G. Birgin, P. Feofiloff, C. G. Fernandes, E. L. de Melo, M. T. I. Oshiro, and D. P. Ronconi, "A MILP model for an extended version of the flexible job shop problem," *Optimization Letters*, vol. 8, no. 4, p. 1417, 2014.

[49] X. Li and L. Gao, "An effective hybrid genetic algorithm and tabu search for flexible job shop scheduling problem," *International Journal of Production Economics*, vol. 174, p. 93, 2016.

Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 8833645, 13 pages
https://doi.org/10.1155/2020/8833645

Research Article

A Genetic Algorithm for a Two-Machine Flowshop with a Limited Waiting Time Constraint and Sequence-Dependent Setup Times

Ju-Yong Lee

Division of Business Administration & Accounting, Kangwon National University, Chuncheon, Republic of Korea

Correspondence should be addressed to Ju-Yong Lee; jy.lee@kangwon.ac.kr

Received 27 August 2020; Revised 18 October 2020; Accepted 26 October 2020; Published 24 November 2020

Academic Editor: Dr. Dilbag Singh

This study considers a two-machine flowshop with a limited waiting time constraint between the two machines and sequence-dependent setup times on the second machine. These characteristics are motivated from semiconductor manufacturing systems. The objective of this scheduling problem is to minimize the total tardiness. In this study, a mixed-integer linear programming formulation was provided to define the problem mathematically and used to find optimal solutions using a mathematical programming solver, CPLEX. As CPLEX required a significantly long computation time because this problem is known to be NP-complete, a genetic algorithm was proposed to solve the problem within a short computation time. Computational experiments were performed to evaluate the performance of the proposed algorithm and the suggested GA outperformed the other heuristics considered in the study.

1. Introduction

The two-machine flowshop scheduling problem with a limited waiting time constraint and sequence-dependent setup times investigated in this study is motivated by semiconductor manufacturing systems. In a two-machine flowshop, all jobs are processed in the same order of machine 1 and then machine 2, which is called permutation schedule. Owing to the limited waiting time constraint, jobs must be started on the second machine within a limited waiting time after the completion of those jobs on the first machine. Additionally, sequence-dependent setup times are incurred between two consecutive jobs on the second machine. The objective function of this scheduling problem is a minimization of the total tardiness. Thus, this problem can be defined by $F2|\max - \text{wait}$ and $s_{ij}|\sum_i T_i$ in the three-field notation suggested by [1], where max-wait and s_{ij} refer to a limited waiting time constraint and sequence-dependent setup times, respectively. T_i is the tardiness of job i defined as $T_i = \max\{C_i - d_i, 0\}$, where C_i and d_i are the completion time and due date of job i, respectively. This scheduling problem is NP-complete because the typical two-machine flowshop tardiness problem is NP-complete [2].

The considered limited waiting time constraint is common in semiconductor wafer fabrication, in which there are many chemical processing processes. One example is a flowshop consisting of cleaning and diffusion processes [3]. Wafers that have completed the cleaning process may have problems that increase the likelihood of contamination and decrease quality if the surface is exposed to air for a long time before the next process (the diffusion process). To prevent this problem, the limited waiting time constraint is set between cleaning and diffusion processes. Additionally, if the chemical treatment effect is valid only when the treated wafer is processed on the next machine within a certain time after the treatment, the waiting time would be limited. If a wafer violates this time limit, the first operation must be reworked or discarded in case of high contamination. Failure to comply with the time constraints usually occurs when production volume is high, reducing productivity and causing management losses.

Setup time refers to the time required for preparation before processing jobs. If the setup time for a succeeding job varies depending on the preceding job, it is called the sequence-dependent setup time. In other words, the setup time depends on the sequence of the jobs. In semiconductor manufacturing systems, wafers for different types of products may require different chemicals or operating temperatures even though they are processed in the same machine. Therefore, job scheduling is important to schedule jobs to minimize the setup times when the sequence-dependent setup times are considered.

A two-machine flowshop scheduling problem with sequence-dependent setup times (SDSTs) to minimize the makespan is very similar to the typical travel salesman problem (TSP) [4], and dynamic programming methods [5, 6] and branch and bound algorithms [7, 8] have been proposed to obtain optimal solutions for the problem. However, since finding optimal solutions of the flowshop problem with SDST requires considerably long computation times, recent studies have focused on development of metaheuristic algorithms: genetic algorithms [9–11], variable neighborhood search algorithm [12], migrating bird optimization algorithm [13], discrete artificial bee colony optimization [14], iterated greedy algorithm [15–17], and local search based heuristic algorithm [18].

A two-machine flowshop scheduling problem with a limited waiting time (LWT) constraint to minimize the makespan has been proven to be NP-hard [19]. For this problem, [19, 20] suggest branch and bound algorithms to find optimal solutions, and [21] proposes several dominance properties for optimal solutions. The study in [22] considers a situation where some jobs skip the first machine in a two-machine flowshop and suggests an approximation algorithm to minimize the waiting time variation of jobs. The study in [23] considers a three-machine flowshop with overlapping waiting time constraints and proposes a branch and bound algorithm to minimize makespan. The study in [24, 25] consider flowshop problems with time lag constraints and develop a simulated annealing algorithm and a constructive heuristic algorithm with and insertion mechanism, respectively. On the other hand, for a flowshop problem with LWT for the objective of minimizing total tardiness, a heuristic algorithm and a Lagrangian relaxation-based lower bound strategy [26] have been developed.

Only a few studies have considered SDSTs and LWT together, although both are common and important in semiconductor manufacturing systems. The study in [3] suggests a branch and bound algorithm for the objective of minimizing makespan. On the other hand, no-wait flowshop problems with SDSTs have been studied recently. The study in [27] proposes a branch and bound algorithm for no-wait flowshop, and [28] proposes a pairwise iterated greedy algorithm, while [29] suggests an insertion neighborhood search algorithm to minimize total flowtime. Note that this study is the first attempt to consider the two-machine flowshop with SDST and LWT for the objective of minimizing total tardiness.

In this study, a mixed-integer programming formulation for the considered scheduling problem is provided to describe the problem clearly and to obtain optimal schedule by CPLEX. Additionally, heuristic algorithms are developed to obtain good (or near-optimal) schedules in a short computation time rather than an optimal schedule because the problem is NP-complete. Three types of heuristics are proposed: list scheduling methods, a constructive heuristic, and a genetic algorithm. To demonstrate the performance of the proposed heuristic algorithms, computational experiments were performed on randomly generated instances.

In the remainder of this paper, Section 2 describes the problem considered in this study in more detail including several assumptions and presents a mixed-integer linear programming formulation. Section 3 proposes heuristic algorithms. Section 4 describes the computational experiments and shows the results. Finally, Section 5 concludes the paper with a short summary.

2. Problem Description

In this section, the considered scheduling problem is described in more detail, and a mixed-integer linear programming formulation is presented. There are n jobs to be processed in the two-machine flowshop with a limited waiting time constraint and sequence-dependent setup times on the second machine. Here, only permutation schedules are considered; that is, job sequences on the two machines are the same. Although permutation schedules are not dominant in the problem, permutation schedules are common in real situations because of the simplicity of work management, flexibility in material handling, and limitation in buffer space. Note that permutation schedules are dominant in a typical two-machine flowshop with regular measures, including total tardiness. Other assumptions include the following:

(i) All jobs are ready to start at time zero

(ii) Information on the jobs to be scheduled is given; that is, processing time, due date, limited waiting time, and sequence-dependent setup time are given in advance

(iii) Preemption is not allowed

(iv) Each machine can process only one job at a time, and a job can be processed on only one machine at a time

In this paper, symbols in Table 1 are used to describe the mathematical formulation and proposed algorithms.

Completion times and tardiness of the jobs in a given sequence are computed as follows:

$$c_{[1]1} = p_{[1]1}, \tag{1}$$

$$c_{[1]2} = p_{[1]1} + p_{[1]2}, \tag{2}$$

$$c_{[r]1} = \max\left\{c_{[r-1]1} + p_{[r]1},\ c_{[r-1]2} + s_{[r-1][r]} - w_{[r]}\right\}, \quad \text{for } r \geq 2, \tag{3}$$

$$c_{[r]2} = \max\left\{c_{[r]1},\ c_{[r-1]2} + s_{[r-1][r]}\right\} + p_{[r]2}, \quad \text{for } r \geq 2, \tag{4}$$

$$T_{[r]} = \max\left\{0, c_{[r]2} - d_{[r]}\right\}, \quad \text{for all } i. \tag{5}$$

The mixed-integer linear programming (MILP) formulation for the problems is given:

Table 1: Notations for the description of the proposed algorithms.

Symbol	Definition
J	Set of jobs, $J = \{1, \dots, n\}$
i, j	Job indices
k	Machine index $(k = 1, 2)$
p_{ik}	Processing time of job i on machine k
d_i	Due date of job i
w_i	Limited waiting time of job i
s_{ij}	Sequence-dependent setup time between jobs i and j
$[r]$	Index of the job at the r^{th} position in a schedule
x_{ir}	1 if job i is assigned to the rth position in a schedule and 0 otherwise
y_{ijr}	1 if jobs i and j are assigned consecutively to positions r and $r+1$ and 0 otherwise
$b_{[r]k}$	Start time of the r^{th} job on machine k
$c_{[r]k}$	Completion time of the r^{th} job on machine k
$T_{[r]}$	Tardiness of the r^{th} job
Σ	Partial schedule
σ_i	Partial schedule obtained with σ followed by job i
U	Set of unscheduled jobs
$C_k(\sigma)$	Completion time of partial schedule σ on machine k

$$[P] \text{ minimize } \sum_r T_{[r]}, \tag{6}$$

subject to

$$\sum_r x_{ir} = 1, \quad \forall i, \tag{7}$$

$$\sum_i x_{ir} = 1, \quad \forall r, \tag{8}$$

$$b_{[r]1} + \sum_i p_{i1} x_{ir} \le b_{[r+1]1}, \quad r \le n-1, \tag{9}$$

$$b_{[r]2} + \sum_i p_{i2} x_{ir} + \sum_i \sum_j s_{ij} y_{ijr} \le b_{[r+1]2}, \quad r \le n-1, \tag{10}$$

$$b_{[r]1} + \sum_i p_{i1} x_r \le b_{[r]2}, \quad \forall r, \tag{11}$$

$$b_{[r]1} + \sum_i (p_{i1} + w_i) x_{ir} \ge b_{[r]2}, \quad \forall r, \tag{12}$$

$$x_{ir} + x_{j(r+1)} - 1 \le y_{ijr}, \quad \forall i \ne j, r = 1, \dots, n-1, \tag{13}$$

$$y_{ijr} \le x_{ir}, \quad \forall i \ne j, r = 1, \dots, r, \dots, n-1, \tag{14}$$

$$y_{ijr} \le x_{j(r+1)}, \quad \forall i \ne j, r = 1, \dots, r, \dots, n-1 \tag{15}$$

$$b_{[r]2} + \sum_i p_{i2} x_{ir} \le c_{[r]2}, \quad \forall r, \tag{16}$$

$$c_{[r]2} - \sum_i d_i x_{ir} \le T_{[r]}, \quad \forall r, \tag{17}$$

$$b_{[r]1}, b_{[r]2}, c_{[r]2} \ge 0, \quad \forall r, \tag{18}$$

$$x_{ir}, y_{ijr} \in \{0, 1\}, \quad \forall i, j, r. \tag{19}$$

Objective function (6) is to minimize the total tardiness. Constraints (7) and (8) ensure that each job is placed at only one position in the sequence, and only one job can be placed at each position. Constraints (9)–(12) define the start time of each job in the considered flowshop. Constraints (13)–(15) define the relationship between x_{ir} and y_{ijr} for sequence-dependent setup times. Constraints (16) and (17) compute the completion time and tardiness of jobs. Constraints (18) and (19) define the domain of the decision variables.

3. Heuristic Algorithms

In this section, three types of heuristic algorithms are proposed to solve the considered problem. The algorithms are list scheduling rules, a modified NEH algorithm (MNEH), and a genetic algorithm (GA). As only permutation schedules are considered in this study, feasible permutation schedules are obtained by these heuristic algorithms, and completion times and tardiness of each jobs are computed with equations (1)–(5) provided in Section 2.

3.1. List Scheduling Rules. List scheduling rules are widely used in practice (e.g., in semiconductor manufacturing systems) because they are intuitive and easy to develop and apply. Four list scheduling rules are used as described below, and schedules are obtained by sorting jobs in a nondecreasing order of the values according to the methods:

(i) Earliest due date (EDD): d_i

(ii) Modified due date (MDD): $d_i'' = \max\{d_i, C_2(\sigma i)\}, \quad \text{for } i \varepsilon U$

(iii) Slack: $\max(d_i - C_2(\sigma i), 0), \quad \text{for } i \varepsilon U$

(iv) Greedy: $\max\{C_2(\sigma i) - d_i, 0\}, \quad \text{for } i \varepsilon U$

3.2. Modified NEH Algorithm. The heuristic algorithm proposed in this subsection is modified from the constructive NEH algorithm [30], which is known to work well in flowshop scheduling problems and has been modified by many researchers for various scheduling problems. The procedure of MNEH is summarized as follows.

3.2.1. Procedure: MNEH

(i) *Step 0.* Obtain a seed sequence from the list scheduling method. Let σ be this sequence and set $\sigma^* = \varnothing, r = 1$. Go to Step 1.

(ii) *Step 1.* Select the rth job in σ. Go to Step 2.

(iii) *Step 2.* For $(i = 1; i \leq (|\sigma^*| + 1); i{+}{+})$, insert the selected job into the ith position of σ^* and compute the total tardiness.

(iv) *Step 3.* Among the $(|\sigma^*| + 1)$ partial schedules obtained in Step 2, select a partial schedule that results in minimum total tardiness. Let σ^* be the selected partial schedule. Go to Step 4.

(v) *Step 4.* $r \longleftarrow r + 1$. If $r < n$, go to Step 1; otherwise, go to Step 5.

(vi) *Step 5.* Set $i = 1$ and $j = 2$. Go to Step 6.

(vii) *Step 6.* Generate a new sequence by interchanging two jobs in the ith and jth positions in σ^*. If the total tardiness of the new sequence is less than that of σ^*, then replace σ^* with the new sequence and go to Step 5; otherwise, go to Step 7.

(viii) *Step 7.* Let $j \leftarrow j + 1$. If $j \leq n$, go to Step 6; otherwise, let $i \leftarrow i + 1$ and $j \leftarrow i + 1$. If $i < n$, go to Step 6; otherwise, terminate. (σ^*) is the final solution of this heuristic.

3.3. Genetic Algorithm. The GA was first introduced [31] as an optimization method that mimics the evolution of the natural world, and it is one of the most popular methods in the field of optimization because of its powerful search mechanism and ability to solve problems. The effectiveness of a GA depends on the procedure design, operators, and parameters. Thus, the following subsections describe the design of the proposed GA in this study.

3.3.1. Solution Representation. In the proposed GA, solutions are expressed in the chromosome structure, and the performance of the algorithm may vary depending on how the chromosome is expressed. In this study, the sequence of jobs is expressed in chromosomal structure because only the permutation schedule is considered.

3.3.2. Initial Population. To generate an initial population, list scheduling methods and the MNEH algorithm are used. That is, four solutions are generated from the list scheduling methods and used to generate four further solutions by MNEH. However, Steps 5–7 of MNEH are not applied here because these steps require a long computation time in large-sized problems. The remainder of the initial population is generated randomly.

3.3.3. Fitness Evaluation. Fitness is the evaluated objective value of a solution; that is, the evaluation value represents how good a solution is. Here, the total tardiness, which is aimed to be minimized by the objective function of the considered problem, is used for fitness evaluation. Then, the lower the total tardiness, the better the solution.

3.3.4. Selection. Selection is to choose a set of parents from the population to generate the offspring. In general, selection is based on the quality of the solutions and randomness. In the proposed GA, the two most widely used schemes in the literature (*tournament* and *roulette*) are used for selection.

In the tournament method, four chromosomes are selected randomly from the population, and pairwise comparisons are performed to choose two chromosomes as parents. That is, the better of the first two chromosomes and the better of the last two chromosomes are selected as the parents.

For the roulette methods, the inverse of the objective function value for all chromosomes (i) in the population is computed; that is, $f_i = 1/\sum_r (T_r + 1)$. Note that 1 is added to the denominator to avoid a case of dividing by zero because the objective function value (total tardiness T_r) can be zero.

The selection probability of each chromosome in the population is then obtained by $\text{prob}_i = f_i / \sum f_i$. Based on these selection probabilities of chromosomes, two chromosomes to be parents are selected randomly.

3.3.5. Crossover. The purpose of the crossover operation is to generate better offspring by exchanging the information of the selected parents. In the proposed GA, nine types of crossover are considered: one-point order crossover (OX1), two-point order crossover (OX2), order-based crossover (OBX), position-based crossover (PBX), partially matched crossover (PMX), similar job crossover (SJX), two-point similar job crossover (SJX2), similar block crossover (SBX), and two-point similar block crossover (SBX2).

For a description of these crossovers, let *parent*$_1$ and *parent*$_2$ be the selected parents to generate *offspring*.

(i) Procedure: OX1

Step 0. Let *point*$_1$ be a randomly selected cutoff point from *parent*$_1$.
Step 1. Offspring inherits the front part (from the first to *point*$_1$) of *parent*$_1$.
Step 2. Delete jobs that are already placed in *offspring* from *parent*$_2$.
Step 3. Place the jobs in *parent*$_2$ into the unplaced positions of *offspring* in the order in which they appear in *parent*$_2$.

(ii) Procedure: OX2

Step 0. Let *point*$_1$ and *point*$_2$ be two randomly selected cutoff points from *parent*$_1$.
Step 1. Offspring inherits the middle part (between *point*$_1$ and *point*$_2$) of *parent*$_1$.
Step 2. Delete jobs that are already placed in *offspring* from *parent*$_2$.

Step 3. Place the jobs in $parent_2$ into the unplaced positions of *offspring* from left to right in the order in which they appear in $parent_2$.

(iii) Procedure: OBX

Step 0. Let set_J be a set of jobs from $parent_1$ at random.
Step 1. Let set_P be a set of positions corresponding jobs in set_J from $parent_2$.
Step 2. Place the jobs in set_J into the positions in set_P of offspring from left to right in the order in which the jobs appear in $parent_1$.
Step 3. Delete jobs that are already placed in *offspring* from $parent_2$.
Step 4. Place the jobs in $parent_2$ into the unplaced positions of *offspring* from left to right in the order in which they appear in $parent_2$.

(iv) Procedure: PBX

Step 0. Select a set of positions from $parent_1$ at random.
Step 1. Offspring inherits the jobs placed in the positions of the set from $parent_1$.
Step 2. Delete jobs that are already placed in *offspring* from $parent_2$.
Step 3. Place the jobs in $parent_2$ into the unplaced positions of *offspring* from left to right in the order in which they appear in $parent_2$.

(v) Procedure: PMX

Step 0. Let $point_1$ and $point_2$ be two randomly selected cutoff points from $parent_1$.
Step 1. Offspring inherits the middle part (between $point_1$ and $point_2$) of $parent_1$.
Step 2. For each unplaced position in *offspring*, the job in the same position in $parent_2$ is placed.
Step 3. Delete jobs that are already placed in *offspring* from $parent_2$.
Step 4. If duplicate jobs occur before $point_1$ and after $point_2$, these jobs are exchanged with the remaining jobs in $parent_2$, considering the order of $parent_2$.

(vi) Procedure: SJX

Step 0. Offspring inherits jobs in the same position in the two parents.
Step 1. Let $point_1$ be a randomly selected cutoff point from $parent_1$.
Step 2. Offspring inherits the front part (from the first to the $point_1$) of $parent_1$.
Step 3. Delete jobs that are already placed in *offspring* from $parent_2$.
Step 4. Place the jobs in $parent_2$ into the unplaced positions of *offspring* in the order in which they appear in $parent_2$.

In SJX2, Steps 1 and 2 of SJX are modified by considering two cutoff points, as in OX2. Additionally, SBX and SBX2 are very similar to SJX and SJX2, respectively. In SBX and SBX2, a block means two consecutive jobs, and the only difference is that an offspring inherits the blocks in the same position in the two parents in Step 0. Thus, the detailed procedures of SJX2, SBX, and SBX2 are omitted.

3.3.6. Mutation. A mutation is an operation to escape from the local optimum and increase the variability and diversity in the population. This operation partially modifies a chromosome to generate *offspring* with a new chromosome. In the proposed GA, three types of mutation schemes are considered: interchange, insertion, and swap.

(i) Interchange: two randomly selected jobs are interchanged.
(ii) Insertion: a randomly selected job is inserted into a randomly selected position.
(iii) Swap: a randomly selected job is swapped with the next job.

3.3.7. Improvement (Local Search). Local search methods are commonly used in genetic algorithms as well as other (meta)heuristics to improve the solutions. A local search strategy based on insertion is also used in the proposed GA. This local search is triggered when the current best objective function value is not improved for a predetermined number of consecutive generations. When the local search is triggered, insertion is applied $3n$ times to the best solution in the population, where n is the number of jobs. If the new solution is better than the current best solution, the current solution is replaced with the new solution. Note that because excessive local search attempts may consume a long computation time, the number of interchanges is limited to $3n$.

3.3.8. Restart. The goal of a restart procedure is to avoid premature convergence in the population and improve the solution quality obtained by GA. The restart procedure in the proposed GA is based on that of Ruiz and Maroto [32]. This restart procedure is triggered when the current best objective function value is not improved for a predetermined number of consecutive generations. The restart procedure is summarized below.

(i) Procedure: restart

Step 0. Sort the chromosomes in the population in a nondecreasing order of their fitness value.
Step 1. Keep the first 20% of chromosomes and remove the remaining 80% of the chromosomes from the population.
Step 2. For the 20% of the population size, chromosomes are selected randomly from the first 20% of chromosomes, and the selected chromosomes are mutated once with an insertion operation.
Step 3. For the 20% of the population size, chromosomes are selected randomly from the first 20% of chromosomes, and the selected chromosomes are mutated once with an interchange operation.

Step 4. The remaining 40% of chromosomes are generated randomly.

3.3.9. Termination Criterion. The termination criterion applied to the proposed GA is the maximum CPU time. That is, when the maximum CPU time from the start of the GA elapses, the GA procedure stops.

3.3.10. Whole Procedure of the Proposed GA. The procedure of the proposed GA is summarized below, using the operators described above.

(i) Procedure: proposed GA

(ii) *Step 0.* Let POP, P_{size}, p_c, and p_m be the population, population size, crossover probability, and mutation probability, respectively.

(iii) *Step 1.* Generate an initial population with P_{size} chromosomes. Let POP be the initial population.

(iv) *Step 2.* Set $i = 1$. While $i < P_{\text{size}}$, do the following.

(v) Perform a selection operation to select two parents from POP. Let *parent*$_1$ and *parent*$_2$ be the selected parents.

(vi) Generate a random number rand in [0, 1]. If rand $< p_c$, perform a crossover operation to generate *offspring*$_1$ and *offspring*$_2$. Otherwise, *offspring*$_1$ and *offspring*$_2$ inherit the information from *parent*$_1$ and *parent*$_2$, respectively.

(vii) Generate a random number rand in [0, 1]. If rand $< p_m$, perform a mutation operation for *offspring*$_1$ and *offspring*$_2$.

(viii) Evaluate the fitness value of *offspring*$_1$ and *offspring*$_2$.

(ix) If *offsping*$_1$ is better than the worst chromosome in POP, replace the worst chromosome with *offsping*$_1$. Repeat the same procedure for *offspring*$_2$.

(x) Let $i = i + 2$.

(xi) *Step 3.* Check the local search condition; if it is met, trigger a local search for the best chromosome in POP.

(xii) *Step 4.* Check the restart condition; if it is met, trigger the restart procedure for POP. Evaluate the fitness values of the chromosomes in POP.

(xiii) *Step 5.* Check the termination condition; if it is met, terminate and the best solution in POP is the final solution. Otherwise, go to Step 2.

4. Computational Experiments

4.1. Test Instance and Environment. Computational experiments were performed with randomly generated problems to evaluate the performance of the proposed algorithms. All algorithms were coded in Java, and the experiments were performed on a PC with a 3.2 GHz Intel Core i7-8700 CPU.

To generate problem instances, job information (processing times, sequence-dependent setup times SDST, and limited waiting times LWT) was generated according to the method presented in [3]. The processing times were generated from $U(1, 50)$, where $U(a, b)$ denotes the discrete uniform distribution with a range $[a, b]$. SDST and LWT were generated from $U(0, 50)$ and $U(1, 100)$, respectively. The due dates of the jobs were generated from $U(X(1 - T - R/2), X(1 - T + R/2))$, where X is the lower bound on the makespan, and T and R are the tardiness factor and range of due dates parameters, respectively. Here, to obtain the values of X, we assumed that SDSTs for each job i were set to $s_{ij} = \min(s_{ij} | \forall j)$ and LWTs were infinite; then X is obtained by Johnson's algorithm [33] that provides the minimum makespan in the typical two-machine flowshop. For the distribution of due dates, three levels for each of T and R were used; that is, $T = (0.2, 0.4, \text{and } 0.6)$ and $R = (0.2, 0.5, \text{and } 0.8)$.

4.2. GA Calibration. The performance of GA significantly depends on its operators and parameters. Hence, a calibration experiment is needed to select the best operators and parameters. The proposed GA has eight operators and parameters, which are listed below, and all possible combinations were evaluated to obtain the best among them.

(i) Selection: two levels (tournament and roulette)

(ii) Crossover: nine levels (OX1, OX2, OBX, PBX, PMX, SJX, SJX2, SBX, and SBX2)

(iii) Mutation: three levels (interchange, insertion, and swap)

(iv) Population size (P_{size}): three levels $(30, 50, 100)$

(v) Crossover probability (p_c): six levels $(0.2, 0.35, 0.5, 0.65, 0.8, 0.95)$

(vi) Mutation probability (p_m): four levels $(0.2, 0.4, 0.6, 0.8)$

(vii) Local search (P_l): three levels $(5, 10, 20)$

(viii) Restart (P_r): four levels $(25, 50, 75, 100)$

For these eight types of operators and parameters, 46,656 different combinations are possible. To simplify the experiment, this test is divided into two experiments. The first experiment was to select the best combination for selection, crossover, and mutation, and the second experiment was to find the best combination for the other parameters.

For the evaluation and selection for the GA operators in the first experiment, a full factorial design of selection, crossover, and mutation is considered; hence, 54 different algorithms are tested. The remaining parameters were fixed to $(P_{\text{size}}, p_c, p_m, P_l, P_r) = (50, 0.5, 0.4, 10, 50)$. Additionally, four levels were considered for the number of jobs ($n = 20, 50, 100 \text{ and } 200$), and five instances for each combination (T, R, and n) were generated. That is, 180 problem instances were generated. For a termination criterion, the maximum CPU time was set to $30n$ milliseconds (*ms*). To compare different GAs for each problem instance, the relative deviation index (RDI) was considered as a measure; RDI is defined as $\text{RDUI} = (\text{Alg}_{\text{sol}} - \text{Min}_{\text{sol}})/(\text{Max}_{\text{sol}} - \text{Min}_{\text{sol}})$, where Alg_{sol} is the objective function value from the current algorithm, and Min_{sol} and Max_{sol} are the minimum and maximum values among the objective function values obtained from the algorithms in this comparison, respectively. Particularly when $\text{Max}_{\text{sol}} = \text{Min}_{\text{sol}}$, RDI will be 0 for all the algorithms.

To see the effect of GA operators on the performance, analysis of variance (ANOVA) was performed using SPSS and the ANOVA table is shown in Table 2. As can be seen in the table, the performance of the proposed GA was significantly affected by the operators with a significance level of 0.05. According to the *F*-values, three operators were significantly effective on the RDI. Figure 1 illustrates the main effect plot to find the best operators with lower RDI values. As shown in the figure, roulette, OX2, and insertion showed better performance. This is possibly because the change of fore and rear parts in sequences can lead to searching for better sequences in this scheduling problem with sequence-dependent setup times. For mutation, insertion is observed to be the best, which has been regarded in various flowshop scheduling studies as the best mutation operator [32]. Therefore, roulette, OX2, and insertion were used in the following experiment.

The second experiment was performed to find the best combination of parameters of the proposed GA. Considering all levels of $(P_{\text{size}}, p_c, p_m, P_l, P_r)$, 864 different combinations were evaluated using the same approach as that for the operators. The ANOVA results are summarized in Table 3, which shows that p_c, p_m, and P_r are the most significant parameters that affect the GA performance. As shown in Figure 2, which illustrates the main effect plot of GA parameters, RDI had the lowest value at each of $(P_{\text{size}}, p_c, p_m, P_l, P_r) = (50, 0.35, 0.6, 5, 75)$, respectively. The *bath-curves* are observed for all parameters except P_l. That is, both higher and lower values from the selected values yielded worse results, validating the chosen range of values for these parameters in the experiments. For P_l, frequent local searches were more effective when the best solution in the population was not improved. Notably, the probability of mutation is larger than that of crossover. This is probably because a small change by a mutation may lead to a better sequence rather than crossover. That is, because limited waiting time constraints and sequence-dependent setup times were considered in this study, the solution may be improved by small changes in a job sequence. Thus, using the selected parameters, the proposed GA attempts to find better solutions by keeping the sequences in good solutions and making a few changes. In summary, the proposed GA uses roulette, OX2, and insertion for the operators and $P_{\text{size}} = 50$, $p_c = 0.35$, $p_m = 0.6$, $P_l = 5$, and $P_r = 75$ for the parameters.

4.3. Performance Evaluation. First, the performance of the proposed MILP formulation was evaluated using CPLEX 12.9. To avoid excessive computation times for CPLEX, the maximum CPU time limit was set to 3,600 seconds (s) for each instance. For the evaluation of MILP, four levels for the number of jobs ($n = 5, 10, 15,$ and 20) were considered, and five instances for each combination of $(T, R, \text{and}\, n)s$ were generated. That is, 36 combinations for $(T, R, \text{and}\, n)$ were considered, and 180 problem instances are generated. The results are summarized in Table 4, which shows the average computation time (ACPUT) and the number of instances (NI) that failed to find an optimal solution within the computation time limit (3,600 s). As can be seen in the table, CPLEX obtained optimal solutions for all problems up to $n = 10$ but could not find optimal solutions for eight problems with $n = 15$. Furthermore, for $n = 20$, no problems were solved to optimality within the computation time limit.

In general, due dates significantly affect due date-related measures (e.g., total tardiness, which is the objective function of the considered scheduling problem in this study). Hence, the impact of the due dates on the performance of CPLEX was analyzed. The computation times of CPLEX with respect to the due date parameters (T, R) are summarized in Table 5, when $n = 15$. As can be seen from the table, the performance of CPLEX was significantly affected by the values of T and R. Particularly, when $T = (0.2\, \text{and}\, 0.4)$, the instances took a significantly long computation time to be solved. However, when $T = 0.6$, the instances were solved within shorter computation times. That is, CPLEX can solve problems quickly when the due dates of jobs are tight. Regarding the range of due dates R, problems in which due dates of jobs are widely distributed were solved in a shorter computation time.

Next, the performances of the proposed list scheduling rules and MNEH algorithms were evaluated by comparison to solutions obtained from CPLEX. Here, let MNEH_x be an MNEH algorithm that starts with an initial solution obtained by a list scheduling rule x. Note that, for the instances not solved within the time limit, the solutions terminated at the time limit were compared. Table 6 presents the average gaps between the heuristics and CPLEX, $(\text{Heu}_{\text{sol}} - \text{CPLEX}_{\text{sol}})/\text{CPLEX}_{\text{sol}}$, and the number of instances where heuristics found solutions which were equal to or better than those of CPLEX. As can be observed in the table, MEDD and Greedy demonstrated good performance among the list scheduling rules, and $\text{MNEH}_{\text{MEDD}}$ was the best performing heuristic algorithm. However, the average gaps from CPLEX were too large. These results explain why a metaheuristic algorithm, GA, was needed to find good solutions within a short computation time in this study.

To evaluate the performance of the suggested GA, a performance evaluation by comparing it with other algorithms was required. However, to the best of our knowledge, there are no algorithms for the same problem considered in this study. Thus, the evaluation was performed by comparing the proposed GA to CPLEX in small-sized problem instances ($n = 5$, 10, and 15) and the proposed heuristic algorithms in medium- and large-sized problem instances ($n = 20$, 50, 100, and 200). Through these comparisons, the effectiveness of the solutions from the proposed GA was evaluated. In these experiments, the maximum computation times of GA were set to four levels $(50n, 100n, 500n, \text{and}\, 1,000n)$ms. Let GA_x be the genetic algorithm in which the maximum computation time is set to $x \cdot n$ ms. The effectiveness of GA solutions compared to those from CPLEX is shown in Table 7. As can be seen from the table, the suggested GA found optimal solutions for

Table 2: ANOVA for the effect of GA operators on the performance.

Sources	Sum. squares	DF	MS	F	p-value
Corrected model	200.087	53	3.775	66.464	≤0.001
Intercept	2110.730	1	2110.730	37160.094	≤0.001
Selection	9.658	1	9.658	170.026	≤0.001
Crossover	35.657	8	4.457	78.468	≤0.001
Mutation	116.139	2	58.070	1022.336	≤0.001
Selection × crossover	27.605	8	3.451	60.749	≤0.001
Selection × mutation	0.588	2	0.294	5.175	0.006
Crossover × mutation	7.298	16	0.456	8.030	≤0.001
Selection × crossover × mutation	3.143	16	0.196	3.458	≤0.001
Error	549.038	9666	0.057	–	–
Total	2859.855	9720	–	–	–
Corrected total	749.125	9719	–	–	–

Significance level = 0.05.

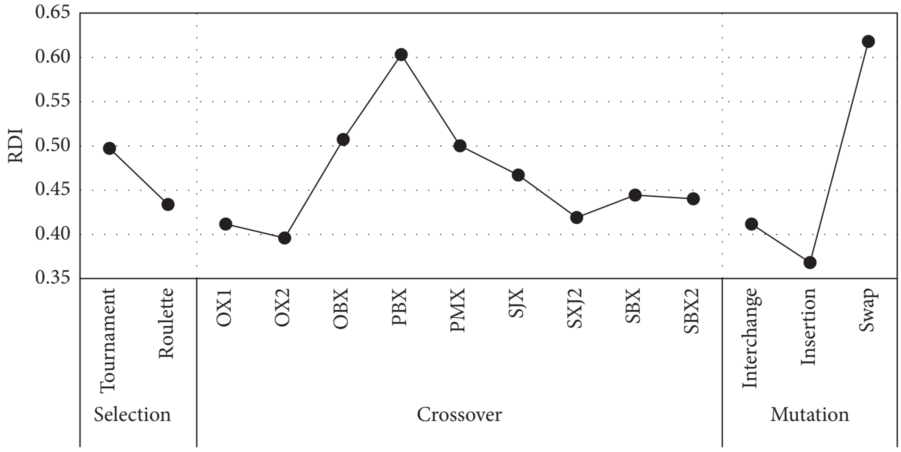


Figure 1: Main effect plot of GA operators.

almost all problem instances. Particularly, GA outperformed CPLEX for $n = 20$; a minus value means that the objective value of solutions from GA is less than that from CPLEX. This is probably because CPLEX fails to obtain optimal solutions within the maximum CPU time for $n = 20$.

For the medium and large problem instances, the GA solutions were compared with those from the MNEH algorithms. The results of this test are summarized in Table 8, which shows RDI values and the number of instances where a heuristic finds the best solutions among the heuristic solutions. The table illustrates that GAs outperformed MNEH algorithms, although GAs took longer CPU times. Additionally, in the case where GA_{50} and $\text{MNEH}_{\text{Greedy}}$ had similar computation times (approximately 10 s), GA_{50} still showed better performance. Overall, although GA requires a longer CPU time, GA solves the problems better, and the CPU time of GA_{50} is also reasonably short.

Additionally, the effectiveness of the proposed GA was evaluated through comparison with a simulated annealing (SA) algorithm, which is also a widely used metaheuristic for various optimization problems. SA attempts to improve an initial solution repeatedly by making small alterations until further improvements cannot be made by such alterations. The SA algorithm can avoid entrapment in a local optimum by allowing occasional uphill moves that deteriorate the objective function value. In this experiment, a simple SA algorithm is devised for comparison with the proposed GA. In this SA algorithm, a neighborhood solution (σ') of a solution (σ) is generated using an insertion method, which is reported as the best mutation in the previous experiment. If $\Delta = T(\sigma') - T(\sigma) < 0$, where $T(x)$ is the total tardiness of solution x, σ is replaced with σ'. Otherwise, if $\Delta \geq 0$, the replacement is done with probability $e^{-\Delta/\tau}$, where τ is a parameter called the temperature. This replacement procedure is repeated L times at a temperature, where L is called the epoch length, which is set to n^2. The temperature is initially set to $\tau_0 = 100$ and decreases gradually by a cooling function, which is set as $\tau_{x+1} = \tau_x - \alpha$, where α is a parameter that is set to 0.025 in this algorithm. The algorithm is terminated when the temperature decreases to zero or the computation time reaches the maximum computation time. The procedure of this SA algorithm is summarized below.

4.3.1. Procedure: SA

(i) *Step 0.* Set initial values for parameters used in the algorithm ($\tau_0 = 100, L = n^2$ and $\alpha = 0.025$). Let $\tau = \tau_0$ and $l = 0$.

Table 3: ANOVA for the effect of GA parameters on the performance.

Sources	Sum. squares	DF	MS	F	p-value
Corrected model	157.103	863	0.182	3.329	≤0.001
Intercept	29023.089	1	29023.089	530673.217	≤0.001
P_{size}	1.629	2	0.815	14.897	≤0.001
p_c	23.043	5	4.609	84.267	≤0.001
p_m	40.992	3	13.664	249.839	≤0.001
P_r	23.989	3	7.996	146.206	≤0.001
P_l	0.197	2	0.098	1.800	0.165
$P_{\text{size}} \times p_c$	1.070	10	0.107	1.957	0.034
$P_{\text{size}} \times p_m$	1.585	6	0.264	4.831	≤0.001
$P_{\text{size}} \times P_r$	1.922	6	0.320	5.859	≤0.001
$P_{\text{size}} \times P_l$	0.528	4	0.132	2.413	0.047
$p_c \times p_m$	12.065	15	0.804	14.707	≤0.001
$p_c \times P_r$	12.324	15	0.822	15.023	≤0.001
$p_c \times P_l$	0.752	10	0.075	1.375	0.185
$p_m \times P_r$	3.827	9	0.425	7.774	≤0.001
$p_m \times P_l$	0.355	6	0.059	1.083	0.370
$P_r \times P_l$	2.804	6	0.467	8.544	≤0.001
$P_{\text{size}} \times p_c \times p_m$	1.502	30	0.050	0.916	0.599
$P_{\text{size}} \times p_c \times P_r$	1.334	30	0.044	0.813	0.754
$P_{\text{size}} \times p_c \times P_l$	0.845	20	0.042	0.773	0.750
$P_{\text{size}} \times p_m \times P_r$	0.280	18	0.016	0.285	0.999
$P_{\text{size}} \times p_m \times P_l$	0.454	12	0.038	0.692	0.761
$P_{\text{size}} \times P_r \times P_l$	5.064	12	0.422	7.716	≤0.001
$p_c \times p_m \times P_r$	4.008	45	0.089	1.628	0.005
$p_c \times p_m \times P_l$	1.068	30	0.036	0.651	0.928
$p_c \times P_r \times P_l$	0.914	30	0.030	0.557	0.976
$p_m \times P_r \times P_l$	0.676	18	0.038	0.687	0.828
$P_{\text{size}} \times p_c \times p_m * P_r$	2.371	90	0.026	0.482	1.000
$P_{\text{size}} \times p_c * p_m \times P_l$	1.741	60	0.029	0.531	0.999
$P_{\text{size}} \times p_c \times P_r \times P_l$	1.798	60	0.030	0.548	0.998
$P_{\text{size}} \times p_m \times P_r \times P_l$	1.024	36	0.028	0.520	0.992
$p_c \times p_m \times P_r \times P_l$	2.236	90	0.025	0.454	1.000
$P_{\text{size}} \times p_c \times p_m \times P_r \times P_l$	4.704	180	0.026	0.478	1.000
Error	8458.303	154656	0.055	—	—
Total	37638.495	155520	—	—	—
Corrected total	8615.406	155519	—	—	—

Significance level = 0.05.

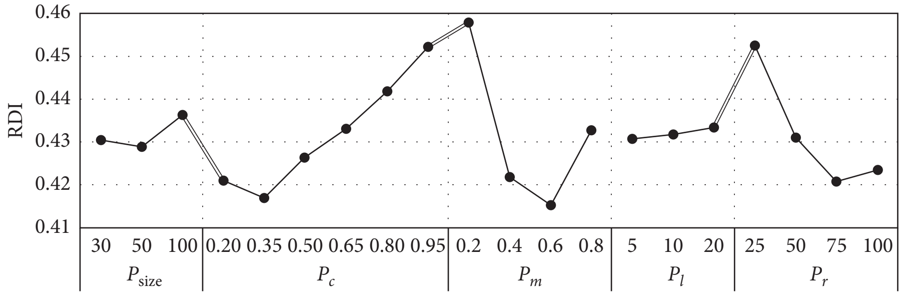


Figure 2: Main effect plots of GA parameters.

Table 4: Performance of CPLEX.

n	ACPUT (s)	NI
5	0.08	0
10	3.22	0
15	1419.53	8
20	3600	45

(ii) *Step 1.* Set an initial solution σ as the best solution among those obtained with MNEH_{EDD}, $\text{MNEH}_{\text{MEDD}}$, $\text{MNEH}_{\text{Slack}}$, and $\text{MNEH}_{\text{Greedy}}$. Let $\sigma^* \longleftarrow \sigma$.

(iii) *Step 2.* If $\tau = 0$, terminate and return σ^* as the final solution. Otherwise, let $l = 0$.

Table 5: Effect of due date parameters on the performance of CPLEX.

T	$R = 0.2$	0.5	0.8	Average
0.2	1435.14	1580.84	2313.12	1776.37
0.4	2172.74	2473.35	880.95	1842.35
0.6	905.23	255.19	759.17	639.86
Average	1504.37	1436.46	1317.75	1419.53

Table 6: Performance of the heuristic algorithms (versus CPLEX).

Algorithm	$n = 5$	10	15	20	Avg, sum
EDD	0.97, 0[a]	2.07, 0	10.58, 0	4.52, 0	4.53, 0
MEDD	0.30, 9	1.1, 0	4.69, 0	2.03, 0	2.03, 9
Slack	0.88, 3	1.48, 0	6.17, 0	2.73, 0	2.81, 3
Greedy	0.44, 9	1.02, 0	4.39, 0	2.25, 0	2.02, 9
MNEH_{EDD}	0.06, 29	0.17, 9	0.71, 0	0.25, 0	0.3, 40
$\text{MNEH}_{\text{MEDD}}$	0.05, 31	0.16, 7	0.66, 0	0.26, 2	0.28, 40
$\text{MNEH}_{\text{Slack}}$	0.04, 32	0.21, 7	0.71, 0	0.3, 1	0.31, 40
$\text{MNEH}_{\text{Greedy}}$	0.08, 29	0.28, 1	1.52, 0	0.38, 0	0.56, 31

[a] Average error gap, number of instances (out of 45 instances) for which the algorithm found solutions better than or equal to those from CPLEX.

Table 7: Evaluation of GA compared to CPLEX.

n	GA_{50}	GA_{100}	GA_{500}	GA_{1000}
5	0.00, 45[a]	0.00, 45	0.00, 45	0.00, 45
10	0.00, 44	0.00, 45	0.00, 45	0.00, 45
15	0.01, 35	0, 39	0, 42	−0.01, 43
20	−0.14, 38	−0.15, 39	−0.17, 41	−0.18, 43

[a] Average error gap, number of instances (out of 45 instances) for which the algorithm found solutions better than or equal to those from CPLEX.

Table 8: Comparison between the proposed heuristic algorithms.

Algorithm	RDI				CPUT			
	$n = 20$	50	100	200	20	50	100	200
MNEH_{EDD}	0.73, 0[a]	0.67, 1	0.57, 1	0.55, 0	<0.01	0.03	0.38	4.63
$\text{MNEH}_{\text{MEDD}}$	0.72, 0	0.64, 0	0.6, 0	0.96, 0	<0.01	0.04	0.4	6.12
$\text{MNEH}_{\text{Slack}}$	0.86, 0	0.93, 1	0.97, 0	0.66, 0	<0.01	0.04	0.44	6.12
$\text{MNEH}_{\text{Greedy}}$	0.10, 15	0.7, 1	0.66, 0	0.23, 4	<0.01	0.04	0.59	10.19
GA_{50}	0.07, 23	0.13, 6	0.19, 4	0.17, 4	1	2.5	5	10
GA_{100}	0.05, 27	0.1, 9	0.14, 5	0.15, 4	2	5	10	20
GA_{500}	0.02, 38	0.02, 33	0.05, 13	0.08, 5	10	25	50	100
GA_{1000}	0.00, 43	0.00, 45	0, 45	0.00, 45	20	50	100	200

[a] Average RDI, number of instances (out of 45 instances) for which the algorithm found solutions better than or equal to those from CPLEX.

(iv) *Step 3*. Generate a neighborhood solution σ' of σ by an insertion method.

(v) *Step 4*. Compute $\Delta = T(\sigma') - T(\sigma)$. If $\Delta < 0$, let $\sigma \leftarrow \sigma'$ and $\sigma^* \leftarrow \sigma$. Otherwise, let $\sigma \leftarrow \sigma'$ with probability $e^{-\Delta/\tau}$.

(vi) *Step 5*. Let $l = l + 1$. If $l < L$, go to Step 3. Otherwise, $\tau = \tau - \alpha$ and go to Step 2.

For fair comparison, the same maximum completion time is set for SA. Let SA_{500} and SA_{1000} be the simulated annealing algorithms in which the maximum CPU times are set to $500n$ and $1000n$, respectively. Here, GA_{500} and GA_{1000} are compared with SA_{500} and SA_{1000}, respectively.

The results of comparisons with the SA algorithms are summarized in Table 9, which shows the average rate of $\text{Alg}_{\text{so}}/\text{max}_{\text{sol}}$, where Alg_{sol} is the objective function value from the current algorithm, and $\text{Max}_{\text{sol}} = \max\{\text{GA}_x, \text{SA}_x\}$, where $x = 500$ and 1000. Note that because the objective value can be zero, the maximum values are used as the denominator rather than the minimum value to avoid dividing by zero; and if Max_{sol} is zero, then the rate will be zero. According to this measurement, algorithms with low

Table 9: Comparison between GA and SA.

n	GA_{500}	SA_{500}	GA_{1000}	SA_{1000}
20	0.966	0.983	0.966	0.987
50	0.907	0.908	0.926	0.903
100	0.818	0.967	0.828	0.968
200	0.844	0.934	0.834	0.926

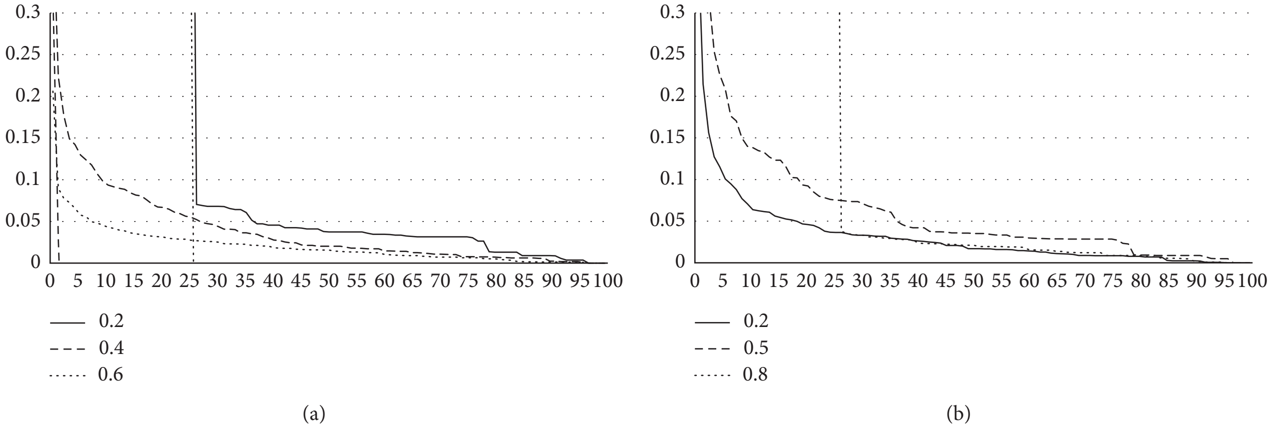


Figure 3: Convergence rate of GA solutions according to the due date parameters, T and R. (a) Convergence rate according to T values. (b) Convergence rate according to R values.

rate value are better algorithms. As can be seen from Table 9, the proposed GA outperformed a simple SA except for $n = 50$. Additionally, the outperformance is clearer in large-sized problems. Although the devised SA for this test was simple, the effectiveness of the proposed GA is validated from the results.

Finally, to check the effects of due date parameters (T and R) on the performance of GA, convergence rate charts are proposed in Figure 3. For T values, instances generated with large T values converged more quickly in the early generations. That is, when the due dates of jobs are tight, GA can find good solutions in early generations. This result is like that of CPLEX. On the other hand, instances with $R = 0.5$ converged at later generations. That is, when the range of due dates is moderate, GA requires more computation time to find good solutions. According to the results, instances with a moderate range of due dates are more complicated than those of other cases.

5. Conclusion

In this study, we considered a two-machine flowshop scheduling problem with limited waiting time constraints and sequence-dependent setup times, which is very important in semiconductor manufacturing systems. A mixed-integer programming formulation is provided to describe the considered problem clearly and used to obtain optimal solutions with CPLEX. However, CPLEX could not obtain optimal solutions when $n = 20$ because the considered problem is NP-complete. Thus, to obtain good solutions in a short computation time, three types of heuristic algorithms were proposed: list scheduling methods, a constructive heuristic algorithm, and a genetic algorithm (GA). To find the best combination of the GA, 46,656 different combinations were evaluated, and the suggested GA outperformed the other heuristics. Additionally, GA found optimal solutions for all instances with $n = 5$, 10, and 15 and showed better performance compared to CPLEX when $n = 20$. Thus, it can be said that GA works well to obtain acceptable solutions within a reasonably short computation time.

Although the limited waiting time constraint and sequence-dependent setup times are very important characteristics in semiconductor manufacturing systems, there are only a few studies considering both characteristics together. Thus, we need to develop more effective and efficient methodologies including many types of metaheuristics. On the other hand, since it is very difficult to find optimal solutions for the considered problem, it is necessary to develop effective lower bounding strategies to evaluate solutions obtained by heuristic algorithms. Also, different waiting time constraints like overlapping waiting time constraints in [23] can be considered in the future research. Finally, since the problem considered in this study is a simple type of flowshop, this problem can be extended to more practical flowshops like hybrid flowshops in which there are parallel machines in each stage.

Data Availability

All data for the computational experiments are generated randomly and the method for generating data is written in Section 4.1 in the paper.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) Grant Funded by the Korea Government (MSIT) (no. NRF-2020R1G1A1006268).

References

[1] R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. R. Kan, "Optimization and approximation in deterministic sequencing and scheduling: a survey," *Annals of Discrete Mathematics*, vol. 5, pp. 287–326, 1979.

[2] C. Koulamas, "The total tardiness problem: review and extensions," *Operations Research*, vol. 42, no. 6, pp. 1025–1041, 1994.

[3] Y.-J. An, Y.-D. Kim, and S.-W. Choi, "Minimizing makespan in a two-machine flowshop with a limited waiting time constraint and sequence-dependent setup times," *Computers & Operations Research*, vol. 71, pp. 127–136, 2016.

[4] R. Bellman, "Dynamic programming treatment of the travelling salesman problem," *Journal of the ACM*, vol. 9, no. 1, pp. 61–63, 1962.

[5] R. Bellman, A. O. Esogbue, and I. Nabeshima, *Mathematical Aspects of Scheduling and Applications*, Pergamon Press, New York, NY, USA, 1982.

[6] B. D. Corwin and A. O. Esogbue, "Two machine flow shop scheduling problems with sequence dependent setup times: a dynamic programming approach," *Naval Research Logistics Quarterly*, vol. 21, no. 3, pp. 515–524, 1974.

[7] J. N. D. Gupta, "A search algorithm for the generalized flowshop scheduling problem," *Computers & Operations Research*, vol. 2, no. 2, pp. 83–90, 1975.

[8] J. N. D. Gupta and W. P. Darrow, "The two-machine sequence dependent flowshop scheduling problem," *European Journal of Operational Research*, vol. 24, no. 3, pp. 439–446, 1986.

[9] R. Ruiz, C. Maroto, and J. Alcaraz, "Solving the flowshop scheduling problem with sequence dependent setup times using advanced metaheuristics," *European Journal of Operational Research*, vol. 165, no. 1, pp. 34–54, 2005.

[10] X. Li and Y. Zhang, "Adaptive hybrid algorithms for the sequence-dependent setup time permutation flow shop scheduling problem," *IEEE Transactions on Automation Science and Engineering*, vol. 9, no. 3, pp. 578–595, 2012.

[11] K. Peng, L. Wen, R. Li, L. Gao, and X. Li, "An effective hybrid algorithm for permutation flow shop scheduling problem with setup time," *Procedia CIRP*, vol. 72, pp. 1288–1292, 2018.

[12] R. Vanchipura, R. Sridharan, and A. S. Babu, "Improvement of constructive heuristics using variable neighbourhood descent for scheduling a flow shop with sequence dependent setup time," *Journal of Manufacturing Systems*, vol. 33, no. 1, pp. 65–75, 2014.

[13] A. Sioud and C. Gagne, "Enhanced migrating birds optimization algorithm for the permutation flow shop problem with sequence dependent setup times," *European Journal of Operational Research*, vol. 264, no. 1, pp. 66–73, 2018.

[14] Q.-K. Pan, L. Gao, X.-Y. Li, and K.-Z. Gao, "Effective metaheuristics for scheduling a hybrid flowshop with sequence-dependent setup times," *Applied Mathematics and Computation*, vol. 303, no. 15, pp. 89–112, 2017.

[15] J.-P. Huang, Q.-K. Pan, and L. Gao, "An effective iterated greedy method for the distributed permutation flowshop scheduling problem with sequence-dependent setup times," *Swarm and Evolutionary Computation*, vol. 59, Article ID 100742, 2020.

[16] F. B. Ozsoydan and M. Sağir, "Iterated greedy algorithms enhanced by hyper-heuristic based learning for hybrid flexible flowshop scheduling problem with sequence dependent setup times: a case study at a manufacturing plant," *Computers and Operations Research*, vol. 125, Article ID 105044, 2021.

[17] R. Ramezanian, M. M. Vali-Siar, and M. Jalalian, "Green permutation flowshop scheduling problem with sequence-dependent setup times: a case study," *International Journal of Production Research*, vol. 57, no. 10, pp. 3311–3333, 2019.

[18] Y. Wang and X. Li, "A hybrid local search algorithm for the sequence dependent setup times flowshop scheduling problem with makespan criterion," *Sustainability*, vol. 9, no. 12, p. 2318, 2017.

[19] D.-L. Yang and M.-S. Chern, "A two-machine flowshop sequencing problem with limited waiting time constraints," *Computers & Industrial Engineering*, vol. 28, no. 1, pp. 63–70, 1995.

[20] J.-L. Bouquard and C. Lenté, "Two-machine flow shop scheduling problems with minimal and maximal delays," *4OR*, vol. 4, no. 1, pp. 15–28, 2006.

[21] B.-J. Joo and Y.-D. Kim, "A branch-and-bound algorithm for a two-machine flowshop scheduling problem with limited waiting time constraints," *Journal of the Operational Research Society*, vol. 60, no. 4, pp. 572–582, 2009.

[22] T.-S. Yu, H.-J. Kim, and T.-E. Lee, "Minimization of waiting time variation in a generalized two-machine flowshop with waiting time constraints and skipping jobs," *IEEE Transactions on Semiconductor Manufacturing*, vol. 30, no. 2, pp. 155–165, 2017.

[23] H.-J. Kim and J.-H. Lee, "Three-machine flow shop scheduling with overlapping waiting time constraints," *Computers and Operations Research*, vol. 101, pp. 93–102, 2019.

[24] E. Dhouib, J. Teghem, and T. Loukil, "Lexicographic optimization of a permutation flow shop scheduling problem with time lag constraints," *International Transactions in Operational Research*, vol. 20, no. 2, pp. 213–232, 2013.

[25] B. Wang, K. Huang, and T. Li, "Permutation flowshop scheduling with time lag constraints and makespan criterion," *Computers & Industrial Engineering*, vol. 120, pp. 1–14, 2018.

[26] I. Hamdi and T. Loukil, "Minimizing total tardiness in the permutation flowshop scheduling problem with minimal and maximal time lags," *Operational Research*, vol. 15, no. 1, pp. 95–114, 2015.

[27] S. Wang, X. Wang, and L. Yu, "Two-stage no-wait hybrid flow-shop scheduling with sequence-dependent setup times," *International Journal of Systems Science: Operations & Logistics*, vol. 7, no. 3, pp. 291–307, 2020.

[28] C.-Y. Cheng, K.-C. Ying, S.-F. Li, and Y.-C. Hsieh, "Minimizing makespan in mixed no-wait flowshops with sequence-dependent setup times," *Computers and Industrial Engineering*, vol. 130, pp. 338–347, 2019.

[29] F. L. Rossi and M. S. Nagano, "Heuristics for the mixed no-idle flowshop with sequence-dependent setup times and total flowtime criterion," *Expert Systems with Applications*, vol. 125, no. 1, pp. 40–54, 2019.

[30] M. Nawaz, E. E. Enscore, and I. Ham, "A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem," *Omega*, vol. 11, no. 1, pp. 91–95, 1983.

[31] J. H. Holland, *Adaptation in Natural and Artificial System*, University of Michigan Press, Ann Arbor, MI, USA, 1975.

[32] R. Ruiz and C. Maroto, "A genetic algorithm for hybrid flowshops with sequence dependent setup times and machine eligibility," *European Journal of Operational Research*, vol. 169, no. 3, pp. 781–800, 2006.

[33] S. M. Johnson, "Optimal two- and three-stage production schedules with setup times included," *Naval Research Logistics Quarterly*, vol. 1, no. 1, pp. 61–68, 1954.

Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 2835023, 15 pages
https://doi.org/10.1155/2020/2835023

Research Article

An Intrusion Detection Method Based on Decision Tree-Recursive Feature Elimination in Ensemble Learning

Wenjuan Lian, Guoqing Nie, Bin Jia, Dandan Shi, Qi Fan, and Yongquan Liang

College of Computer Science & Engineering, Shandong University of Science and Technology, Qingdao, Shandong 266590, China

Correspondence should be addressed to Bin Jia; jiabin@sdust.edu.cn

Received 27 April 2020; Revised 22 September 2020; Accepted 5 October 2020; Published 23 November 2020

Academic Editor: Manjit Kaur

With the rapid development of the Internet, various forms of network attack have emerged, so how to detect abnormal behavior effectively and to recognize their attack categories accurately have become an important research subject in the field of cyberspace security. Recently, many hot machine learning-based approaches are applied in the Intrusion Detection System (IDS) to construct a data-driven model. The methods are beneficial to reduce the time and cost of manual detection. However, the real-time network data contain an ocean of redundant terms and noises, and some existing intrusion detection technologies have lower accuracy and inadequate ability of feature extraction. In order to solve the above problems, this paper proposes an intrusion detection method based on the Decision Tree-Recursive Feature Elimination (DT-RFE) feature in ensemble learning. We firstly propose a data processing method by the Decision Tree-Based Recursive Elimination Algorithm to select features and to reduce the feature dimension. This method eliminates the redundant and uncorrelated data from the dataset to achieve better resource utilization and to reduce time complexity. In this paper, we use the Stacking ensemble learning algorithm by combining Decision Tree (DT) with Recursive Feature Elimination (RFE) methods. Finally, a series of comparison experiments by cross-validation on the KDD CUP 99 and NSL-KDD datasets indicate that the DT-RFE and Stacking-based approach can better improve the performance of the IDS, and the accuracy for all kinds of features is higher than 99%, except in the case of U2R accuracy, which is 98%.

1. Introduction

Cyber-attacks are becoming universal and one type of the most common cyberspace security threats. The attackers exploit the vulnerabilities and security flaws in the computer network and information system to launch attack, which causes the disclosure of system data and the invasion of user privacy and undermines the integrity or availability of data [1]. Cyber-attack is still spreading, targeting information systems, industrial infrastructures, computer networks, and personal end-devices. In addition, it is a typical network intrusion behavior. Intrusion detection is a means to identify the attempted intrusion, ongoing intrusion, or violation.

Since 2014, the National Information Security Vulnerability Sharing Platform (CNVD) [2] in China has witnessed an average annual growth of 15.0% for security vulnerabilities. Among them, the total number of security vulnerabilities recorded in 2018 was 14,201, including 4,898 high-risk vulnerabilities (34.5%). In 2018, Chinese National Internet Emergency Center (CNCERT) sample monitoring found that the number of large-scale distributed denial of service (DDoS) attack with peak traffic exceeding 10 Gbps in China averaged more than 4,000 per month. The denial of service (DoS) attacks usually inject a large number of redundant requests into the target computer or resource. These requests can overload the system to deny sectional or all legitimate requests. Criminals usually attack sites or services located on well-known web servers, such as banks or credit card payment gateways. Because the collapse of public systems will cause great losses, the attacks on above systems (such as on banks) are more terrible.

The CNCERT found that there are 2,108 resource utilization controlled by command and control (C&C) server to initiate DDoS attack. Meanwhile, there are 1.44 million broilers, 1.97 million reflected attack servers, and about 90,000 destination IP addresses. Such distributed attack sources make it difficult for us to defend against all malicious IP. If we block the request IP on a large scale, it will cause a lot of normal requests to be killed. In addition, it should not be forgotten that the number of broilers was about 1.44 million. This shocking number means that more than 1.44 million computers or mobile phones have been intruded or controlled by attackers. Attackers can control our devices to launch illegal attacks and even read the privacy data of our devices, such as text messages, location, accounts, and passwords.

In order to defend the huge network intrusion, academia and industry have carried out a lot of exploration. An intrusion detection system (IDS) is a network security device that performs real-time monitoring of network transmissions and issues alerts. In addition, it takes the proactive response measures when suspicious transmissions are found. The IDS differs from other network security devices in that it owned the forward-looking security protection technology [3]. But, faced with the explosively severe cyberspace security situation, the traditional intrusion detection method has gradually exposed many drawbacks against the protection of network security. The typical defect is the existences of more serious False Positive (FP) and False Negative (FN).

With the vigorous development of artificial intelligence (AI), many machine learning technologies have been applied to the IDS. Machine learning (ML) improves the above problems to some extent. However, ML has its own shortcomings. When a single machine learning model meets a huge amount of data, its fitting adaptive ability is lower. This leads to poor generalization of ML as facing new data. No matter whether it is supervised learning [4–6] or unsupervised learning clustering [7] and other algorithms, there is no strong generalization ability.

In recent years, the extremely hot deep learning model has not exerted its due advantages in the IDS in the absence of the data like ImageNet. Deep learning needs plentiful of good quality data to support it, so it is less applied in the field of cybersecurity [8, 9].

In order to overcome the shortcomings of existing methods, this paper proposes a novel scheme based on the Recursive Feature Elimination and Stacking model in ensemble learning for the first time and tries to apply it in intrusion detection. Compared with the previous works, our proposed method and model have the following advantages:

(i) The Stacking technology is used, which combines the advantages of traditional machine learning. Stacking is an ensemble learning method that uses a model to perform adaptive voting weighting on the classifier. Stacking can solve the problem of insufficient fitting and generalization ability of traditional machine learning models.

(ii) A novel data processing method based on the Decision Tree-Recursive Feature Elimination (DT-RFE) is used to select features and to reduce the feature dimension. Our method eliminates uncorrelated and redundant data from the dataset to achieve better accuracy and to reduce time complexity.

(iii) We use four distributed models to learn different features so as to predict different types of attacks. In this way, we can further improve the accuracy of the model to a certain extent.

KDD CUP 99 is the most widely used dataset in the field of intrusion detection, and NSL-KDD is an improved version of KDD CUP 99. A series of comparison experiments on the KDD CUP 99 and NSL-KDD datasets show that our method can preferably improve the performance of the IDS. The proposed method in this paper improves and optimizes the existing IDS, which would reduce the feature dimension of network flow. Our method uses the idea of Stacking-based ensemble learning, and it can improve the generalization and adaptive ability of the model and have the higher accuracy.

The rest of this paper is arranged as following: Section 2 mainly presents the related work to intrusion detection research. Section 3 introduces the NSL-KDD and KDD CUP 99 datasets and analyses related processing procedure. Section 4 gives the dimension reduction method of the dataset. Section 5 raises the proposed RFE-Stacking algorithm. Section 6 carries out the experiments to verify our method and model, and Section 7 concludes the work.

2. Related Work

The IDS includes hardware and software which can actively or passively control hosts or network to detect some intrusions [3]. It embeds intrusion detection technology into a deployable system to identify and handle the violations of security policies in the computer network and system. In addition, industrial Internet security systems usually use the IDS to make up for the deficiencies of traditional network defense strategies [10]. According to the IDS input data source (undetected data), the IDS is usually divided into hybrid intrusion detection system (hybrid IDS), network-based intrusion detection system (NIDS), and host-based intrusion detection system (HIDS).

Although intrusion detection technology has been developed for many years, there are still serious problems such as higher FP and FN. Recently, with the booming development of machine learning, many artificial intelligence techniques have increasingly used in the intrusion detection field. Intrusion detection based on the classification method can extract the features of network flow and host session from an ocean of online data and audit data. In addition, it learns the classification model to discover the classification rules of hidden intrusion behavior in data. Some typical machine learning methods applied into intrusion detection are as follows: Decision tree [4], Naive Bayesian [5], k-nearest neighbor (k-NN) [6], semisupervised machine learning [11], and unsupervised machine learning and deep learning [12].

Shojafar et al. [7] proposed an unsupervised machine learning method. The method uses an automatic clustering algorithm to find the clusters with the maximum similarity between the proposed cluster elements and the smallest similarity with other clusters. The supervised learning method requires a lot of label data [13]. But, the unsupervised learning method can automatically cluster without a large amount of labeled data. It can improve the situation where there is little labeling data in the field of cybersecurity. Meanwhile, it can improve the situation in which there is few label data in the field of network security. However, the accuracy of the above method is not high, and the detection ability is not enough in the face of unknown attacks.

The traditional IDS mostly uses individual classification techniques, which do not provide the best attack detection rate. The single-model approach is more difficult to accurately predict every type of invasion. Moreover, the generalization ability of the single model is insufficient, and the detection ability is not enough as facing unknown attacks.

A new two-stage hybrid classification method is proposed, in which support vector classification (SVM) is used as the first stage of anomaly detection [14] and artificial neural network (ANN) is used as the second stage of misuse detection. Its core method is to improve accuracy by integrating the respective advantages of the two models. The two-stage model further improves the detection capability. However, the types of two models are insufficient, i.e., the advantages of multiple models cannot be maximized.

Pierre-Francois Marteau proposed covering similarity and a new similarity measure [15]. The dormant attack sequences in the normal sequences within the scope of HIDS are separated by the above similarity. Two well-known coverage similarity and three similarity measures were compared and analyzed. It shows that the covering similarity is an important index for anomaly detection in system calls sequence.

The raise of next-generation information and communication technology has led to significant growth in the number of attack and intrusion. The IDS has some dimensional flaws that tend to add time complexity and reduce resource utilization. The intelligent IDS should analyze the important characteristics of the data to reduce dimensions. The feature extraction can solve the problem to find the most informative and compact feature set. Aiming at performing each single algorithm of feature extraction to the maximum extent and developing a novel intelligent IDS, Hussain et al. [16] realized a set of linear discriminant analysis (LDA) and principal component analysis (PCA) feature extraction algorithms. The overall PCA-LDA method generates the better results and shows a higher precision ratio than a single feature extraction method. The eigenvalue decomposition of the PCA algorithm has some limitations. The principal components obtained by the PCA method may not be optimal in the case of non-Gaussian distribution.

Aburomman and Reaz [17] designed a feature sorting model based on information correlation and gain. Then, the useful or useless features are identified by combining the levels obtained from information correlation and information gain, thereby to complete feature reduction. Next, it feeds the simplified features into the feedforward neural network. In Ref. [18], a deep learning method is introduced to learn the optimal characteristics of network connection and then to choose the memetic algorithm as the final classifier in order to detect abnormal traffic. The results of NSL-KDD and KDD CUP 99 datasets both show the detection rate of 98.11%, except for the detection rate of 92.72% of the R2L attack group in the NSL-KDD dataset. In order to solve the problem of distinguishing between attack traffic and normal data flow in big data, Jia et al. [19] proposed a new real-time DDoS attack detection mechanism. First, the multidimensional characteristics of network traffic are reduced by the PCA algorithm. Next, the correlation of lower dimensional variables is analyzed. In addition, Musafer et al. [20] also proposed a feature extraction method based on trigonometric simplexes for the IDS. Similarly, Taheri et al. [21] used Hamming distance of static binary features. Andresini et al. [8] proposed a multichannel deep feature learning method, and Jiang et al. [9] also use a hierarchical deep learning method. However, the deep learning relies heavily on data volume and data quality, and its model has a sea of parameters and strong adaptive ability. It is easy to cause overfitting on the small dataset, which results in lower performance than expected.

Nowadays, to improve the performance of intrusion detection systems, various machine learning methods have been widely used. As a hot method, the ensemble learning is paid growing attentions [22]. A classifier combination tactic is generally preferred to substitute a single classifier. Mohammadi and Namadchian [23] proposed a new integrated construction method, which used the weights generated by the particle swarm optimization (PSO) to create a classifier set for higher intrusion detection accuracy. Gu et al. [24] applied a logarithmic marginal density ratio transformation on the original features in order to obtain the newest and better-quality transformed training data and then to use SVM integration to establish an intrusion detection framework. Although there are many collection methods, it is still difficult to find a suitable collection configuration for a specific dataset.

With the rapid development of 5G, Big Data, Blockchain, and Industrial Internet, the active defense and endogenous security against network intrusion has become increasingly important. Jia et al. [25] proposed a new deep neural network model to apply to the IDS. The model with four hidden layers improves the detection rate and detection accuracy on the KDD and NSL dataset. However, the experiment showed that the accuracy of U2R is only 90.91%. Jiang and Zhou [26] invented an intrusion detection method based on asymmetric deep belief network (ADBN). The ADBN model can extract features that are more conducive to classification and save more test time in the model initialization stage. It would achieve better detection accuracy for small class samples. However, the overall detection rate of the dataset is relatively low. Lu et al. [27] proposed a method by using the deep self-encoder with unsupervised learning for migration learning. It is a positive exploration because there are still some inherent shortcomings of unsupervised learning.

In conclusion, the previous work mainly used simple dimensionality reduction algorithms, such as PCA to perform dimensionality reduction for all features only once. In addition, they use the single model to directly learn and classify features. The previous work focused on optimization and improvement of the single model, and the most of them only focused on overall classification accuracy, while ignoring the accuracy on small samples. Similarly, the deep learning methods used in previous works often fail to obtain particularly ideal results due to insufficient dataset quantity and quality. In order to improve the above methods, this paper proposes an intrusion detection method based on DT-RFE in ensemble learning. Compared with the PCA algorithm, the DT-RFE has simpler application conditions and pays more attention to the actual effect. In addition, compared with other single-model methods and simple ensemble learning methods, our multilayer and multimodel Stacking method has stronger integration ability. The adaptive ensemble learning method based on machine learning models can better reflect the advantages of heteroid models.

3. Preliminaries

The process of sorting data in the data source into the data warehouse according to certain rules is called data preprocessing. In this paper, the original NSL-KDD and KDD CUP 99 datasets need be preprocessed to verify our method and model. On the one hand, the data in the original sample should be normalized, and the sample data are fabricated into the format that is suitable for calculation. On the other hand, some important features that affect the prediction result are selected by the feature selection algorithms, with the purpose of reducing data redundancy and computation complexity.

3.1. Data Preprocessing. There are four types of intrusions in the original dataset, and each intrusion record is made up of the 41-dimensional feature vector. The example of a raw record in the intrusion detection dataset is as follows:

$$\begin{aligned} X_i = \{&0, \text{tcp}, \text{http}, \text{SF}, 215, 45076, 0, 0, 0, 0, 0, 1, 0,\\ &0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0.00, 0.00, 0.00, 0.00,\\ &1.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00,\\ &0.00, 0.00, 0.00, 0.00, 0.00, \text{normal}\}. \end{aligned} \tag{1}$$

X_i has 42 dimensions, including 41 attributes and one label. Here, "normal" is the label that records X_i. In addition, i in X_i means that X_i is a row of the dataset. The first step in data processing is data filtering. Many intrusion records are the same in actual captured data, so we remove duplicate data to eliminate information redundancy. Furthermore, the 3 features in them are character-type features, and they are "protocol_type," "service," and "flag." Therefore, we use LabelEncoder() to convert all the data captured into digital types from different IDS input sources to simplistically process the data. The target labels with values between 0 and n_classes-1 can be transformed into a continuous numeric variable by LabelEncoder(). As shown in Table 1, symbol features are mapped to digital features.

The difference of value range and metrics varies greatly among different features. For avoiding the disappearance of the small-valued attribute and to reduce the repetitive calculation of the iteration amount, the numerical data need be encoded by one-hot. The one-hot coding is used to represent the protocol_type, service, and flag attributes in X_i, and the 84-dimensional data are obtained. Here, the example obtained is shown in Table 2.

Table 1: Example of data numeralization.

	Protocol _type	Service	Flag
0	1	20	9
1	2	44	9
2	1	49	5
3	1	24	9
4	1	24	9

Because the NSL-KDD and KDD CUP 99 datasets are divided into five categories, each of which has a different amount of data, for example, denial of service (DoS) attacks and normal data have hundreds of thousands of data. With only thousands of data, this situation is called the uneven distribution of data categories. For the distribution, the common accuracy rate cannot be used as an indicator of the evaluation model. We rename each attack tag, i.e., normal = 0, DoS = 1, Probe = 2, R2L = 3, and U2R = 4. The rules are as follows:

(i) 'normal': 0

(ii) 'back': 1, 'worm': 1, 'land': 1, 'pod': 1, 'smurf': 1,'teardrop': 1, 'mailbomb': 1, 'apache2': 1, 'processtable': 1, 'Neptune': 1, and 'udpstorm': 1

(iii) 'portsweep': 2, 'satan': 2, 'ipsweep': 2, 'mscan': 2, 'saint': 2', and nmap': 2

(iv) 'guess_passwd': 3, 'multihop': 3, 'phf': 3, 'warezclient': 3, 'httptunnel': 3, 'warezmaster': 3, 'sendmail': 3, 'named': 3, 'snmpgetattack': 3, 'snmpguess': 3, 'xlock': 3, 'ftp_write': 3, 'imap': 3, 'spy': 3, and 'xsnoop': 3

(v) 'loadmodule': 4, 'buffer_overflow': 4, 'ps': 4, 'perl': 4, 'rootkit': 4, 'sqlattack': 4, and 'xterm': 4

After encoding the 41-dimensional feature vector by one-hot, X_i becomes 122-dimensional feature vectors. Next, we normalize the dataset and turn all data into the same value interval. Normalization can prevent some large-value features from erroneously affecting the results. The data are converted from X_{ij} to X_{ij}', where i and j represent the rows and columns of the data in the dataset. The function is shown as follows:

$$\begin{aligned} X_{ij}' &= \frac{X_{ij-\mathrm{AVG}_j}}{\mathrm{STAD}_j},\\ \mathrm{AVG}_j &= \frac{1}{n}\left(X_{1j} + X_{2j} + \cdots + X_{nj}\right),\\ \mathrm{STAD}_j &= \frac{1}{n}\left(\left|X_{1j} - \mathrm{AVG}_j\right| + \left|X_{2j} - \mathrm{AVG}_j\right| + \cdots + \left|X_{nj} - \mathrm{AVG}_j\right|\right), \end{aligned} \tag{2}$$

where the average value is AVG_j and STAD_j is the average absolute deviation.

Table 2: Example of data one-hot encoding result.

	Protocol_type_icmp	Protocol_type_tcp	Protocol_type_udp	Service_IRC	Service_X11	...	Flag_RSTR	Flag_S0	Flag_SF	Flag_SH
0	0	1	0	0	0	...	0	0	1	0
1	0	0	1	0	0	...	0	1	0	0
...	...	...	...	...	...	...	...	...	...	...
125968	0	1	0	0	0	...	0	1	0	0
125969	0	1	0	0	0	...	0	0	1	0

In the above calculations, the following judgments are required:

$$X'_{ij} = \begin{cases} \dfrac{X_{ij} - \mathrm{AVG}_j}{\mathrm{STAD}_j}, & \text{if } \mathrm{AVG}_j > 0, \\ 0, & \text{if } \mathrm{AVG}_j = 0. \end{cases} \quad (3)$$

After this function, the normalization of the dataset and the preliminary work of the dataset have been completed.

3.2. Feature Extraction. After data preprocessing is completed, it is usually impossible to directly input the data into a learner due to the high dimensions of data, so it is necessary to select some fewer valuable features to train by machine learning. Good feature extraction can find the most useful and important features.

In order to speed up the subsequent training algorithm, in the above obtained 122-dimensional feature data, the main characteristics of DoS, Probe, R2L, and U2R were found by reducing the dimensions or extracting features. In this paper, a novel DT-RFE is used. Here, RFE is to iteratively build the model and then pick the best (or worst) features that are selected according to the coefficients. Next, the selected features are put aside, and it repeats on the remaining features. The process continues until all the features are traversed. The eliminated sequences in this process are the ordering of features. The character of the RFE itself allows us to better perform manual feature selection. The stability of the RFE depends largely on which the model is used at the bottom during iteration. If the relationship between a feature and a response variable is nonlinear, a tree-based method or an extended linear model can be used. Usually, some tree-based methods are easier to use; this is because they model nonlinear relationships and do not require much debugging. In feature extraction, the underlying model is selected as a simple Decision Tree algorithm.

In addition, information entropy is an important index of feature selection in Decision Tree. There are many types of sample in the training dataset that need to be classified. Decision Tree calculates the information entropy of the dataset and divides the dataset layer by layer. Finally, each type of sample is divided separately. Entropy is the measure of the uncertainty of a random variable. Suppose that X is a random variable with a finite number of values, and its probability distribution is denoted as follows:

$$P(X = x_i) = p_i. \quad (4)$$

Among them, x_i corresponds to p_i one by one. The entropy of the random variable X is defined as follows:

$$H(X) = -\sum_{i=1}^{n} p_i \log p_i. \quad (5)$$

It is not difficult to find that the uncertainty of the random variable is greater, while the entropy is greater. Because the probability value must be less than 1, the logarithm of this probability must be less than 0. So, there is a minus sign in the formula to counteract the negative number produced by log. When the difference of p_i corresponding to different x_i is greater, the entropy $(H(X))$ is greater.

Similarly, suppose that the joint probability distribution of random variable (X, Y) is expressed as follows:

$$P\left(X = x_i, Y = y_j\right) = p_{ij}. \quad (6)$$

Conditional entropy $H(Y|X)$ represents the uncertainty of the random variable Y under the given condition X, and it is calculated as follows:

$$H(Y|X) = \sum_{i=1}^{n} p_i H(Y|X = x_i). \quad (7)$$

The information gain of feature A to training dataset D is $G(D|A)$. The formula is shown as follows:

$$G(D|A) = H(D) - H(D|A). \quad (8)$$

The information gain represents the degree to which the inaccuracy of Y information is reduced after the information of feature X is learned. Using $G(D|A)$ as a feature to divide the dataset may cause the problem of preferring to select features with more values. So, information gain ratio is another criterion of feature selection, which can correct the above problem:

$$G_R = \frac{G(D|A)}{H(D)}. \quad (9)$$

Suppose that the number of leaf nodes in Decision Tree T is $|T|$, and t is one of the leaf nodes. There are N_t samples in this node, and the number of sample points of k is N_k. H_t is the entropy on the leaf node, and $\alpha(\geq 0)$ is an optional parameter related to the penalty term. So, the loss function of Decision Tree T is defined as follows:

$$L_\alpha(T) = \sum_{t=1}^{|T|} N_t H_t(T) + \alpha|T|, \quad (10)$$

where the calculation of entropy is as follows:

$$H_t(T) = -\sum_{k} \frac{N_{tk}}{N_t} \log \frac{N_{tk}}{N_t}. \quad (11)$$

The loss function (it is also called as objective function) is used to evaluate the difference degree between the true value and the predicted value. The goal of model learning is to reduce the loss function.

The first term on the right side of the equal sign in the loss function (10) can be defined as follows:

$$C(T) = \sum_{t=1}^{|T|} N_t H_t(T) = -\sum_{t=1}^{|T|}\sum_{k=1}^{K} N_{tk} \log \frac{N_{tk}}{N_t}. \quad (12)$$

In the case, the loss function can be simplified as follows:

$$C_\alpha(T) = C(T) + \alpha|T|, \quad (13)$$

where the $C(T)$ represents the prediction error of the model to the training data. In addition, the complexity of the model is also important. $|T|$ represents the complexity of the model, which can be regarded as a penalty term in the loss function. In addition, α can determine the degree of penalty and can balance the model complexity and prediction error.

In our proposed method, the Recursive Feature Elimination Cross-Validation (RFECV) uses cross-validation based on RFE, and it is to preserve the most representative characteristics. The cross-validation based on RFE is performed on different feature combinations. By calculating the sum of its decision coefficients, the score with importance of different features is finally got, and then the best feature combination is retained.

As shown in Algorithm 1, the REF method uses Decision Tree as the training of multiple rounds. According to the weight coefficients generated by training, better features are retained for the next round of training. For prediction models with feature weights, RFE recursively reduces the size of feature set under review to select features. Firstly, based on the original features, the prediction models are trained to assign a weight to each feature. And then, the feature set can be simplified through deleting the features whose weight has the smallest absolute value. Such recursion will continue until the number of remaining features reaches the required number. Compared with RFE, RFECV adds a cross-validation process to better select the optimal number of features. For a feature set with d, the number of all its subsets is 2^{d-1} (including the empty set). The Decision Tree calculates the validation error of all subsets and selects the subset with the smallest error as the selected feature.

4. Model Building

The Stacking fusion algorithm by combining DT-RFE is creatively proposed, and its implementation consists of three stages. Firstly, the dataset should be prepared and normalized. Next, a feature extraction for dimension reduction is built.

After the feature extraction, a machine learning algorithm is used to classify and verify the dataset, and finally the ensemble learning is used to generate the generic function classifier. Here, the ensemble learning will test a series of classifiers to integrate the learning results through some rules so that it can obtain better generalization performance than a single learner.

However, there are still two main problems of the integrated algorithm: one is how to select some individual learners and the other is how to choose the strategies to integrate these individual learners into a powerful learner. A good integrated algorithm is to ensure the diversity of individual learners (excellent and different), and the integration of unstable algorithms can also get a significant performance improvement. Common kinds of integration learning are as follows: (1) bagging for reducing variance, (2) boosting for reducing bias, and (3) stacking for improving prediction results.

In this paper, Stacking is used as a powerful ensemble learning model to apply to the fusion algorithm. Stacking was proposed by Wolpert [28] in 1992. Its basic idea is to use a model to fuse the prediction results of several single modules in order to reduce the generalization error of the single individual. Unlike the voting and weighting methods used by the bagging and boosting algorithms, in order to obtain the weight value of each basic classifier, the Stacking algorithm will train another classification model that can learn the weight value of each classifier. Therefore, these single modules are called primary classifiers, and the Stacking fusion model is called the secondary (or meta [29]) classifier. As shown in Figure 1, Stacking first trains several single classifiers from the initial training set, then integrates the output of the single module as sample features, and uses the original sample labels as new data sample labels in order to generate a new training set. Subsequently, a new model for the new training set is trained, and finally the new model is adopted to predict the samples. The model of the Stacking fusion algorithm is essential to design a hierarchical structure, and each layer contains a sea of classification models. All single classification models are generated using different learning algorithms (some heterogeneous models). Algorithm 2 shows the operation flow of the Stacking fusion model.

Here, we randomly divide the initial training set D into k similarly-sized sets $D_1, D_2, \ldots, D_k$, D_j. In addition, $\overline{D_j}$ represents the j-th testing set and training set, respectively. When the T primary learning algorithms are given, the primary learner $h_{jt}$is obtained by using the t-th learning algorithm on $\overline{D_j}$. For each sample x_i in $\overline{D_j}$, we define the prediction result of the t-th model as $Z_{jt} = h_{jt}(x_i)$. Then, the secondary model training sample feature generated by sample x_i is $Z_i = Z_{i1}, Z_{i2}, \ldots, Z_{iT}$. The sample label is still the original sample, and it is labeled as y_i. Therefore, after k-round T model training and prediction, the secondary training set $D' = \{(Z_i, y_i)\}_{i=1}^{m}$ is obtained, and then $D'$is used to train the secondary model. The secondary model h' is a function of $(Z_{i1}, Z_{i2}, \ldots, Z_{iT})$ for y.

In the training phase of Stacking, the training set of the secondary model is generated by using the primary model. If the first-level model is used to directly predict the initial training set samples to generate a second-level training set, there will be a great risk of overfitting. Therefore, the unused samples that are used to train the first-level model will generate the training set of the second-level model. In general, the cross-validation is a more common method. Next, we use k-fold cross-validation as an example to show how the training set of the secondary model is acquired.

Figure 1 shows the process that constructs a fused model training set for a single module through 5-fold cross-validation. It is seen that the whole of 5 times are trained, and predictions are made on the five 1-fold verification sets and the test set, respectively. By concatenating the prediction results of five 1-fold verification sets and averaging the prediction results of five test sets, a list of features on the new training set and the new test set can be obtained.

Here, we first try to choose some simpler models, including Logistic Regression, Random Forest [30], SVM, and Decision tree. After analysis, it is found that these models do not have outstanding utilization of features. Therefore, some mainstream models such as AdaBoost and Gradient Boosted Decision Tree (GBDT) have been tried to enhance the nonlinear expression ability of the models. Aiming at

```
Input: Training sample set
Initialize original feature set $S = \{1, 2, \ldots, D\}$ and feature ordering set $R = []$.
for $d = 1, 2, \ldots, D$ do
    The Decision Tree classifier is trained, and the feature selection of single variable by $F$-test (ANOVA) is obtained.
    Calculate ranking criterion score
    Find the feature with the lowest ranking score
    Update feature set $R = [p, R)$
    Remove other features in $S$: $S = S/p$
    until $s = [\ ]$
end for
output: Feature sort set R
```

Algorithm 1: The DT-RFE algorithm.

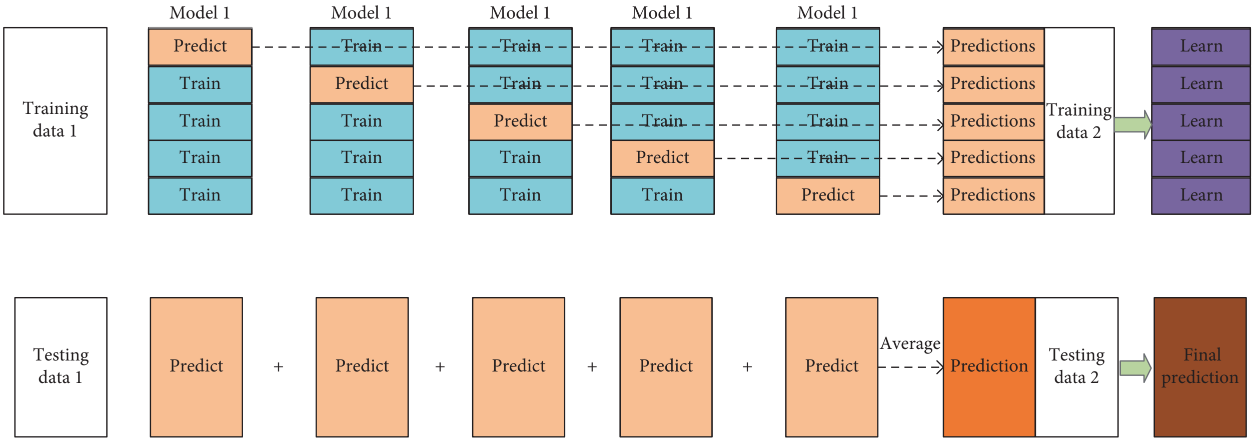


Figure 1: 5-fold cross-validation process for building a fusion model training set.

```
Input: $\mathbf{D} = \{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \ldots, (\mathbf{x}_\mathbf{m}, \mathbf{y}_\mathbf{m})\}$; the single model learning algorithm $\delta_1, \delta_2, \ldots, \delta_\mathbf{T}$; and the fusion learning algorithm
model $\delta$
Steps:
    for $t = 1, 2, \ldots, T$ do
    $h_t = \delta_t(D)$;
    end for
 $D' = \varnothing$
    for $i = 1, 2, \ldots,$ m do
        for $t = 1, 2, \ldots,$ T do
        $Z_{it} = h_t(x_i)$;
        end for
        $D' = D' \cup ((Z_{i1}, Z_{i2}, \ldots, Z_{iT}), y_i)$;
        end for
        $h' = \delta(D')$;
Output: $H(x) = h'(h_1(x), h_2(x), \ldots, h_T(x))$
```

Algorithm 2: Stacking fusion model.

improving the accuracy and recall of the model, different types of machine learning methods are used for stacking in the end, and it will give full play to the greatest advantages of each model to improve the stability of the model and to avoid the bias of a single model. Among the attacks, DoS attack and Probe use three types of machine learning methods, i.e., Random Forest, AdaBoost, and GBDT as first-level models. R2L uses Decision Tree, Random Forest, and GBDT as first-level models. U2R uses Decision Tree, AdaBoost, and GBDT as first-level models. The Decision Tree is used as the whole secondary model. Compared with other types of the integrated model, the Stacking algorithm has a stronger ability of nonlinear expression. When the meta-model with learning weights is increased, it will help to reduce the generalization error.

Stacking is a hierarchical structure which contains at least two layers. The last layer contains only one model classifier, which is an ensemble of the above models. The computational complexity of Stacking depends on the selected basic classifiers, such as Decision Tree, SVM, or deep learning model. The models must be serial between layers but can be parallel within layers. So, the Stacking's model complexity is the sum of complexity of the most complex model in each layer:

$$o(\text{Stacking}) = \max\left[o\left(M_{11}\right), \ldots, o\left(M_{1m}\right)\right] + \cdots + o\left(M_n\right). \tag{14}$$

The overall framework of the model proposed in Figure 2 consists of three steps.

5. Experiment and Analysis

5.1. Attack Description in Dataset. This paper uses the KDD CUP 99 [31] and NSL-KDD [32] datasets to demonstrate the superiority of Stacking with the DT-REF method. KDD CUP 99 was created in the United States of America by the Defense Advanced Research Projects Agency (DARPA) of the US Department of Defenses' MIT Lincoln Laboratory [33].

NSL-KDD improves and optimizes the existing defects in KDD CUP 99 which are mentioned in [32]. Training set and test set of the NSL-KDD has a rational distribution ratio. The redundant records in the original dataset are solved, so it will not cause the classifier to bias records with a large amount of data. Therefore, the achievements in different papers on the NSL-KDD can be reasonably compared. Because KDD CUP 99 is very classic in the field, this paper retains the experiment on it.

The NSL-KDD and KDD CUP 99 datasets divide various attacks into four categories and are described as follows:

(i) DoS: DoS attacks disturb normal access to network services. Its main purpose is to make network resources unable to serve normal network requests by completely utilizing memory resources, thereby it denies users to access to services. In addition, DoS includes such attacks as smurf, neptune, ping of death, back, and so on.

(ii) Probe: Before launching an attack, hackers first need to scan the target website to collect and analyze target information. In addition, the information is searched and used in order to find the known vulnerabilities in target objects. Hackers will use the information to launch accurate and efficient attacks. These scanning and analysis tasks are called Probe, for example, satan, portsweep, nmap, and so on.

(iii) Remote to local (R2L): An attack that illegally accesses local resources from an external network is called R2L. Through the Internet, users can send the packets to remote terminals. But, they have no right to disclose vulnerabilities and take advantage of the permissions that local users possess on the computer. In addition, R2L includes ftp_write, phf, multihop, and so on.

(iv) User to root (U2R): This type of attack is generally referred to as privilege escalating and an attack that illegally promote common users' permissions to administrator privileges. Attackers use the vulnerabilities to increase their ordinary permissions to root permissions. U2R comprises perl, rootkit, and so on.

5.2. Evaluation Indicators. The key to measuring the quality of an intrusion detection system is whether it has a high detection rate and a low FP rate. Therefore, in order to evaluate the performance of the intrusion detection model proposed, in this paper, accuracy (ACC), precision (PRE), detection rate (DR), and F1-Score (f_1) are selected. The accuracy rate is used to measure the accuracy of the classification. The detection rate is used to measure the percentage of abnormal behaviors that are detected. The accuracy is used to measure how many use cases in the test results are abnormal behaviors. In addition, the f_1 is used to comprehensively judge DR and ACC. The corresponding calculation formulas are shown as follows:

$$\begin{aligned}
\text{ACC} &= \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}},\\
\text{PRE} &= \frac{\text{TP}}{\text{TP} + \text{FP}},\\
\text{DR} &= \frac{\text{TP}}{\text{TP} + \text{FN}},\\
f_1 &= \frac{2 * \text{DR} * \text{PRE}}{\text{DR} + \text{PRE}},
\end{aligned} \tag{15}$$

where True Positive (TP) is the number of positive samples correctly identified, True Negative (TN) is the number of positive samples correctly identified, False Positive (FP) is the number of positive samples identified by mistake, and False Negative (FN) is misidentified that the number of negative samples.

Due to the influence of noise, there will be a certain bias between the training set and the test set, which often causes the model to obviously perform very well on the training set, but the performance on the test set is greatly reduced. For cross-validation, this paper uses 2-, 5-, 10-, 30-, and 50-fold cross-validation for comparison.

5.3. Analysis of Experimental Results. The environment and tool information needed for experiment are shown in Table 3.

Each step of RFECV yields a specific number of characteristics of ACC so that the minimum number of features can be obtained as ACC reaches a maximum. At the same time, RFECV can pick out the selected features. Therefore, RFECV is used to select the optimal number of features, as shown in Figure 3. The features of the final selection of DoS, Probe, R2L, and U2R are shown in Table 4.

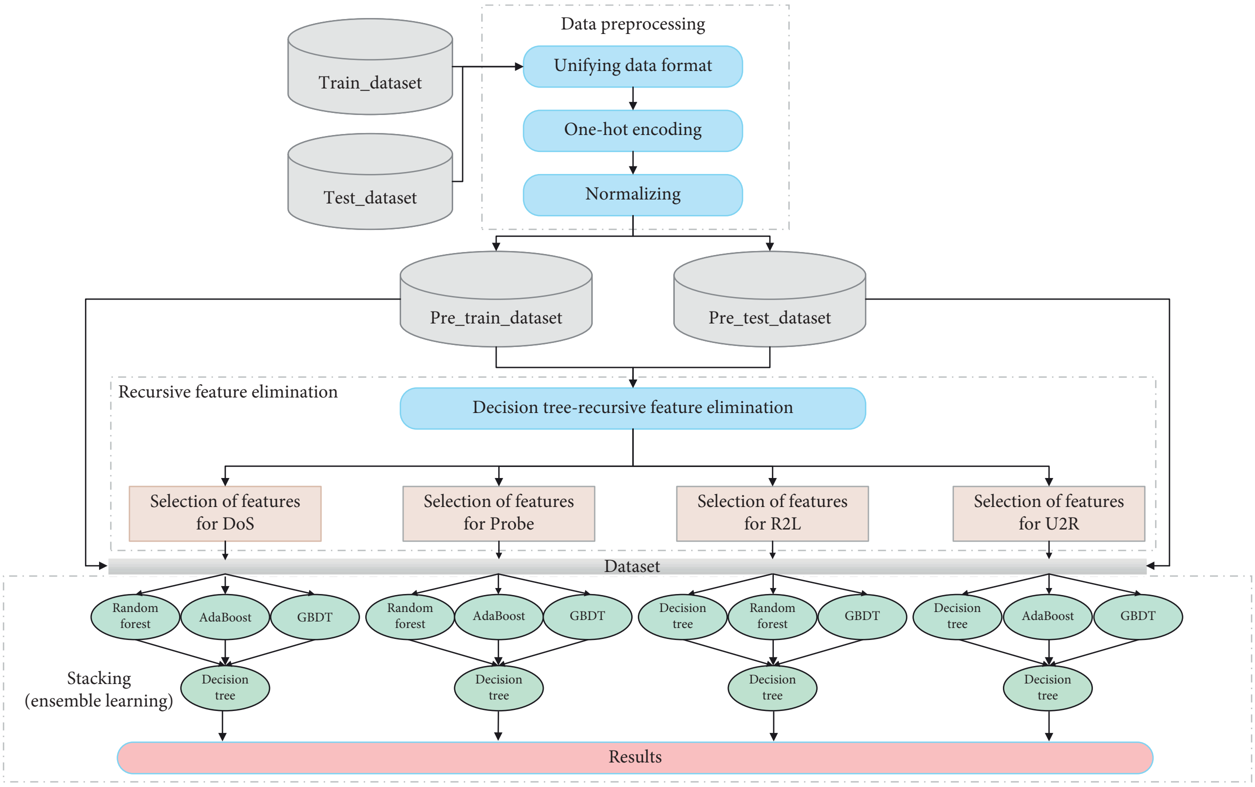


Figure 2: The overall framework of the proposed model.

Table 3: Experimental environment.

Project	Environment
Operating system	Ubuntu 18.04 LTS
CPU	I5-9300H
Memory	16G
Python	3.6.8
Scikit-learn	0.21.2
Anaconda	4.8.3
Mlxtend	0.17.2

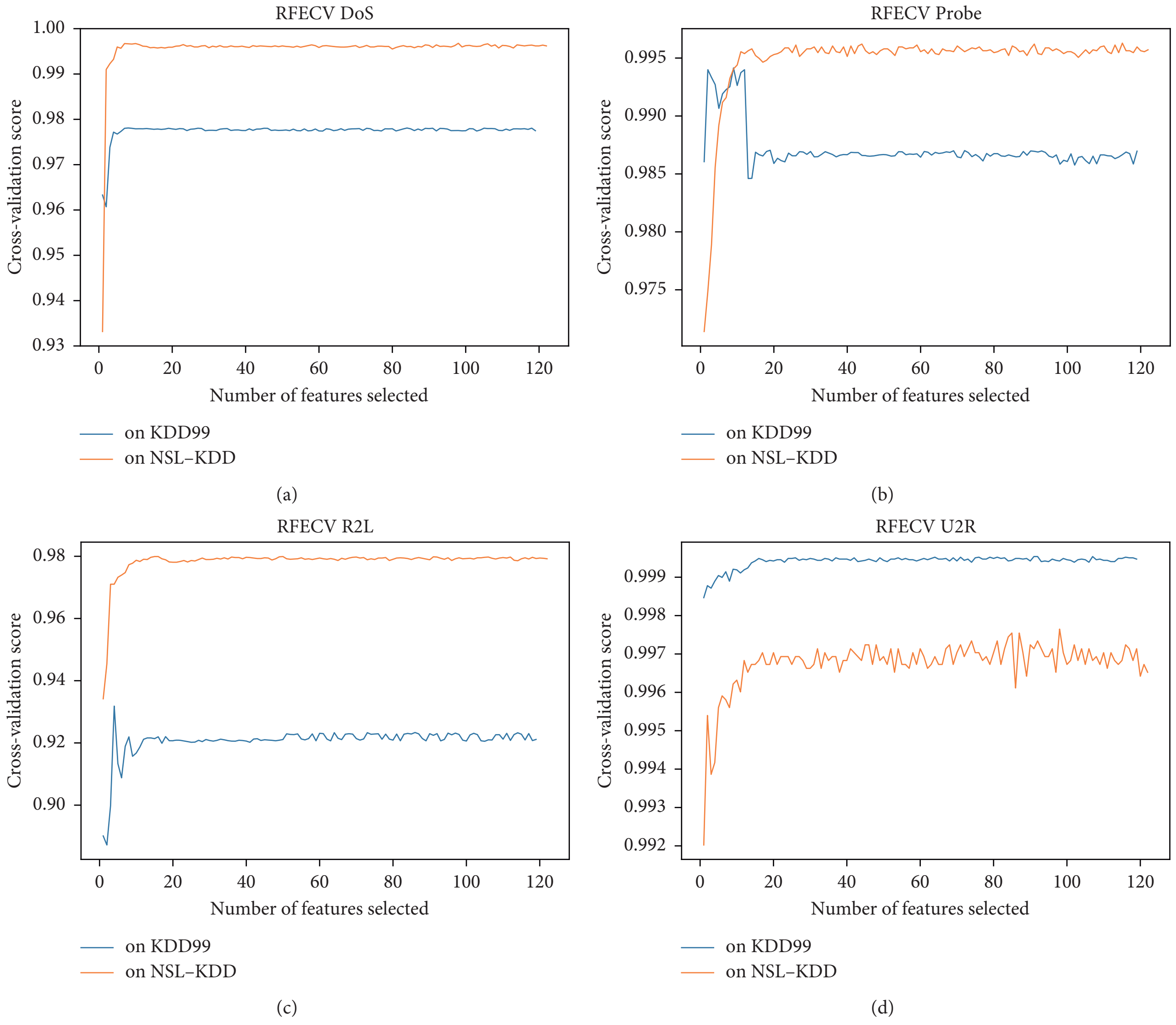
The x-axis of Figure 3 represents the number of features currently selected by REFCV in the recursive process, and the y-axis represents the cross-validation score under the number. DT-RFE calculates the ranking of all features' importance of result under the current goal. It can be seen that different optimal numbers of features are required for the prediction of different attack types. In other words, the predicting different types of targets requires the different features to get better results. With the ranking and optimal number of features, we can obtain the required features like Table 4. As shown in Figure 3, we conduct experiments on NSL-KDD and KDD CUP 99, respectively. On the result of feature selection, we take NSL-KDD as an example to show it and as shown in Table 4.

Firstly, the feature learning is performed by using the marked training set as input to the IDS. After several experiments, the optimal parameters were selected and saved, and the IDS model trained was retained. Then, the remaining data form a test dataset to measure the detection accuracy the model preforms on each attack. The experimental results on NSL-KDD and KDD CUP 99 datasets are shown in Figure 4 and Tables 5 and 6.

Tables 5 and 6 show the accuracy of each model for all attack types, and the last column is the accuracy of ensemble learning (Stacking). It can be seen that the traditional Logistic Regression and SVM have low DR for the four attack types, and the classification accuracy is not high. The AdaBoost algorithm is a bit worse than other ensemble learning algorithms. In addition, Stacking has the highest accuracy among all models.

Figure 4 shows a more detailed visualization in Table 6. It not only includes accuracy of each model but also corresponding precision, detection rate, F1-score, and 5-fold accuracy. These indicators can evaluate the model comprehensively. It can be seen that any score of our Stacking method is higher than the other models. In Figure 4, for the R2L attack type, no matter which algorithm is selected, except ACC, other detection indicators are not high. The experimental results in using the stacking algorithm show that our method can maintain high detection accuracy for all kinds of attack. Except for the lower accuracy of R2L, the others reach more than 99%, especially for small samples.

In Figure 4, we show the results of the 5-fold cross-validation, and the results obtained on the 2-, 10-, 30-, and 50-fold are similar. Therefore, only the most representative 5-fold cross-validation results are shown in the figure. This shows that our method not only achieves better results on the training set but also achieves better results on the unlearned testing set. It reflects that our method is not only



(a)

(b)

(c)

(d)

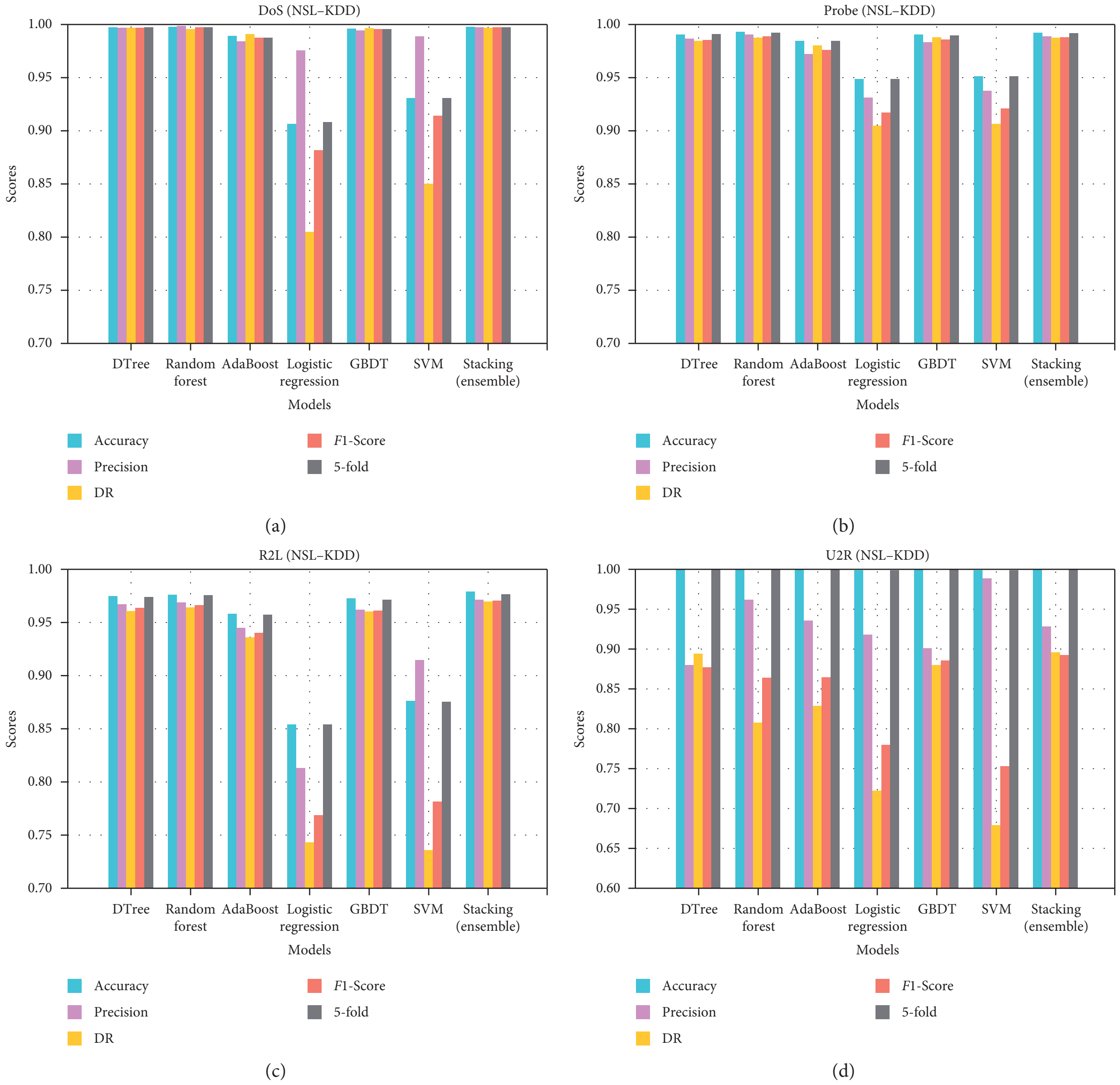
Figure 3: Graphs generated by RFECV based on the ACC of specific quantitative features: (a) RFECV DoS, (b) RFECV Probe, (c) RFECV R2L, and (d) RFECV U2R.

Table 4: Features of DoS, Probe, R2L, and U2R on NSL-KDD

DoS	’dst_host_same_srv_rate’; ’dst_host_serror_rate’; ’num_compromised’; ’same_srv_rate’; ’diff_srv_rate’; ’dst_host_count’; ’dst_host_srv_serror_rate’; ’ecr_i’; ’RSTR’; ’wrong_fragment’; ’dst_bytes’; ’src_bytes.’
Probe	’src_bytes’; ’telnet’; ’smtp’; ’private’; ’http’; ’ftp_data’; ’finger’; ’dst_host_rerror_rate’; ’dst_host_same_src_port_rate’; ’dst_host_diff_srv_rate’; ’dst_host_same_srv_rate’; ’rerror_rate’; ’dst_bytes.’
R2L	’duration’; ’imap4’; ’ftp_data’; ’dst_host_srv_diff_host_rate’; ’dst_host_same_src_port_rate’; ’dst_host_same_srv_rate’; ’dst_host_srv_count’; ’dst_host_count’; ’num_access_files’; ’num_failed_logins’; ’hot’; ’dst_bytes’; ’src_bytes.’
U2R	’duration’; ’dst_host_srv_diff_host_rate’; ’dst_host_count’; ’srv_count’; ’num_shells’; ’num_file_creations’; ’root_shell’; ’dst_bytes’; ’dst_host_same_srv_rate’; ’hot’; ’src_bytes.’

accurate in each category but also has good generalization performance.

Table 7 and Figure 5 show the classification accuracy of our method and these methods in [8, 9, 16, 18, 25, 34]. The latest paper [9] does not provide the accuracy of each category, so their overall accuracy is used as the average. Our proposed Stacking with the DT-RFE method outperforms the other methods in terms of the ACC of DoS, Probe, and U2L, especially U2R. Compared with the previous methods, this paper divides four different subsets on the NSL-KDD and KDD CUP 99 datasets for training and extracts unique feature attributes for each type of attack. The detection result is shown in Figure 5, but the simple machine learning algorithm is as high as 80%. In detection of U2R and R2L



(a)

(b)

(c)

(d)

Figure 4: Comparison results in ACC, PRE, DR, and F1-score: (a) DoS, (b) Probe, (c) R2L, and (d) U2R.

Table 5: Accuracy of several machine learning algorithms on KDD CUP 99.

	Decision tree	Random forest	AdaBoost	Logistic regression	GBDT	SVM	Stacking
DoS	0.98944	0.9975	0.98899	0.90641	0.99598	0.9307	0.99755
Probe	0.99521	0.99316	0.98483	0.94881	0.99093	0.95145	0.99412
R2L	0.94047	0.97626	0.95816	0.85392	0.97253	0.87607	0.97920
U2R	0.99652	0.99693	0.99683	0.9955	0.99693	0.9955	0.99744

Table 6: Accuracy of several machine learning algorithms on NSL-KDD.

	Decision tree	Random forest	AdaBoost	Logistic regression	GBDT	SVM	Stacking
DoS	0.99802	0.99689	0.98682	0.92374	0.99451	0.95318	0.99744
Probe	0.98975	0.98231	0.98337	0.95274	0.98988	0.94291	0.99204
R2L	0.97004	0.96664	0.96385	0.90723	0.97924	0.91325	0.98207
U2R	0.98973	0.99601	0.98937	0.99644	0.99702	0.98961	0.99765

Table 7: Accuracy for each class of different methods.

Author	Dataset	DoS	Probe	R2L	U2R	Average
Our method	KDD CUP 99	0.9976	0.9941	0.9792	0.9974	**0.9921**
Our method	NSL-KDD	0.9974	0.9920	**0.9821**	**0.9977**	**0.9923**
Hussain	NSL-KDD	**1.0000**	**0.9990**	0.7740	0.8860	0.9148
Akashdeep	KDD CUP 99	0.9993	0.9879	0.9190	0.8660	0.9431
Jia	KDD CUP 99	0.9990	0.9818	0.9706	0.8182	0.9424
Jia	NSL-KDD	0.9867	0.9773	0.9694	0.8182	0.9379
Sun	KDD CUP 99	0.9741	0.9313	0.1124	0.2542	0.5680
Andresini	KDD CUP 99	—	—	—	—	0.9249
Jiang	NSL-KDD	0.9621	0.6856	0.6045	0.6132	0.8358

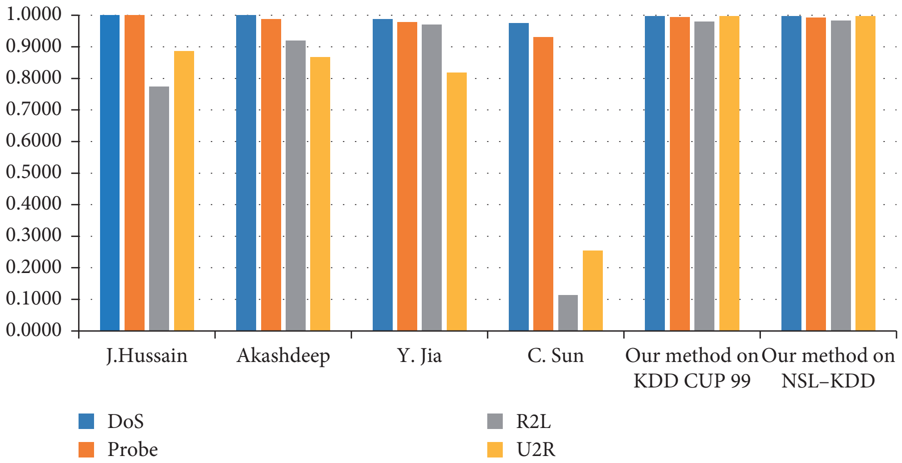


Figure 5: Our method compared with other previous methods.

features, it far exceeds the detection results of the algorithms proposed by Sun et al. [34] and Hussain et al. [16].

Although Akashdeep et al. [18] also proposed feature reduction by feature ordering based on information gain and correlation, however, the preprocessing work is done manually from the levels of information gain and correlation to identify useful and useless features. Jia et al. [25] classified records of NSL-KDD and KDD CUP 99 datasets with a deep learning method, but the U2R testing results are lower.

Overall, our method has better performance in NSL-KDD and KDD CUP 99 datasets. Our method has higher accuracy in the detection of attacks. In order to show our improvement, the average accuracy of multiple attacks is calculated in Table 7. In addition, Figure 5 shows the visualization in Table 7, which obviously reflects that our method has a very balanced accuracy for each type of attack.

6. Conclusions and Future Scope

In this paper, a novel intelligent intrusion detection system based on Stacking is proposed, and it used a DT-RFE algorithm to extract less features. Our method can improve and optimize the dataset and increase the resource utilization through deleting uncorrelated and redundant records. When the accuracy of a single machine learning model is difficult to improve, Stacking can be used to stack machine learning models to improve accuracy. Overall, our method has higher DR and lower FPR and has higher recognition accuracy for each type of attack. The designed IDS shows that feature reduction can reduce the system size and shorten training time. The lower time complexity resulted through the above measures can make the system performance better. Our method in the IDS can execute security functions in networks, organizations, and social groups with critical security.

Although the current work can optimize the feature set and acquire some excellent achievements, the R2L detection rate is still less than ideal, and the accuracy, DR, and FPR are relatively low. It can improve the disadvantages through a variety of extensions. According to the interactive network intrusion detection data 3D visualization method proposed in [35], it geometrically visualizes the relationship between every two different types of network traffic so as to explain the composition of the intrusion detection dataset in a more intuitive way. By using a fast convergence learning algorithm to fast check DR, the performance of the system can be further improved. And how we apply stacking into unsupervised learning is also a future work that needs to be explored. The above research will be the focus of our future work.

Data Availability

The data used to support the results of this study are provided in given links in this article. The datasets can be obtained from the online websites [36, 37]: KDD

CUP 99 dataset, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html [36], and NSL-KDD dataset, https://www.unb.ca/cic/datasets/nsl.html [37].

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was funded by the Scientific Research Foundation of Shandong University of Science and Technology for Recruited Talents, grant no. 0104060511314, and the National Key Research and Development Program of China, grant no. 2017YFC0804406.

References

[1] D. E. Denning, "An intrusion-detection model," *IEEE Transactions on Software Engineering*, vol. 13, no. 2, pp. 222–232, 1987.

[2] National Computer Network Emergency Technical Processing Coordination Center, *The 2018 China Internet Network Security Report*, People's Posts and Telecommunications Press, Beijing, China, 2019.

[3] L. N. Tidjon, M. Frappier, and A. Mammar, "Intrusion detection systems: a cross-domain overview," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3639–3681, 2019.

[4] J. H. Lee, J. H. Lee, S. G. Sohn et al., "Effective value of decision tree with KDD 99 intrusion detection datasets for intrusion detection system," in *Proceedings of the 10th International Conference on Advanced Communication Technology*, pp. 1170–1175, Gangwon-Do, Republic of Kore, February 2008.

[5] N. B. Amor, S. Benferhat, and Z. Elouedi, "Naive bayes vs. decision trees in intrusion detection systems," in *Proceedings of the 2004 ACM Symposium on Applied Computing*, pp. 420–424, Nicosia, Cyprus, March 2004.

[6] Y. Liao and V. R. Vemuri, "Use of k-nearest neighbor classifier for intrusion detection," *Computers & Security*, vol. 21, no. 5, pp. 439–448, 2002.

[7] M. Shojafar, R. Taheri, Z. Pooranian, R. Javidan, A. Miri, and Y. Jararweh, "Automatic clustering of attacks in intrusion detection systems," in *Proceedings of the 2019 IEEE/ACS 16th International Conference on Computer Systems and Applications (AICCSA)*, pp. 1–8, Abu Dhabi, UAE, May 2019.

[8] G. Andresini, A. Appice, N. D. Mauro, C. Loglisci, and D. Malerba, "Multi-channel deep feature learning for intrusion detection," *IEEE Access*, vol. 8, pp. 53346–53359, 2020.

[9] K. Jiang, W. Wang, A. Wang, and H. Wu, "Network intrusion detection combined hybrid sampling with deep hierarchical network," *IEEE Access*, vol. 8, pp. 32464–32476, 2020.

[10] W. Liang, K.-C. Li, J. Long, X. Kui, and A. Y. Zomaya, "An industrial network intrusion detection algorithm based on multifeature data clustering optimization model," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 3, pp. 2063–2071, 2020.

[11] H. Yao, D. Fu, P. Zhang, M. Li, and Y. Liu, "MSML: a novel multilevel semi-supervised machine learning framework for intrusion detection system," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 1949–1959, 2019.

[12] K. Kim, M. E. Aminanto, and Tanuwidjaja, *Network Intrusion Detection Using Deep Learning: A Feature Learning Approach*, Springer, Berlin, Germany, 2018.

[13] V. P. Singh, R. Pathak, S. Tiwari, and K. Kaur, "Content-based image retrieval based on supervised learning and statistical-based moments," *Modern Physics Letters B*, vol. 33, no. 19, Article ID 1950213, 2019.

[14] A. A. Aburomman and M. B. Ibne Reaz, "A novel SVM-kNN-PSO ensemble method for intrusion detection system," *Applied Soft Computing*, vol. 38, pp. 360–372, 2016.

[15] P.-F. Marteau, "Sequence covering for efficient host-based intrusion detection," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 4, pp. 994–1006, 2019.

[16] J. Hussain, S. Lalmuanawma, and L. Chhakchhuak, "A two-stage hybrid classification technique for network intrusion detection system," *International Journal of Computational Intelligence Systems*, vol. 9, no. 5, pp. 863–875, 2016.

[17] A. A. Aburomman and M. B. I. Reaz, "Ensemble of binary SVM classifiers based on PCA and LDA feature extraction for intrusion detection," in *Proceedings of the 2016 IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, pp. 636–640, Xi'an, China, October 2016.

[18] I. M. Akashdeep, I. Manzoor, and N. Kumar, "A feature reduced intrusion detection system using ANN classifier," *Expert Systems with Applications*, vol. 88, pp. 249–257, 2017.

[19] B. Jia, Y. Ma, X. Huang, Z. Lin, and Y. Sun, "A novel real-time DDoS attack detection mechanism based on MDRA algorithm in big data," *Mathematical Problems in Engineering*, vol. 2016, Article ID 1467051, 10 pages, 2016.

[20] H. Musafer, A. Abuzneid, M. Faezipour et al., "An enhanced design of sparse autoencoder for latent features extraction based on trigonometric simplexes for network intrusion detection systems," *Electronics*, vol. 9, no. 2, p. 259, 2020.

[21] R. Taheri, M. Ghahramani, R. Javidan, M. Shojafar, Z. Pooranian, and M. Conti, "Similarity-based android malware detection using hamming distance of static binary features," *Future Generation Computer Systems*, vol. 105, pp. 230–247, 2020.

[22] M. Jin, Z. Xu, R. Li, and D. Wu, "Fuzzy ARTMAP ensemble based decision making and application," *Mathematical Problems in Engineering*, vol. 2013, Article ID 124263, 7 pages, 2013.

[23] S. Mohammadi and A. Namadchian, "A new deep learning approach for anomaly base IDS using memetic classifier," *International Journal of Computers Communications & Control*, vol. 12, no. 5, pp. 677–688, 2017.

[24] J. Gu, L. Wang, H. Wang, and S. Wang, "A novel approach to intrusion detection using SVM ensemble with feature augmentation," *Computers & Security*, vol. 86, pp. 53–62, 2019.

[25] Y. Jia, M. Wang, and Y. Wang, "Network intrusion detection algorithm based on deep neural network," *IET Information Security*, vol. 13, no. 1, pp. 48–53, 2019.

[26] Z. T. Jiang and T. S. Z. Zhou, "Intrusion detection method based on ADBN," *Application Research of Computers*, vol. 37, no. 9, p. 46, 2020.

[27] M. X. Lu, G. Z. Du, and Z. X. Ji, "Network intrusion detection based on deep transfer learning," *Application Research of Computers*, vol. 37, no. 9, p. 44, 2019.

[28] D. H. Wolpert, "Stacked generalization," *Neural Networks*, vol. 5, no. 2, pp. 241–259, 1992.

[29] A. Gupta, D. Singh, and M. Kaur, "An efficient image encryption using non-dominated sorting genetic algorithm-III

based 4-D chaotic maps," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 3, pp. 1309–1324, 2020.

[30] M. Kaur, H. K. Gianey, D. Singh, and M. Sabharwal, "Multi-objective differential evolution based random forest for e-health applications," *Modern Physics Letters B*, vol. 33, no. 5, Article ID 1950022, 2019.

[31] K. Siddique, Z. Akhtar, and F. Kim, "KDD Cup 99 data sets: a perspective on the role of data sets in network intrusion detection research," *Computer*, vol. 52, no. 2, pp. 41–51, 2019.

[32] M. Tavallaee, E. Bagheri, W. Lu et al., "A detailed analysis of the KDD CUP 99 data set," in *Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, pp. 1–6, Ottawa, Canada, July 2009.

[33] M. T. Lincoln Laboratory, https://www.ll.mit.edu/.

[34] C. Sun, K. Lv, C. Z. Hu et al., "A double-layer detection and classification approach for network attacks," in *Proceedings of the 27th International Conference on Computer Communication and Networks (ICCCN)*, pp. 1–8, Hangzhou, China, July 2018.

[35] W. Zong, Y.-W. Chow, and W. Susilo, "Interactive three-dimensional visualization of network intrusion detection data for machine learning," *Future Generation Computer Systems*, vol. 102, pp. 292–306, 2020.

[36] KDD CUP 99 Dataset, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[37] NSL-KDD Dataset, https://www.unb.ca/cic/datasets/nsl.html.

Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 5473501, 24 pages
https://doi.org/10.1155/2020/5473501

Research Article

Exploring Further Advantages in an Alternative Formulation for the Set Covering Problem

Jose M. Lanza-Gutierrez,[1] **N. C. Caballe**,[2] **Broderick Crawford**,[3] **Ricardo Soto**,[3] **Juan A. Gomez-Pulido**,[4] **and Fernando Paredes**[5]

[1] *Carlos III University of Madrid, Electronic Technology Department, Leganés, Spain*
[2] *University of Alcalá, Physics and Mathematics Department, Faculty of Sciences, Alcalá de Henares, Spain*
[3] *Pontificia Universidad Católica de Valparaíso, Escuela de Ingeniería Informática, Valparaíso, Chile*
[4] *University of Extremadura, School of Technology, Cáceres, Spain*
[5] *Universidad Diego Portales, Escuela de Ingeniería Industrial, Santiago de Chile, Chile*

Correspondence should be addressed to Jose M. Lanza-Gutierrez; jlanza@ing.uc3m.es

Received 28 February 2020; Accepted 1 June 2020; Published 15 July 2020

Guest Editor: Dilbag Singh

The set covering problem (SCP) is an NP-complete optimization problem, fitting with many problems in engineering. The traditional SCP formulation does not directly address both solution unsatisfiability and set redundancy aspects. As a result, the solving methods have to control these aspects to avoid getting unfeasible and nonoptimized in cost solutions. In the last years, an alternative SCP formulation was proposed, directly covering both aspects. This alternative formulation received limited attention because managing both aspects is considered straightforward at this time. This paper questions whether there is some advantage in the alternative formulation, beyond addressing the two issues. Thus, two studies based on a metaheuristic approach are proposed to identify if there is any concept in the alternative formulation, which could be considered for enhancing a solving method considering the traditional SCP formulation. As a result, the authors conclude that there are concepts from the alternative formulation, which could be applied for guiding the search process and for designing heuristic feasibilit\y operators. Thus, such concepts could be recommended for designing state-of-the-art algorithms addressing the traditional SCP formulation.

1. Introduction

The set covering problem (SCP) is a classical problem shown to be NP-complete by Karp [1] and whose optimization version is NP-hard. Although this is a traditional problem, SCP is widely considered in the current scientific literature because it fits problems in relevant areas, such as engineering, vehicle routing, medical domain, and facilities allocation (see e.g., [2–6]).

Most contributions in the SCP field considered the traditional SCP formulation introduced by Chvatal [7] and defined as follows. Let E be a set of m objects and let S be a collection of n subsets of E, where each subset has a nonnegative cost associated. Then, the purpose of SCP is to get a minimum cost family of subsets $X \subseteq S$, such that each element of E belongs to at least one subset of the family X. This traditional formulation does not directly deal with two aspects: solution unsatisfiability and set redundancy. The solution unsatisfiability aspect is related to the possibility of generating unfeasible solutions during the search. The set redundancy aspect is related to the possibility of generating nonoptimized solutions in cost, including redundant components (subsets). The noninclusion of these two aspects in the formulation means that the solving method has to address them to ensure good performance.

In the last years, Bilal et al. [8] proposed an alternative SCP formulation. Its main contribution was that both redundant sets and unfeasible solutions were directly penalized in the fitness function. Therefore, the solving method does not have to control such aspects in contrast to the traditional SCP formulation. Nevertheless, the contribution of the alternative formulation becomes questionable due to two main issues. First, Vasko et al. [9] demonstrated that the calculation effort to remove redundant components from an SCP solution is almost negligible. Thus, including a redundancy removal operator in a solving method addressing the traditional formulation does not increase excessively the computational cost. Second, there are simple methods for transforming unfeasible solutions into feasible ones, such as the one proposed by Beasley and Chu [10]. As a consequence, alternative formulation seems not to be advantageous.

Analysing the work proposed in the alternative formulation in Bilal et al. [8], they compared the alternative formulation to the traditional one. To this end, they solved the standard Beasley's OR library [11] through two algorithms: a simple descent heuristic (DH) addressing the alternative formulation and a standard greedy heuristic (GH) addressing the traditional one. As a result, DH outperformed GH, which is valuable for justifying the alternative formulation. However, Vasko et al. [9] later applied the same GH using the traditional formulation on the same instances, but included a simple redundancy removal operator, obtaining better results than the ones shown in Bilal et al. [8] for DH. Once again, the alternative formulation seems to have questionable merit. However, this comparison initiated in Bilal et al. [8] might have some limitations:

(i) The heuristic techniques considered might not be the most appropriate according to the current state of the art, in which metaheuristics, especially swarm intelligence algorithms (SIAs), provide the best results in general.

(ii) The authors independently compared the two formulations using different algorithms. This focus could be correct because the algorithm best suited for a formulation could be very different, even in type, from the algorithm for the other formulation. However, there is no study combining aspects from the two formulations that could provide some advantages for the same solving method.

(iii) The authors did not consider any statistical method for comparing both formulations. Instead, they only compare the average solution obtained.

On this basis, this paper questions whether there is any advantage from the concepts involved in the alternative formulation, beyond the novel problem formulation. This idea leads us to propose two studies focused on solution quality as a way to know if there is any concept in the alternative formulation, which could be considered for enhancing a method using the traditional formulation. To this end, the authors select two different metaheuristics adequate for the studies, although other metaheuristics could have been selected without loss of generality.

The only demonstration of the concept utility from the alternative formulation is valuable. This means that future solving methods could include these novel concept. This research focus implies that the authors are not focused on getting the best absolute results solving Beasley's OR library, but they understand that the solutions obtained should be reasonable, as will be discussed later.

The first study focuses on identifying if there is any concept in the alternative formulation, which could be applied for guiding the search process of a solving method addressing the traditional formulation. To this end, the authors generate two versions of the same SIA addressing the traditional formulation. In the first version, (i), the search process of SIA is guided by concepts from the traditional formulation. In the second version, (ii), the search process of SIA is guided by concepts from the alternative formulation. Further details of this first study are as follows:

(i) This study requires a solving method, whose search process is closely linked to the optimization problem. The ant colony optimization (ACO) algorithm meets this requirement, being very sensitive to the heuristic information operator designed based on the problem to solve. Thus, two heuristic information operators are considered: in (i), a usual operator based on the traditional formulation and in (ii), a novel operator based on concepts from the alternative formulation.

(ii) The two ACO approaches in (i) and (ii) include the same usual operator for removing redundant sets. No operator for transforming unfeasible solutions into feasible ones is considered because ACO does not generate them.

(iii) The two ACO approaches in (i) and (ii) are applied for solving Beasley's OR library. The results obtained were analysed through a widely accepted statistical method. Both approaches were tuned through the automatic method iterated F-Race, preventing errors from a manual method [12].

The second study focuses on identifying if there is any concept in the alternative formulation, which could be considered for designing the operators needed for transforming unfeasible solutions into feasible ones while removing redundant columns. Note that this type of operators is widely applied in most SIAs solving SCP. To this end, the authors generate two versions of the same SIA addressing the traditional formulation. In the first version, (iii), SIA considers the widely applied operator proposed in Beasley and Chu [10]. In the second version, (iv), SIA considers a novel operator inspired by concepts from the alternative formulation. Further details of this second study are as follows:

(i) The second study requires a solving method, which could generate unfeasible solutions. The artificial bee colony (ABC) is one of the many metaheuristics meeting this requirement. Thus, two different feasibility operators are considered in (iii) and (iv).

(ii) The two ABC approaches in (iii) and (iv) are applied for solving Beasley's OR library. The results obtained were analysed through a widely accepted statistical method. In this case, the parameter configuration is taken from the literature (see [13]) because the feasibility operator is considered as an external tool.

To summarize, the motivation of this research is to identify if there is any concept from the alternative formulation, which could be used to solve the traditional SCP problem. To the best of our knowledge, this is the first work performing this research. Figure 1 summarizes the main tasks performed in the two studies. The contributions to the field are as follows:

(i) A first study is proposed to identify concepts of interest in the alternative SCP formulation, which could be applied for guiding a search method addressing the traditional SCP formulation. The study results in that the gain concept in the alternative SCP formulation is useful for guiding the search, outperforming the results obtained by a usual heuristic information operator from the literature.

(ii) A second study is proposed to identify concepts of interest in the alternative SCP formulation, which could be considered for designing feasibility operators. The study results in that the gain concept in the alternative SCP formulation is useful for designing this type of operators, outperforming a usual operator in the literature. This contribution is especially interesting because this type of operators is widely applied in the metaheuristic SCP field as a black-box method.

These two contributions are especially interesting for future works, implementing techniques for solving the traditional SCP formulation. From the first study, it is shown that the gain concept in the alternative SCP formulation is useful for guiding the search. That means that this concept could be considered during the design of novel solving methods for SCP. From the second study, it is shown that the gain concept is useful for designing feasibility operators. That means that this concept could be considered to improve techniques already shown to be useful for solving the problem, as well as proposing novel feasibility operators. As stated before, the second future scope line is especially interesting because feasibility operators are widely applied in the literature as black-box techniques outside the solving method.

The rest of this paper is structured as follows. Section 2 discusses related work. In Section 3, a formal statement of both SCP formulations is provided. In Section 4, the main aspects of the ACO algorithm in the first study are discussed. Section 5 discusses the main aspects of the ABC algorithm in the second study. In Section 6, the experimental methodology followed is discussed and the solution quality results are analysed. Finally, Section 7 concludes and introduces future works. Table 1 includes a summary of the notation considered throughout this work.

2. Related Work

The literature about SCP is extensive. Some authors considered exact algorithms, such as branch-and-bound and branch-and-cut techniques [14–16] or linear programming [17–19]. More recently, Caprara et al. [20] compared several exact algorithms, concluding that the best exact technique was CPLEX.

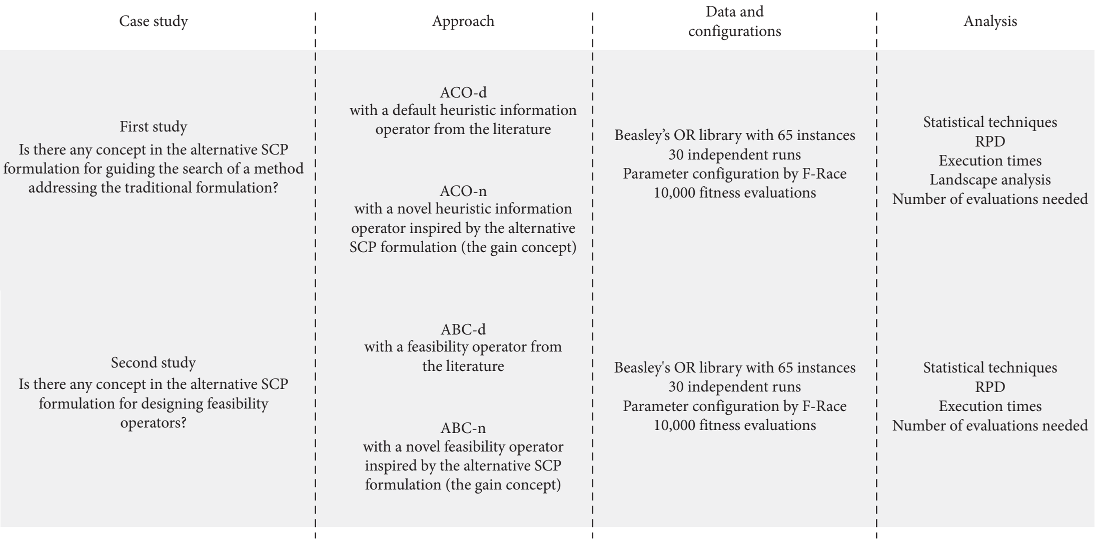
It is well known that exact techniques require excessive computer resources on large problems. Therefore, much effort was focused on exploring heuristic and metaheuristic algorithms, which could find near-optimal (or even optimal) solutions for large problems in reasonable computing time.

Starting from heuristic methods, Chvatal [7] applied a classical GH. Although GHs are simple and fast to implement, they seldom produce good quality solutions. Some researchers tried to improve GHs by adding randomness (see e.g., [9, 21–24]). Highly sophisticated heuristics based on Lagrangian relaxation were also considered, yielding very good solutions (see e.g., [20, 25–27]). From this brief review, it is shown that the number of proposals considering heuristic methods is limited for SCP in the last years. Note that in other optimization problems, the proposal of heuristics is usual (e.g., [28]). This situation is opposite for metaheuristics, acquiring a great relevance during the last decades [29]. Thus, metaheuristics were applied to fields as networking [30], biological ontology alignment [31], shop scheduling [32], chemical analysis [33], and image encryption [34].

Metaheuristics combine effectiveness exploring the search space and basic heuristic methods. Such techniques are usually split into three large groups: evolutionary algorithms (EAs), trajectory algorithms (TAs), and SIAs. To solve SCP, some authors considered EAs (see e.g., [10, 35–37]). Other authors applied TAs (see e.g., [38, 39]). However, the most widely applied metaheuristics for solving SCP are SIAs. Some examples are the artificial bee colony (ABC) algorithm [13, 40, 41], the ant colony optimization (ACO) algorithm [42–44], the firefly algorithm (FA) [45, 46], the teaching-learning-based optimization (TLBO) algorithm [47, 48], the electromagnetism-like (EM-like) algorithm [49, 50], the shuffled frog leaping algorithm (SFLA) [51], the fruit fly optimization algorithm (FFOA) [52], the cuckoo search algorithm (CSA) [53, 54], the cat swarm optimization (CSO) algorithm [55, 56], the jumping particle swarm optimization (JPSO) method [57], the black hole optimization [54, 58], and the monkey search algorithm [59].

Analysing the previous contributions according to the results obtained, exact algorithms provided excellent results, solving reduced SCP problems. Focusing on larger SCP problems addressed by approximate techniques (heuristics and metaheuristics), the authors check that heuristics do not provide as good results as the more sophisticated metaheuristics. Thus, the best results were usually obtained by SIAs. In this line, we should mention the valuable contributions of Naji-Azimi et al. [48, 49] and Balaji and Revathi [57] who got optimal or near-optimal solutions for classical SCP benchmarks.

All the works listed before have in common that they considered the traditional SCP formulation. On the contrary, the alternative formulation received limited attention. As far as the authors know, there are only two works



Case study	Approach	Data and configurations	Analysis
First study Is there any concept in the alternative SCP formulation for guiding the search of a method addressing the traditional formulation?	ACO-d with a default heuristic information operator from the literature ACO-n with a novel heuristic information operator inspired by the alternative SCP formulation (the gain concept)	Beasley's OR library with 65 instances 30 independent runs Parameter configuration by F-Race 10,000 fitness evaluations	Statistical techniques RPD Execution times Landscape analysis Number of evaluations needed
Second study Is there any concept in the alternative SCP formulation for designing feasibility operators?	ABC-d with a feasibility operator from the literature ABC-n with a novel feasibility operator inspired by the alternative SCP formulation (the gain concept)	Beasley's OR library with 65 instances 30 independent runs Parameter configuration by F-Race 10,000 fitness evaluations	Statistical techniques RPD Execution times Number of evaluations needed

Figure 1: Summary of the main tasks performed in the two studies in this paper.

Table 1: Notation.

$\|\cdot\|$	Cardinal of a given set.
α	Relative importance of pheromone trails, $\alpha \geq 0$.
α_i	The set of columns that row $i \in I$ covers, $\alpha_i \subseteq J$.
A	Binary matrix of m-rows and n-columns. The rows are the elements E of the universe and the columns are the subsets S.
abc_{add}	Percentage of columns to be added during the generation of a solution in ABC.
abc_{eli}	Percentage of columns to be removed during the generation of a solution in ABC.
$a\eta_j^t$	Heuristic factor of column $j \in R_{t-1}$ at step $t \geq 0$ for the alternative formulation, $\eta(a)_j^t > 0$.
$a_{i,j}$	Value in the cell (i, j) of A. It equals 1 if the j-th column covers the i-th row and 0 otherwise, $i \in I$, $j \in J$.
argmax$\{\cdot\}$	Point/points in which a function gets its maximum value/values.
argmin$\{\cdot\}$	Point/points in which a function gets its minimum value/values.
$a\phi_j^t$	The sum of the gains of covering the noncovered rows which could be covered by column $j \in R_{t-1}$ at $t \geq 0$.
β	Relative importance of heuristic information, $\beta \geq 0$.
C	Set of costs, $C = \{c_1, c_2, \ldots, c_n\}$.
$c\eta_j^t$	Heuristic factor of column $j \in R_{t-1}$ at step $t \geq 0$ for the classical formulation, $\eta(c)_j^t > 0$.
c_j	Cost associated to the j-th column, $c_j \in \mathbb{R}^+$, $j \in J$.
$c_{\min}(e_i)$	Cost of the cheapest set among the sets covering e_i, $c_{\min}(e_i) \in C$, $i \in I$.
$c\phi_j^t$	Number of noncovered rows which could be covered by column $j \in R_{t-1}$ at step $t \geq 0$.
Δ_t	Solution generated by an ant at step $t \geq 0$, $\Delta_t \subseteq J$.
Δ	A solution to the problem, $\Delta \subseteq J$.
ϵ	Ratio coefficient, $\epsilon \in (0, 1)$.
η_j^t	Heuristic factor of column $j \in R_{t-1}$ at step $t \geq 0$, $\eta_j^t > 0$.
E	Universe of elements, $E = \{e_1, e_2, \ldots, e_m\}$.
e_i	i-th element of E, $e_i \in E$, $i \in I$.
evals	Average fitness evaluations needed for reaching the best solution found during the exploration.
f_1	Fitness function of the classical SCP formulation, $f_1 \in \mathbb{R}^+$.
f_1'	Fitness function of the alternative SCP formulation, $f_1' \in \mathbb{R}$.
g_i	Gain of covering an element e_i, $g_i \in \mathbb{R}^+$, $i \in I$.
I	Row set, $I = \{1, 2, \ldots, m\}$.
I_j	Row set covered by column $j \in J$, $I_j \subseteq I$.
ipv	Percentage of improvement by considering the alternative approach of the algorithm instead of the default version.
J	Column set, $J = \{1, 2, \ldots, n\}$.
L	The best solution found from the beginning of the algorithm, $L \subseteq J$.
limit	Threshold value based on trials for deciding if a worker bee is transformed into a scout one in ABC, limit > 0.
m	Cardinal of E or number of rows.
n	Cardinal of S or number of columns.
n_w	Number of worker bees in ABC, $n_w \leq \text{ps}_{abc}$.
ω_j	Amount of pheromone put on column $j \in R_{t-1}$, $\omega_j \geq 0$.
ψ	Very small positive constant, $\psi \in \mathbb{R}^+$.
ps_{abc}	Population size in ACO.

Table 1: Continued.

ps_{aco}	Population size in ABC.
QMetric	Landscape solution quality metric, QMetric $\in [0,1]$.
q	Random number uniformly distributed in $[0,1]$.
q_0	Parameter determining the relative importance of exploitation versus exploration, $q_0 \in [0,1]$.
ρ	Pheromone persistence, $\rho \in [0,1)$.
R_t	Set of unselected columns in Δ_t, $R_t \subseteq J$, $t \geq 0$.
$\overline{\text{rpd}}$	Average RPD from a distribution, $\overline{\text{rpd}} \in [0,1]$.
$\overline{\text{rpd}}_a$	Average RPD of algorithm a, with $a \in \{\text{d}-\text{ACO}, \text{n}-\text{ACO}\}$.
$\overline{\text{rpd}}_b$	Average RPD of algorithm a, with $b \in \{\text{d}-\text{ACO}, \text{n}-\text{ACO}\}$.
$\overline{\text{rpd}}_c$	Average RPD of algorithm a, with $c \in \{\text{d}-\text{ABC}, \text{n}-\text{ABC}\}$.
$\overline{\text{rpd}}_d$	Average RPD of algorithm a, with $c \in \{\text{d}-\text{ABC}, \text{n}-\text{ABC}\}$.
$\text{rpd}_{\min}$	Minimum RPD from a distribution, $\text{rpd}_{\min} \in [0,1]$.
$\text{rpd}_{\max}$	Maximum RPD from a distribution, $\text{rpd}_{\max} \in [0,1]$.
S	Set of subsets, $S = \{s_1, s_2, \ldots, s_n\}$, $\cup S = E$.
s_j	j-th subset of S, $s_j \in S$, $s_j \subseteq E$, $j \in J$.
SRate	Landscape rate of success, SRate $\in [0,1]$.
SSpeed	Landscape speed of reaching a solution, SSpeed $\in [0,1]$.
τ_j	Pheromone trail of column $j \in R_{t-1}$, $\tau_j > 0$.
t	Construction step in ACO, $t \geq 1$.
$\overline{\text{time (s)}}_a$	Average execution time of algorithm a, with $a \in \{\text{d}-\text{ACO}, \text{n}-\text{ACO}\}$.
$\overline{\text{time (s)}}_b$	Average execution time of algorithm b, with $a \in \{\text{d}-\text{ACO}, \text{n}-\text{ACO}\}$.
$\overline{\text{time (s)}}_c$	Average execution time of algorithm c, with $c \in \{\text{d}-\text{ABC}, \text{n}-\text{ABC}\}$.
$\overline{\text{time (s)}}_d$	Average execution time of algorithm d, with $d \in \{\text{d}-\text{ABC}, \text{n}-\text{ABC}\}$.
V	Number of uncovered rows in a solution, $V \leq m$.
V_t	Set of uncovered rows in Δ_t, $V_t \subseteq I$, $t \geq 0$.
ξ_i	Number of columns in a solution Δ that cover the row $i \in I$, $\xi_i \leq n$.
X	An SCP solution expressed as binary vector, $X = \{x_1, x_2, \ldots, x_n\}$.
x_j	j-th element of X. It equals 1 if s_j is part of the solution X and 0 otherwise, $j \in J$.
y_i	Variable equaling 1 if the element e_i is covered in X and 0 otherwise, $i \in I$.
ζ_t	Column provided by SROM at step $t \geq 0$, $\zeta_t \in R_{t-1}$.
$\overline{z}$	Average cost obtained from a distribution, $\overline{z} \in \mathbb{R}^+$.
$z_{\max}$	Maximum cost obtained from a distribution, $z_{\max} \in \mathbb{R}^+$.
$z_{\min}$	Minimum cost obtained from a distribution, $z_{\min} \in \mathbb{R}^+$.
z_{opt}	Optimum solution of a given instance, $z_{\text{opt}} \in \mathbb{R}^+$.
z_t	Column selected at step $t \geq 0$, $z_t \in R_{t-1}$.

considering the alternative formulation. In Bilal et al. [60], they solved an SCP variant through an iterated tabu-search metaheuristic. In Crawford et al. [61], they compared the results obtained solving the traditional and alternative formulations through the ACO algorithm.

The research presented in this paper was inspired by a very preliminary work discussed before (see Crawford et al. [61]). In this contribution, there is no study regarding the existence of concepts in the alternative formulation, which could be considered for solving methods addressing the traditional formulation. In Lanza-Gutierrez et al. [56], the authors applied an SIA to solve SCP by a CSO algorithm but with a completely different approach.

3. Set Covering Problem Statements

Let $I = \{1, 2, \ldots, m\}$ and $J = \{1, 2, \ldots, n\}$ be the row and column sets, respectively. Let $E = \{e_1, e_2, \ldots, e_m\}$ be a universe of m elements and let $S = \{s_1, s_2, \ldots, s_n\}$ be a collection of n subsets of E, such that $s_j \subseteq E$ and $\cup S = E$, with $j \in J$. Each subset s_j has a non-negative cost associated $c_j \in C$, where $C = \{c_1, c_2, \ldots, c_n\}$.

The optimization problem is formally defined by assuming a binary matrix A of m-rows and n-columns, where the rows are the elements of the universe and the columns are the subsets. Let a_{ij} be the value in the cell (i, j) of A given by

$$a_{ij} = \begin{cases} 1, & \text{if } e_i \in s_j, \\ 0, & \text{otherwise,} \end{cases} \tag{1}$$

for $i \in I$ and $j \in J$, where $e_i \in E$. Thus,

$$A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix}. \tag{2}$$

The objective of SCP is to find a subset of S covering (containing) all the elements of E at a minimal cost. A solution to SCP is usually expressed as a binary vector $X = \{x_1, x_2, \ldots, x_n\}$, where

$$x_j = \begin{cases} 1, & \text{if the set } s_j \text{ is part of the solution,} \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

Then, the cost of the solution X is

$$\sum_{j \in J} c_j x_j. \tag{4}$$

Next, we give a formal statement of the two SCP formulations.

3.1. Traditional Formulation. The SCP fitness function is

$$f_1 = \sum_{j\in J} c_j x_j, \quad f_1 \in \mathbb{R}^+. \tag{5}$$

Then, given m elements and n subsets, the objective is to find a collection of subsets to

$$\min(f_1), \tag{6}$$

subject to

$$\sum_{j\in J} a_{ij} x_j \geq 1, \quad \forall i \in I, \tag{7}$$

$$x_j \in \{0,1\}, \quad \forall j \in J. \tag{8}$$

The constraint in equation (7) ensures that each row is covered by at least one column. If this constraint is not satisfied, the solution is considered unfeasible. The constraint in equation (8) is only for the integrity of the mathematical programming. Hence, this equation does not need to be addressed as a constraint in heuristic approaches.

3.2. Alternative Formulation. In this formulation, covering an element is identified with collecting a gain at a given cost. Let $c_{\min}(e_i) \in C$ be the cost of the cheapest set among the sets covering the element e_i given by

$$c_{\min}(e_i) = \arg\min_{c_j\in C}\{a_{ij} x_j c_j > 0\}, \quad \forall j \in J, \tag{9}$$

where argmin{·} provides the point/points in which a function gets its minimum value/values. Then, the gain $g_i \in \mathbb{R}^+$ of covering an element e_i is

$$g_i = c_{\min}(e_i) + \psi, \tag{10}$$

where $\psi \in \mathbb{R}^+$ is a very small positive constant. Based on this gain concept, the SCP fitness function is

$$f_1' = \sum_{i\in I} g_i y_i - \sum_{j\in J} c_j x_j, \quad f_1' \in \mathbb{R}, \tag{11}$$

where

$$y_i = \begin{cases} 1, & \text{if } e_i \text{ is covered in the solution } X, \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

Then, given m elements and n subsets, the objective is to find a collection of subsets to

$$\max(f_1'), \tag{13}$$

subject to

$$y_i \leq \sum_{j\in J} a_{i,j} x_j, \quad \forall i \in I, \tag{14}$$

$$x_j,\ y_i \in \{0,1\}, \quad \forall i \in I, \forall j \in J. \tag{15}$$

The constraints in equations (14) and (15) are only for the integrity of the mathematical programming. According to this formulation, there are no unfeasible solutions as happens with the traditional formulation. Note that unfeasible solutions still exist for the problem. However, the alternative formulation penalizes such issue instead of discarding the solution. Moreover, it also penalizes directly redundant sets beyond having a higher cost as occurs for the traditional formulation. Thus, the use of redundancy removal operators is not needed, in contrast to the traditional formulation, where it is highly recommended.

4. Ant Colony Optimization

The ACO algorithm is inspired by ant colony behaviours. The ACO process is focused on the search of the optimal path in a graph based on an artificial ant colony. Thus, ants work cooperatively and communicate through heuristic information depending on the problem and pheromone trails. Pheromone trails are a type of distributed information, which is dynamically updated by the ants. Pheromones keep the experience gained during the search process while remarking promising areas of the search space.

Let $\Delta_{t-1} \subseteq J$ be the solution generated by an ant at construction step $t-1 \geq 0$. Let $V_{t-1} \subseteq I$ be the set of uncovered rows in Δ_{t-1}. Let $R_{t-1} \subseteq J$ be the set of unselected columns in Δ_{t-1}. Reviewing the scientific literature [42, 62], a usual heuristic information expression for a column $j \in R_{t-1}$ at step t is

$$c\eta_j^t = \frac{c\phi_j^t}{c_j}, \tag{16}$$

where $c\phi_j^t$ is the number of noncovered rows in Δ_{t-1}, which could be covered by column j at step t. This value is

$$c\phi_j^t = \left\| I_j \cap V_{t-1} \right\|, \tag{17}$$

where $\|\cdot\|$ is the cardinal of a set and I_j denotes the row set covered by column j, for $j \in J$ and $I_j \subseteq I$.

In this work, we propose a heuristic information inspired by the gain concept from the alternative formulation introduced in Section 3.2 as

$$a\eta_j^t = \frac{a\phi_j^t}{c_j}, \tag{18}$$

where $a\phi_j^t$ is the sum of the gains of covering the noncovered rows in Δ_{t-1} by column j at step t. Thus,

$$a\phi_j^t = \sum_{i\in\{I_j \cap V_{t-1}\}} g_i, \tag{19}$$

where g_i is given in equation (10).

To simplify the notation, we define the heuristic information for a column j at step t based on whether we consider the traditional formulation or the alternative one. That is,

$$\eta_j^t = \begin{cases} c\eta_j^t, & \text{if traditional formulation,} \\ a\eta_j^t, & \text{if alternative formulation.} \end{cases} \tag{20}$$

Algorithm 1 shows the procedure of a general ACO. Next, the main steps are detailed.

(1) *Initialization*: in the beginning, we propose to preprocess the SCP instances by using column domination and column inclusion [18]. Next, the algorithm parameters are initialized. Traditionally, ACO algorithms do not include an initialization step to generate the ps_{aco} solutions in the population. Instead, pheromone trails are randomly assigned and then solutions are generated according to this random information. That means that the algorithm could need to run some iterations before having the right information about the solution component quality. At this point, we propose to include a greedy population initialization step in ACO based on Lu and Vasko [48]. This step corresponds to line 1 of Algorithm 1.

(2) *Solution construction method*: each ant starts with an empty solution where columns are added iteratively until all rows are covered. Consequently, this strategy causes all solutions generated to be feasible. Most ACO-based algorithms consider a similar state transition rule, preferring solution components with high pheromone and heuristic values (see e.g., [42, 62]. A possible way to generate solutions is the single row oriented method (SROM) proposed by Ren et al. [43]. In that work, it was demonstrated that SROM reduces the computation burden compared to other methods. Thus, SROM is used in this paper as the solution construction method. Additionally, we also consider the ant colony system (ACS) proposed by Dorigo and Gambardella [63] as an extension of the ACO algorithm. ACS includes a pseudo-random-proportional rule, providing a direct way to balance between exploration and exploitation during the selection of the solution component. If $z_t \in R_{t-1}$ denotes the column selected at step t, then the ACS rule is

$$z_t = \begin{cases} \arg\max\limits_{j \in R_{t-1}} \left\{ \left(\tau_j\right)^{\alpha} \left(\eta_j^t\right)^{\beta} \right\}, & \text{if } q \leq q_o \text{ (exploitation)}, \\ \zeta_t, & \text{otherwise (exploration)}, \end{cases} \tag{21}$$

where q is a random number uniformly distributed in $[0, 1]$, $q_0 \in [0, 1]$ is a parameter determining the relative importance of exploitation versus exploration, $\zeta_t \in R_{t-1}$ is the column provided by SROM at step t, and $\arg\max\{\cdot\}$ provides the point/points in which a function gets its maximum value/values. Thus, if $q \leq q_0$, then it returns the nonselected column having the highest value of $(\tau_j)^{\alpha}\ (\eta_j^t)^{\beta}$ at step t, where $\tau_j > 0$ denotes the pheromone trail of column j and $\alpha \geq 0$ and $\beta \geq 0$ denote the relative importance of pheromone trails and heuristic information, respectively. This step corresponds to line 4 of Algorithm 1.

(3) *Local search*: it is well known that local search is effective to improve ACO performance. We consider the local search proposed by Ren et al. [43], where for each column in Δ_t, the algorithm determines if the column should be removed or replaced by one or more columns while keeping solution feasibility. This step corresponds to line 5 of Algorithm 1.

(4) *Update pheromone trails*: we consider that pheromone trails are updated based on the max-min ant system (MMAS) approach proposed by Stützle and Hoos [64]. In this method, after each ant generates a full solution, all pheromone trails are decreased uniformly to simulate evaporation, forgetting part of the historical experience. Next, a small amount of pheromone is deposited on the columns corresponding to the best solution found. To this end, MMAS considers the best solution found in the current iteration, instead of the best solution found from the beginning of the algorithm. We opted for the second option as did Ren et al. [43]. Thus, the search can concentrate fast around the best solution found. This strategy could result in a bad performance if the algorithm is trapped in bad solution areas. However, this risk is reduced due to the ACS strategy detailed in Step 2. Formally, pheromone trails are updated following

$$\tau_j = \rho\,\tau_j + \omega_j, \quad \forall j \in J, \tag{22}$$

where $\rho \in [0, 1)$ is the pheromone persistence and $\omega_j \geq 0$ is the amount of pheromone put on column j provided in Stützle and Hoos [64]. Additionally, they also proposed that the range of pheromone trails is in $[1/[(1-\rho)(\sum_{j \in L} c_j)], \epsilon/\ [(1-\rho)(\sum_{j \in L} c_j)]]$, where $\epsilon \in (0, 1)$ denotes a ratio coefficient and L is the best solution found from the beginning of the algorithm, $L \subseteq J$. This step corresponds to line 7 of Algorithm 1.

5. Artificial Bee Colony

The ABC algorithm is inspired by honey bee behaviours, the search process being guided by three types of artificial bees: workers, onlookers, and scouts. The general procedure of ABC is shown in Algorithm 2. It starts by generating an initial population of ps_{abc} solutions. For every row, a random column with covering possibilities is selected until all rows are covered. Next, along iterations, the population is managed by $n_w - \text{ps}_{abc}$ workers and $1 - n_w$ onlookers, which are randomly recruited in each iteration. The behaviour of each bee is as follows:

(i) A worker takes a random solution from the population to generate a new solution by adding a random number of columns between 0 and abc_{add} (in percentage) of columns in the SCP instance. This step is followed by an elimination of random columns between 0 and abc_{eli} (in percentage) of columns in the SCP instance. The fitness value of the individual generated by the worker is obtained. In the case that the fitness value of the new individual is better than the previous individual assigned to the worker, then the new individual replaces the previous one. In the opposite case, the counter is increased for the number of trials for improving the current solution. Otherwise, the counter is set to zero. If such counter reaches the limit threshold, the worker is transformed into a scout bee.

```
Initialize
while not stop condition do
    for each ant in the nest do
        generate a new solution
        apply local search to the new solution
    end for
    update pheromone trails
end while
return the best solution found
```

Algorithm 1: The ACO algorithm.

```
Generate $ps_{abc}$ initial solutions
while not stop condition do
    Recruit $n_w - ps_{abc}$ worker bees
    Recruit $1 - n_w$ onlooker bees
    recruit scout bees if needed
    update the best solution found
end while
return the best solution found
```

Algorithm 2: ABC algorithm.

```
initialize $\xi_i = \|\Delta \cap \alpha_i\|, \forall i \in I$
initialize $V = \{i \mid \xi_i = 0, \forall i \in I\}$
for each row $i \in V$ in increasing order of $i$ do
    find the first column $j$ in $\alpha_i$ minimizing equation (23)-default or (24)-proposal
    add $j$ to $\Delta$
    set $\xi_i = \xi_i + 1, \forall i \in I_j$, and $V = V - I_j$
end for
for each column $j \in \Delta$ in decreasing order of $j$ do
    if $\xi_i \geq 2, \forall i \in I_j$then
        set $\Delta = \Delta - j$ and $\xi_i = \xi_i - 1, \forall i \in I_j$
    end if
    end for
return $\Delta$, which is a feasible SCP solution, without redundant columns
```

Algorithm 3: Heuristic operator for solution feasibility in ABC.

(ii) An onlooker generates a new solution following a similar procedure as for workers, but selecting the solution with probability to its quality, instead of randomly. The concept of the limit threshold is not used in onlookers.

(iii) A scout discards its current solution and generates a new one by following the same strategy as for generating the initial population. As expected, the counter of trials is initialized to zero.

As both workers and onlookers can generate unfeasible solutions because of the random elimination of columns, it is mandatory to manage this issue. Crawford et al. [13] proposed to consider the usual heuristic by Beasley and Chu [10] for transforming unfeasible solutions into feasible ones while reducing the cost of the solution in a later step. This heuristic is shown in Algorithm 3. Here, the first stage in lines 3 – 7 transforms an unfeasible solution into a feasible one. The second stage in lines 8 – 12 removes redundant columns. In this algorithm, note that Δ, α_i, and ξ_i are the set of columns in a solution, the set of columns that row $i \in I$ covers, and the number of columns in Δ that cover the row i, respectively.

Focusing on the first stage, the steps required to make a solution feasible include the identification of uncovered rows and the addition of columns to the solution so that all rows are covered. The search for the missing columns in the proposal of Beasley and Chu [10] is guided by

$$\min \frac{c_j}{\left\| I_j \cap V \right\|}, \quad (23)$$

where V is the number of noncovered rows in the solution, i.e., the ratio between the cost of a column and the number of noncovered rows, which could be covered by such column.

As an alternative strategy, this paper proposes to guide the search based on the concepts from the alternative SCP formulation, that is,

$$\min \frac{c_j}{\sum_{i \in \{I_j \cup V\}} g_i}, \quad (24)$$

i.e., the ratio between the cost of a column and the sum of the gains of covering the noncovered rows by such column.

6. Experimentation

This section discusses the experimental methodology and analyses the results obtained in the first and second studies.

6.1. Experimental Methodology. We apply the two approaches in each study for ACO and ABC algorithms to solve Beasley's OR library. This dataset is widely used to report empirical results in the current literature (see e.g. [9, 40, 48]). This library includes 65 non-unicost instances generated randomly, as detailed in Table 2. For further details about the random generation of these instances, see [18, 65]. For each instance in the library, the number of rows, the number of columns, and the cost of each column are provided. Additionaly, for each row, the number of columns that covers and also the list of columns which cover that row are also provided. For a complexity study of the search space in this benchmark, we refer readers to the work by Finger et al. [66], which considered the fitness-distance correlation landscape metric to this end. In Table 2, "*Density* (%)" contains the percentage of $1's$ in the A matrix in equation (2). "*Optimal solution*" shows two possible values, known and unknown, according to whether the instances have a solution tested to be optimal, or instead it could not be checked because of problem complexity. Thus, we only know the best historical solutions found for the sets nrg and nrh.

We combine two stop condition criteria for performing the experimentation: reaching a given number of fitness evaluations or getting the optimal solution. If at least a condition holds, the algorithm ends. For ACO, we assume 10,000 fitness evaluations as a stop condition. As we will discuss later, this value is enough for performing the experimentation. For ABC, we consider 500 iterations based on Crawford et al. [13].

Before running the experimentation, we should configure both algorithms. In the case of ABC, we can assume the parameters provided in Crawford et al. [13] for the two approaches of ABC considered here because (i) the authors also solved SCP and (ii) the approach based on the alternative formulation only modifies the heuristic operator for solution feasibility and the operators guiding the search are not modified. In the case of ACO, we should configure the two approaches of ACO considered here because (i) we do not have any set of parameters from previous works for the approach of ACO used and (ii) the approach based on the alternative formulation modifies how the search is performed in comparison to the traditional one, and then we should configure the two approaches independently.

Thus, for the first study, we consider $\mathrm{ps}_{abc} = 200$, $n_w = 100$, limit $= 50$, $abc_{\mathrm{add}} = 0.5\%$, and $abc_{\mathrm{eli}} = 1.2\%$ of Crawford et al. [13]. For the second study, we get the parameters of the two ACO approaches using F-Race. This method configures a metaheuristic starting with a set of candidate values for each parameter. Then, it discards bad performance configurations as soon as statistically sufficient evidence is reached against them, focusing on the most promising ones.

Concretely, we consider the iterated F-Race implementation for R software by López-Ibáñez et al. [67]. Following the authors' recommendations, we divided the benchmark into three groups according to the problem size ($m \times n$) to get a consistent configuration. Thus, Group A includes instance sets 4, 5, and 6; Group B includes instance sets a, b, c, and d; and, finally, Group C includes instance sets nre, nrf, nrg, and nrh. Table 3 shows the candidate values for each parameter based on previous works [43] and the configurations obtained for each group and ACO approach. Note that d-ACO denotes ACO with the traditional heuristic information expression and n-ACO denotes ACO with the alternative heuristic information expression.

Once both ACO approaches are configured, 30 independent runs are performed for each instance and algorithm. Next, we analyse if there are significant differences between the behaviour of the two algorithms regarding solution quality and execution time for each instance. To this end, the authors consider the Wilcoxon–Mann–Whitney test [68] to validate several hypotheses. The implementation of this test is the one provided in the assessment performance tool described in Knowles et al. [69] and available in Fonseca et al. [70].

However, indirectly the RPD metrics used for the assessment evaluation consider the optimal solution for the instances, which can be considered as the solutions provided by the corresponding exact techniques. Thus, the RPD metric evaluates how far the solution found by the metaheuristic is from the optimal solution provided by an exact technique.

As a solution quality metric, we consider the relative percentage deviation (RPD), which evaluates how far the solution found by the metaheuristic is from the optimal solution known in the literature. The lower the RPD value, the better the solution obtained. Thus, indirectly, this metric evaluates the performance of the technique in comparison with the corresponding generic exact technique solving the same problem instance. Three RPD metrics are included: the average RPD, $\overline{\mathrm{rpd}} \in [0, 1]$; the minimum RPD, $\mathrm{rpd}_{\min} \in [0, 1]$; and the maximum RPD, $\mathrm{rpd}_{\max} \in [0, 1]$. They are calculated as

Table 2: Beasley's OR library description.

Instance set	Number of instances	m	n	Cost range	Density (%)	Optimal solution
4	10	200	1,000	[1, 100]	2.00	Known
5	10	200	2,000	[1, 100]	2.00	Known
6	5	200	1,000	[1, 100]	5.00	Known
a	5	300	3,000	[1, 100]	2.00	Known
b	5	300	3,000	[1, 100]	5.00	Known
c	5	400	4,000	[1, 100]	2.00	Known
d	5	400	4,000	[1, 100]	5.00	Known
nre	5	500	5,000	[1, 100]	10.00	Known
nrf	5	500	5,000	[1, 100]	20.00	Known
nrg	5	1000	10,000	[1, 100]	2.00	Unknown
nrh	5	1000	10,000	[1, 100]	5.00	Unknown

Table 3: F-Race settings and configurations obtained for each ACO approach.

Parameter	Candidate values	d-ACO			n-ACO		
		A	B	C	A	B	C
α	0.25, 0.50, 1.00, 2.00, 5.00, 8.00	1.00	1.00	1.00	1.00	1.00	1.00
β	1.00, 2.00, 5.00, 8.00, 10.00, 13.00	8.00	8.00	5.00	8.00	8.00	8.00
ρ	0.80, 0.85, 0.90, 0.95, 0.98, 0.99, 0.995, 0.999	0.95	0.90	0.90	0.98	0.99	0.90
q_0	0.10, 0.25, 0.50, 0.75, 0.90, 0.95, 0.99	0.50	0.75	0.50	0.90	0.90	0.25
ϵ	0.0001, 0.0005, 0.001, 0.002, 0.005, 0.010	0.001	0.005	0.005	0.005	0.005	0.001
ps_{aco}	5, 10, 20, 50, 100, 150, 200, 250	20	10	100	150	150	50

$$\overline{\text{rpd}} = \left(\frac{\overline{z} - z_{\text{opt}}}{z_{\text{opt}}}\right), \tag{25}$$

$$\text{rpd}_{\min} = \left(\frac{z_{\min} - z_{\text{opt}}}{z_{\text{opt}}}\right), \tag{26}$$

$$\text{rpd}_{\max} = \left(\frac{z_{\max} - z_{\text{opt}}}{z_{\text{opt}}}\right), \tag{27}$$

where $\overline{z}$, $z_{\min}$, and $z_{\max}$ denote the average solution cost, the minimum solution cost, the maximum solution cost from a distribution of 30 samples solving a instance, respectively, and z_{opt} is the optimum solution cost of the instance. Note that the cost of the best solution found during one run is given by equation (4). Thus, although both SCP fitness formulations are different, we can compare the results obtained without loss of generality.

Regarding the computing platform considered to perform the experimentation, the authors used two computing nodes in a computing cluster. Each node has two 2.33 GHz Intel Xeon E5410 with four cores each and a 1600 MHz DDR3 16 GB RAM, running a Linux operating system. All executions were performed in a single core without parallelism because the goal of this paper is not to explore parallelism. The reason for considering such unconventional infrastructure is the possibility of performing many independent executions because of the needs for the statistical test required to validate the proposal. That means that for a single execution or a reduced set of them, a conventional computer could be considered. To avoid the operating system tasks affecting the total computing time obtained during the experimentation, one core in each computing node was idle. Additionally, the authors also checked that the RAM in the computing node was enough to not apply memory swap. As expected, the computing power capacity of the processor definitively affects the time required to find the solution to the problem, most of the operations being related to CPU computing and accessing the principal memory (RAM). Note the same computing nodes are considered for all the experiments in this work to not bias the conclusions reached regarding computing time.

Regarding programming languages, ACO algorithm was fully implemented in Java for Java Development Kit (JDK) 1.7. ABC algorithm was fully implemented in C. The scripts for managing the executions and collecting the results were implemented in bash. Note that the usage of two different programming languages for implementing ABC and ACO does not affect the conclusions reached in computing time. This fact is because ABC and ACO computing times are not compared in this work.

6.2. Analysis of the Experimental Results. Tables 4 and 5 show for each instance and case study, the RPD metrics ($\overline{\text{rpd}}$, $\text{rpd}_{\min}$, $\text{rpd}_{\max}$), average execution time reaching the stop condition (time (s)), and average fitness evaluations needed for reaching the best solution found during the exploration (evals). In both tables, lower $\overline{\text{rpd}}$ and time (s) values are given in bold for each instance. In Table 5, d-ABC denotes ABC with the default heuristic feasibility operator and n-ABC denotes ABC with the heuristic feasibility operator based on the alternative SCP formulation.

Analysing both tables regarding RPD metrics, we check that (i) n-ACO seems to outperform or match d-ACO in most instances and (ii) n-ABC seems to outperform or match d-ABC in most instances. Focusing on computing

Table 4: RPD metrics, the average total execution time, and average number of evaluations obtained during the first study for ACO algorithm.

Instance	z_{opt}	d-ACO					n-ACO				
		$\overline{\text{evals}}$	$\overline{\text{time}}$ (s)	$\overline{\text{rpd}}$ (%)	rpd_{min} (%)	rpd_{max} (%)	$\overline{\text{evals}}$	$\overline{\text{time}}$ (s)	$\overline{\text{rpd}}$ (%)	rpd_{min} (%)	rpd_{max} (%)
4_1	429	1880	845	0.23	0.23	0.23	1350	**109**	**0.00**	**0.00**	**0.00**
4_2	512	3950	900	0.20	**0.00**	0.59	2700	**224**	**0.00**	**0.00**	**0.00**
4_3	516	1600	916	0.97	**0.19**	1.16	4200	**704**	**0.19**	**0.00**	**0.78**
4_4	494	4850	880	0.40	**0.20**	0.81	2100	**829**	**0.20**	**0.20**	**0.20**
4_5	512	1950	680	0.10	**0.00**	**0.39**	600	**340**	**0.00**	**0.00**	**0.39**
4_6	560	3780	574	**0.00**	**0.00**	0.36	2625	**240**	**0.00**	**0.00**	**0.00**
4_7	430	1430	858	0.70	**0.00**	1.16	1425	**118**	**0.00**	**0.00**	**0.00**
4_8	492	2440	415	**0.00**	**0.00**	0.20	1350	**97**	**0.00**	**0.00**	**0.00**
4_9	641	5120	932	1.25	0.31	2.34	5550	**848**	**0.31**	**0.00**	**0.78**
4_10	514	1390	115	**0.00**	**0.00**	**0.00**	750	**60**	**0.00**	**0.00**	**0.00**
5_1	253	4860	891	0.40	**0.00**	1.58	3000	**526**	**0.00**	**0.00**	**1.19**
5_2	302	4970	945	1.49	0.66	2.32	2700	**870**	**1.32**	**0.33**	**1.99**
5_3	226	2080	753	0.22	**0.00**	1.33	1725	**125**	**0.00**	**0.00**	**0.00**
5_4	242	2780	245	**0.00**	**0.00**	**0.00**	1500	**110**	**0.00**	**0.00**	**0.00**
5_5	211	1160	857	0.47	0.47	**0.47**	750	**223**	**0.00**	**0.00**	**0.47**
5_6	213	640	59	**0.00**	**0.00**	**0.00**	300	**25**	**0.00**	**0.00**	**0.00**
5_7	293	5490	901	0.51	**0.34**	1.02	1575	**837**	**0.34**	**0.34**	**0.34**
5_8	288	3820	653	0.35	**0.00**	0.69	1650	**122**	**0.00**	**0.00**	**0.00**
5_9	279	20	823	**0.00**	**0.00**	0.36	975	**78**	**0.00**	**0.00**	**0.00**
5_10	265	2370	617	0.38	**0.00**	0.75	1350	**112**	**0.00**	**0.00**	**0.00**
6_1	138	3050	802	1.45	**0.00**	**1.45**	2175	**332**	**0.00**	**0.00**	**1.45**
6_2	146	1180	**122**	**0.00**	**0.00**	**0.00**	2025	164	**0.00**	**0.00**	**0.00**
6_3	145	1530	128	**0.00**	**0.00**	**0.00**	900	**70**	**0.00**	**0.00**	**0.00**
6_4	131	930	70	**0.00**	**0.00**	**0.00**	300	**22**	**0.00**	**0.00**	**0.00**
6_5	161	1130	831	1.24	**0.00**	2.48	2025	**378**	**0.00**	**0.00**	**1.86**
a_1	253	2725	**904**	0.79	0.79	0.79	4875	976	**0.40**	**0.40**	**0.40**
a_2	252	4700	**908**	1.19	1.19	1.19	3000	967	**0.40**	**0.00**	**0.79**
a_3	232	2925	**903**	**0.43**	**0.43**	**0.43**	5700	971	**0.43**	**0.43**	**0.43**
a_4	234	1575	895	0.43	0.43	**0.43**	3150	**444**	**0.00**	**0.00**	**0.43**
a_5	236	1815	902	0.42	0.42	**0.42**	3675	**478**	**0.00**	**0.00**	0.85
b_1	69	360	**45**	**0.00**	**0.00**	**0.00**	450	**45**	**0.00**	**0.00**	**0.00**
b_2	76	370	**37**	**0.00**	**0.00**	**0.00**	900	93	**0.00**	**0.00**	**0.00**
b_3	80	320	34	**0.00**	**0.00**	**0.00**	300	**33**	**0.00**	**0.00**	**0.00**
b_4	79	380	**38**	**0.00**	**0.00**	**0.00**	450	44	**0.00**	**0.00**	**0.00**
b_5	72	10	**1**	**0.00**	**0.00**	**0.00**	150	16	**0.00**	**0.00**	**0.00**
c_1	227	2690	**973**	1.32	**0.88**	1.76	6225	1034	**0.88**	**0.88**	**0.88**
c_2	219	2780	**979**	**0.91**	**0.91**	**0.91**	3975	1056	**0.91**	**0.91**	**0.91**
c_3	243	3305	**993**	1.65	0.82	2.47	5775	1067	**0.82**	**0.41**	**1.23**
c_4	219	2400	**235**	**0.00**	**0.00**	**0.00**	3900	380	**0.00**	**0.00**	**0.00**
c_5	215	1305	**979**	**0.47**	**0.00**	0.93	2925	1052	**0.47**	0.47	**0.47**
d_1	60	415	1174	1.67	**0.00**	1.67	2025	**473**	**0.00**	**0.00**	**0.00**
d_2	66	1675	**236**	**0.00**	**0.00**	**0.00**	1800	350	**0.00**	**0.00**	**0.00**
d_3	72	775	1198	**1.39**	1.39	**1.39**	975	**1019**	**1.39**	**0.00**	**1.39**
d_4	62	235	25	**0.00**	**0.00**	**0.00**	150	**20**	**0.00**	**0.00**	**0.00**
d_5	61	600	**63**	**0.00**	**0.00**	**0.00**	675	75	**0.00**	**0.00**	**0.00**
nre_1	29	100	29	**0.00**	**0.00**	**0.00**	50	**18**	**0.00**	**0.00**	**0.00**
nre_2	30	1650	902	**0.00**	**0.00**	3.33	1400	**423**	**0.00**	**0.00**	**0.00**
nre_3	27	900	216	**0.00**	**0.00**	**0.00**	700	**170**	**0.00**	**0.00**	**0.00**
nre_4	28	1200	452	**0.00**	**0.00**	3.57	1000	**260**	**0.00**	**0.00**	**0.00**
nre_5	28	200	49	**0.00**	**0.00**	**0.00**	100	**31**	**0.00**	**0.00**	**0.00**
nrf_1	14	200	118	**0.00**	**0.00**	**0.00**	200	**112**	**0.00**	**0.00**	**0.00**
nrf_2	15	100	64	**0.00**	**0.00**	**0.00**	75	**40**	**0.00**	**0.00**	**0.00**
nrf_3	14	100	5464	**7.14**	**0.00**	**7.14**	50	**4966**	**7.14**	**0.00**	**7.14**
nrf_4	14	500	271	**0.00**	**0.00**	**0.00**	200	**122**	**0.00**	**0.00**	**0.00**
nrf_5	13	100	**5511**	**7.69**	**7.69**	**7.69**	100	5614	**7.69**	**7.69**	**7.69**

Table 4: Continued.

Instance	z_{opt}	d-ACO					n-ACO				
		$\overline{\text{evals}}$	$\overline{\text{time}}$ (s)	$\overline{\text{rpd}}$ (%)	rpd_{min} (%)	rpd_{max} (%)	$\overline{\text{evals}}$	$\overline{\text{time}}$ (s)	$\overline{\text{rpd}}$ (%)	rpd_{min} (%)	rpd_{max} (%)
nrg_1	176	5750	1905	0.28	**0.00**	2.27	5350	**1250**	**0.00**	**0.00**	**0.00**
nrg_2	154	4800	**2166**	1.95	1.95	**1.95**	3425	2171	**1.30**	**0.65**	**1.95**
nrg_3	166	4350	**2253**	3.61	3.01	4.82	2775	2253	**3.01**	**2.41**	**4.22**
nrg_4	168	5250	**2207**	2.68	**1.19**	3.57	4100	2217	**2.38**	**1.19**	**2.98**
nrg_5	168	6450	2258	0.60	0.60	**0.60**	5500	**1830**	**0.00**	**0.00**	**0.60**
nrh_1	63	3650	**5499**	**3.17**	**1.59**	4.76	3525	5768	**3.17**	3.17	**3.17**
nrh_2	63	3300	**5437**	**3.17**	**3.17**	**3.17**	2100	5706	**3.17**	**3.17**	**3.17**
nrh_3	59	4200	**5511**	**3.39**	**3.39**	**3.39**	2750	5776	**3.39**	**3.39**	**3.39**
nrh_4	58	3600	**5434**	**3.45**	**1.72**	**3.45**	2200	5664	**3.45**	**1.72**	**3.45**
nrh_5	55	4250	2329	**0.00**	**0.00**	**0.00**	2975	**1708**	**0.00**	**0.00**	**0.00**

Table 5: RPD metrics, average total execution time, and average number of evaluations obtained during the second study for ABC algorithm.

Instance	z_{opt}	d-ABC					n-ABC				
		$\overline{\text{evals}}$	$\overline{\text{time}}$ (s)	$\overline{\text{rpd}}$ (%)	rpd_{min} (%)	rpd_{max} (%)	$\overline{\text{evals}}$	$\overline{\text{time}}$ (s)	$\overline{\text{rpd}}$ (%)	rpd_{min} (%)	rpd_{max} (%)
4_1	429	125,808	485	0.82	**0.23**	0.93	124,122	447	**0.23**	**0.23**	**0.23**
4_2	512	9998	93	**0.00**	**0.00**	**0.00**	8194	76	**0.00**	**0.00**	**0.00**
4_3	516	56027	390	**0.00**	**0.00**	0.19	11,191	103	**0.00**	**0.00**	**0.00**
4_4	494	128,399	535	**0.20**	**0.00**	0.61	126,209	488	**0.20**	0.20	**0.20**
4_5	512	11,161	102	**0.00**	**0.00**	**0.00**	118,961	354	0.39	**0.00**	**0.39**
4_6	560	127,988	537	0.36	**0.00**	0.54	28,428	254	**0.00**	**0.00**	**0.36**
4_7	430	110,922	207	0.93	**0.00**	1.16	110,224	194	**0.70**	0.47	**0.70**
4_8	492	126,117	491	0.20	0.20	**0.20**	34,292	283	**0.00**	**0.00**	**0.20**
4_9	641	128,322	526	0.94	**0.00**	2.18	128,276	523	**0.47**	**0.00**	**0.78**
4_10	514	18,162	164	**0.00**	**0.00**	**0.00**	55,670	333	**0.00**	**0.00**	0.58
5_1	253	125,020	1797	**0.40**	**0.00**	1.58	123,792	1706	**0.40**	**0.00**	**0.40**
5_2	302	127,193	1956	1.99	1.66	2.98	125,859	1856	**0.99**	**0.00**	**1.32**
5_3	226	123,812	1713	1.33	1.33	1.33	123,627	1696	**0.88**	**0.88**	**0.88**
5_4	242	17,751	650	**0.00**	**0.00**	**0.00**	6760	239	**0.00**	**0.00**	**0.00**
5_5	211	14,945	521	**0.00**	**0.00**	**0.00**	7927	277	**0.00**	**0.00**	**0.00**
5_6	213	19,299	684	**0.00**	**0.00**	**0.00**	5575	200	**0.00**	**0.00**	**0.00**
5_7	293	126,401	1924	0.34	0.34	1.02	14,753	547	**0.00**	**0.00**	**0.00**
5_8	288	17,153	652	**0.00**	**0.00**	**0.00**	23,534	814	**0.00**	**0.00**	**0.00**
5_9	279	124,770	1754	0.36	0.36	0.36	8961	316	**0.00**	**0.00**	**0.00**
5_10	265	86,539	1149	0.19	**0.00**	1.13	12,431	407	**0.00**	**0.00**	**0.00**
6_1	138	124,417	1198	1.45	**0.00**	2.17	91,269	916	**0.36**	**0.00**	**1.45**
6_2	146	8784	217	**0.00**	**0.00**	**0.00**	6381	156	**0.00**	**0.00**	**0.00**
6_3	145	14,699	359	**0.00**	**0.00**	**0.00**	4776	115	**0.00**	**0.00**	**0.00**
6_4	131	16,928	400	**0.00**	**0.00**	**0.00**	3961	95	**0.00**	**0.00**	**0.00**
6_5	161	13,760	338	**0.00**	**0.00**	**0.00**	4377	106	**0.00**	**0.00**	**0.00**
a_1	253	123,573	5729	**0.40**	**0.40**	0.79	122,222	5381	**0.40**	**0.40**	**0.40**
a_2	252	126,016	6309	1.39	0.79	2.38	126,212	6346	**0.79**	**0.00**	**1.19**
a_3	232	126,103	6334	1.29	0.86	1.72	125,826	6189	**0.43**	**0.00**	**0.86**
a_4	234	125,908	6287	**0.43**	**0.43**	**0.43**	124,897	6004	**0.43**	**0.43**	**0.43**
a_5	236	124,785	5948	0.42	0.42	0.42	11,144	1346	**0.00**	**0.00**	**0.00**
b_1	69	120,832	13,012	1.45	**0.00**	1.45	3970	1228	**0.00**	**0.00**	**0.00**
b_2	76	6962	2054	**0.00**	**0.00**	**0.00**	3372	1043	**0.00**	**0.00**	**0.00**
b_3	80	4170	1296	**0.00**	**0.00**	**0.00**	3572	1106	**0.00**	**0.00**	**0.00**
b_4	79	6166	1932	**0.00**	**0.00**	**0.00**	3965	1227	**0.00**	**0.00**	**0.00**
b_5	72	3172	988	**0.00**	**0.00**	**0.00**	2774	857	**0.00**	**0.00**	**0.00**
c_1	227	121,978	12,589	**0.88**	**0.44**	1.76	116,551	9431	**0.88**	0.88	**0.88**
c_2	219	110,309	13,899	0.23	**0.00**	1.37	21,664	6492	**0.00**	**0.00**	**0.00**
c_3	243	126,728	15,246	1.65	0.41	2.47	126,018	14,881	**0.82**	**0.00**	**2.06**
c_4	219	126,196	14,951	0.46	**0.00**	1.83	28,355	7658	**0.00**	**0.00**	**0.00**
c_5	215	19,075	5434	**0.00**	**0.00**	**0.00**	14,546	4086	**0.00**	**0.00**	**0.00**

Table 5: Continued.

Instance	z_{opt}	d-ABC					n-ABC				
		$\overline{\text{evals}}$	$\overline{\text{time}}$ (s)	$\overline{\text{rpd}}$ (%)	$\text{rpd}_{\min}$ (%)	$\text{rpd}_{\max}$ (%)	$\overline{\text{evals}}$	$\overline{\text{time}}$ (s)	$\overline{\text{rpd}}$ (%)	$\text{rpd}_{\min}$ (%)	$\text{rpd}_{\max}$ (%)
d_1	60	5970	4455	**0.00**	**0.00**	**0.00**	7572	5603	**0.00**	**0.00**	**0.00**
d_2	66	121,272	31,555	1.52	1.52	1.52	11,343	8371	**0.00**	**0.00**	**0.00**
d_3	72	122,829	34,063	1.39	1.39	2.78	18,808	13,510	**0.00**	**0.00**	**0.00**
d_4	62	121,976	32,383	1.61	1.61	1.61	4768	3510	**0.00**	**0.00**	**0.00**
d_5	61	109,457	14,077	1.64	1.64	1.64	5090	3690	**0.00**	**0.00**	**0.00**
nre_1	29	27,684	68,066	**0.00**	**0.00**	**0.00**	3479	9634	**0.00**	**0.00**	**0.00**
nre_2	30	119,319	111,213	6.67	6.67	6.67	18,666	51,982	**0.00**	**0.00**	**3.33**
nre_3	27	120,070	115,028	7.41	7.41	7.41	19,339	53,470	**0.00**	**0.00**	**0.00**
nre_4	28	119,063	109,812	10.71	3.57	14.29	10,559	30,055	**0.00**	**0.00**	**0.00**
nre_5	28	118,585	106,781	3.57	3.57	7.14	4053	11,203	**0.00**	**0.00**	**0.00**
nrf_1	14	119,200	230,417	14.29	14.29	14.29	2353	13,741	**0.00**	**0.00**	**0.00**
nrf_2	15	119,831	238,705	13.33	13.33	13.33	1950	11,337	**0.00**	**0.00**	**0.00**
nrf_3	14	119,332	236,611	14.29	14.29	21.43	3929	23,123	**0.00**	**0.00**	**0.00**
nrf_4	14	119,910	241,666	14.29	14.29	14.29	2746	16,074	**0.00**	**0.00**	**0.00**
nrf_5	13	106,580	79,141	15.38	15.38	23.08	120,731	248,033	**7.69**	**7.69**	**7.69**
nrg_1	176	128,776	257,664	2.84	2.27	3.41	126,471	235,636	**1.14**	**0.57**	**1.70**
nrg_2	154	128,678	253,564	3.25	2.60	3.90	126,160	233,168	**1.95**	**1.30**	**2.60**
nrg_3	166	134,516	243,567	3.61	3.01	4.22	124,477	217,781	**2.41**	**1.81**	**3.01**
nrg_4	168	135,789	223,456	3.57	2.98	4.17	125,268	224,757	**1.79**	**1.19**	**2.38**
nrg_5	168	126,782	227,346	4.17	3.57	4.76	126,077	232,047	**2.38**	**2.38**	**2.98**
nrh_1	63	126,421	228,326	3.17	3.17	4.76	145,678	253,467	**1.59**	**1.59**	**3.17**
nrh_2	63	134,282	217,456	3.17	4.76	4.76	155,671	243,457	**1.59**	**1.59**	**3.17**
nrh_3	59	123,481	237,222	3.39	5.08	5.08	132,451	233,245	**1.69**	**1.69**	**3.39**
nrh_4	58	143,562	242,456	3.45	3.45	5.17	142,457	231,002	**1.72**	**1.72**	**3.45**
nrh_5	55	134,612	232,216	3.64	3.64	3.64	143,417	224,452	**1.82**	**1.82**	**3.64**

times, we reach a similar behaviour, where n-ACO appears to need a shorter time than d-ACO, except for b, c, and nrh instances, and n-ABC appears to need a shorter $\overline{\text{time}}$ than d-ABC in general. Focusing on evaluations, the $\overline{\text{evals}}$ field is related to the number of iterations reached as follows. In ACO, ps_{aco} evaluations are performed for each iteration. In ABC, a number of evaluations varying between ps_{abc} and two times ps_{abc} are performed for each iteration. Thus, for ABC, the maximum number of evaluations will be a value in the range $[100,000, 200,000]$. For ACO, the maximum number of evaluations will be 10,000 as defined before. Analysing the $\overline{\text{evals}}$ field, the authors reach that the number of evaluations needed is distant from the stop condition defined, and then the stop condition is adequate in both studies.

Table 6 shows the average RPD metrics for each instance group, where "ipv" field denotes the percentage of improvement by considering the alternative approach of the algorithm instead of the default version. Analysing this table, it is observed that (i) n-ACO provides better RPD values than d-ACO for all the groups and (ii) n-ABC also provides better RPD values than its default version. The RPD metrics obtained are in line with other works from the literature, with RPD values lower than 1.0%. In this regard, Table 7 shows the values of some recent successful approaches solving the problem. However, we should remark that the purpose of this work is not to outperform other techniques solving the standard SCP benchmark.

At this point, it seems that the alternative approach of the algorithms provides better performance in both cases. However, we do not know if the differences observed are significant. To this end, the statistical methodology procedure described by Lanza-Gutierrez et al. [56] was applied. First, we removed all possible outliers. Then, we analysed the normality of data, obtaining that we cannot assume normal distribution in any case. Consequently, the median should be considered as average value for calculating $\overline{z}$ in equation (25).

Next, we study if there are significant differences in the solution quality of the algorithms. Starting with the first study, we consider the Wilcoxon–Mann–Whitney test with hypotheses H_0: $\overline{\text{rpd}}_a \geq \overline{\text{rpd}}_b$ and H_1: $\overline{\text{rpd}}_a < \overline{\text{rpd}}_b$, $\forall a, b \in \{\text{d} - \text{ACO}, \text{n} - \text{ACO}\}$ with $a \neq b$, where $\overline{\text{rpd}}_a$ and $\overline{\text{rpd}}_b$ are the average RPD of the algorithm a and b for a given instance, respectively. The p values obtained for each instance and ACO approach are shown in Table 8 under the title *RPD analysis*, where p values lower than the significance level $\alpha = 0.05$ are given in bold, i.e., the confidence level is 0.95. Note that the unilateral test performed between the two possibilities was the one that matches with the descriptive analysis, the other test being marked with a dash in the table. Also note that in case of equality between the average RPD values, the two unilateral tests are performed. For the second study, we consider the Wilcoxon–Mann–Whitney test with similar hypotheses as before H_0: $\overline{\text{rpd}}_c \geq \overline{\text{rpd}}_d$ and H_1: $\overline{\text{rpd}}_c < \overline{\text{rpd}}_d$, $\forall c, d \in \{\text{d} - \text{ABC}, \text{n} - \text{ABC}\}$ with $c \neq d$. The p values obtained are also shown in Table 9 with the same notation as in Table 8.

Table 6: RPD metrics for each instance group and algorithm in both studies.

Instance group	$\overline{\text{rpd}}$			rpd_{min}			rpd_{max}		
	d-ACO (%)	n-ACO (%)	ipv	d-ACO (%)	n-ACO (%)	ipv	d-ACO (%)	n-ACO (%)	ipv
First study									
A	0.43	**0.08**	81.80	0.08	**0.03**	63.83	0.79	**0.43**	46.03
B	0.53	**0.28**	46.60	0.36	**0.17**	51.89	0.62	**0.39**	37.24
C	1.86	**1.74**	6.54	1.22	**1.17**	3.74	2.49	**1.89**	24.06
ALL	0.94	**0.70**	25.69	0.55	**0.46**	17.20	1.30	**0.90**	30.60
				Second study					
Instance group	$\overline{\text{rpd}}$			rpd_{min}			rpd_{max}		
	d-ABC (%)	n-ABC (%)	ipv	d-ABC (%)	n-ABC (%)	ipv	d-ABC (%)	n-ABC (%)	ipv (%)
A	0.38	**0.19**	51.25	0.16	**0.07**	56.65	0.66	**0.30**	54.25
B	0.74	**0.19**	74.56	0.50	**0.09**	82.81	1.11	**0.29**	73.77
C	10.71	**1.07**	90.01	9.91	**0.90**	90.95	12.75	**1.48**	88.39
ALL	3.94	**0.48**	87.80	3.52	**0.35**	90.04	4.84	**0.69**	85.73

Table 7: Review of recent approaches solving Beasley's OR library.

Author/s	SIA	$\overline{\text{rpd}}$ (%)	rpd_{min} (%)	rpd_{max} (%)
Farahani et al. [41]	ABC	0.16	0.01	—
Ren et al. [43]	ACO	0.20	0.03	0.39
Lu and Vasko [48]	TLBO20	—	0.06	—
Lu and Vasko [48]	TLBO10	—	0.09	—
Naji-Azimi et al. [49]	EM-like	—	0.20	—
Lu and Vasko [48]	TLBO	—	0.28	—

Table 8: p values obtained analysing the differences observed regarding execution time and solution quality during the first study for the ACO algorithm.

	RPD analysis (H_1)		Execution time analysis (H_1)	
	d-ACO < n-ACO	n-ACO < d-ACO	d-ACO < n-ACO	n-ACO < d-ACO
Inst				
4_1	—	**≤0.001**	—	**≤ 0.001**
4_2	—	**≤0.001**	—	**≤0.001**
4_3	—	**≤0.001**	—	**≤0.001**
4_4	—	**≤0.001**	—	**≤0.001**
4_5	—	**0.003**	—	**0.006**
4_6	—	**≤0.001**	—	**≤0.001**
4_7	—	**≤0.001**	—	**≤0.001**
4_8	—	**≤0.001**	—	**≤0.001**
4_9	—	**≤0.001**	—	**≤0.001**
4_10	0.500	0.500	—	**≤0.001**
5_1	—	**0.004**	—	**≤0.001**
5_2	—	0.131	—	**≤0.001**
5_3	—	**≤0.001**	—	**≤0.001**
5_4	0.500	0.500	—	**≤0.001**
5_5	—	**≤0.001**	—	**≤0.001**
5_6	0.500	0.500	—	**≤0.001**
5_7	—	**≤0.001**	—	**≤0.001**
5_8	—	**≤0.001**	—	**≤0.001**
5_9	—	0.060	—	**≤0.001**
5_10	—	**≤0.001**	—	**≤0.001**
6_1	—	**0.002**	—	0.150
6_2	0.500	0.500	0.117	—
6_3	0.500	0.500	—	**≤0.001**
6_4	0.500	0.500	—	**≤0.001**
6_5	—	**0.005**	—	0.054
a_1	—	**≤0.001**	**≤ 0.001**	—

Table 8: Continued.

	RPD analysis (H_1)		Execution time analysis (H_1)	
	d-ACO < n-ACO	n-ACO < d-ACO	d-ACO < n-ACO	n-ACO < d-ACO
a_2	—	≤ **0.001**	**0.006**	—
a_3	0.500	0.500	≤ **0.001**	—
a_4	—	≤ **0.001**	—	**0.001**
a_5	—	≤ **0.001**	—	≤ **0.001**
b_1	0.500	0.500	0.291	—
b_2	0.500	0.500	≤ **0.001**	—
b_3	0.500	0.500	—	0.565
b_4	0.500	0.500	0.106	—
b_5	0.500	0.500	≤ **0.001**	—
c_1	—	≤ **0.001**	≤ **0.001**	—
c_2	0.500	0.500	≤ **0.001**	—
c_3	—	≤ **0.001**	≤ **0.001**	—
c_4	0.500	0.500	**0.013**	—
c_5	—	**0.002**	≤ **0.001**	—
d_1	—	≤ **0.001**	—	**0.002**
d_2	0.500	0.500	0.124	—
d_3	—	≤ **0.001**	—	**0.003**
d_4	0.500	0.500	—	**0.017**
d_5	0.500	0.500	0.059	—
nre_1	0.500	0.500	—	≤ **0.001**
nre_2	—	≤ **0.001**	—	≤ **0.001**
nre_3	0.500	0.500	—	0.112
nre_4	—	**0.002**	—	**0.006**
nre_5	0.500	0.500	—	≤ **0.001**
nrf_1	0.500	0.500	—	0.108
nrf_2	0.500	0.500	—	≤ **0.001**
nrf_3	0.500	0.500	—	0.292
nrf_4	0.500	0.500	—	≤ **0.001**
nrf_5	0.500	0.500	≤ **0.001**	—
nrg_1	—	≤ **0.001**	—	≤ **0.001**
nrg_2	—	≤ **0.001**	0.261	—
nrg_3	—	≤ **0.001**	0.378	—
nrg_4	—	**0.038**	0.574	—
nrg_5	—	≤ **0.001**	—	≤ **0.001**
nrh_1	—	0.001	≤ **0.001**	—
nrh_2	0.500	0.500	≤ **0.001**	—
nrh_3	0.500	0.500	≤ **0.001**	—
nrh_4	—	0.261	≤ **0.001**	—
nrh_5	0.500	0.500	—	**0.003**

Table 9: p values obtained analysing the differences observed regarding execution time and solution quality during the second study for the ABC algorithm.

Inst	RPD analysis (H_1)		Execution time analysis (H_1)	
	d-ABC < n-ABC	n-ABC < d-ABC	d-ABC < n-ABC	n-ABC < d-ABC
4_1	—	**≤0.001**	—	**≤0.001**
4_2	0.500	0.500	—	**≤0.001**
4_3	—	**≤0.001**	—	**≤0.001**
4_4	**0.037**	—	—	0.054
4_5	**≤0.001**	—	**≤0.001**	—
4_6	—	**≤0.001**	—	**≤0.001**
4_7	—	**≤0.001**	—	**≤0.001**
4_8	—	**≤0.001**	—	**≤0.001**
4_9	—	**≤0.001**	—	0.459
4_10	**≤0.001**	—	**≤0.001**	—

Table 9: Continued.

Inst	RPD analysis (H_1)		Execution time analysis (H_1)	
	d-ABC < n-ABC	n-ABC < d-ABC	d-ABC < n-ABC	n-ABC < d-ABC
5_1	—	**≤0.001**	—	**0.001**
5_2	—	**≤0.001**	—	**≤0.001**
5_3	—	**≤0.001**	0.415	—
5_4	0.500	0.500	—	**≤0.001**
5_5	0.500	0.500	—	**≤0.001**
5_6	0.500	0.500	—	**≤0.001**
5_7	—	**≤0.001**	—	**≤0.001**
5_8	0.500	0.500	0.348	—
5_9	—	**≤0.001**	—	**≤0.001**
5_10	—	**≤0.001**	—	**≤0.001**
6_1	—	**0.001**	—	**≤0.001**
6_2	0.500	0.500	—	**≤0.001**
6_3	0.500	0.500	—	**≤0.001**
6_4	0.500	0.500	—	**≤0.001**
6_5	0.500	0.500	—	**≤0.001**
a_1	—	**0.001**	—	**≤0.001**
a_2	—	**≤0.001**	0.479	—
a_3	—	**≤0.001**	—	0.135
a_4	0.500	0.500	—	**0.025**
a_5	—	**≤0.001**	—	**≤0.001**
b_1	—	**≤0.001**	—	**≤0.001**
b_2	0.500	0.500	—	**≤0.001**
b_3	0.500	0.500	—	**0.012**
b_4	0.500	0.500	—	**≤0.001**
b_5	0.500	0.500	—	**≤0.001**
c_1	—	**0.001**	—	**≤0.001**
c_2	—	**≤0.001**	—	**≤0.001**
c_3	—	**≤0.001**	—	**0.022**
c_4	—	**≤0.001**	—	**≤0.001**
c_5	0.500	0.500	—	**0.001**
d_1	0.500	0.500	0.097	—
d_2	—	**≤0.001**	—	**≤0.001**
d_3	—	**≤0.001**	—	**≤0.001**
d_4	—	**≤0.001**	—	**≤0.001**
d_5	—	**≤0.001**	—	**≤0.001**
nre_1	0.500	0.500	—	**≤0.001**
nre_2	—	**≤0.001**	—	**0.003**
nre_3	—	**≤0.001**	—	**≤0.001**
nre_4	—	**≤0.001**	—	**≤0.001**
nre_5	—	**≤0.001**	x**0.001**	—
nrf_1	—	**≤0.001**	—	**≤0.001**
nrf_2	—	**≤0.001**	—	**≤0.001**
nrf_3	—	**≤0.001**	—	**≤0.001**
nrf_4	—	**≤0.001**	—	**≤0.001**
nrf_5	—	**≤0.001**	**≤0.001**	—
nrg_1	—	**≤0.001**	0.500	0.500
nrg_2	—	**≤0.001**	0.500	0.500
nrg_3	—	**≤0.001**	0.500	0.500
nrg_4	—	**≤0.001**	0.500	0.500
nrg_5	—	**≤0.001**	0.500	0.500
nrh_1	—	**≤0.001**	0.500	0.500
nrh_2	—	**≤0.001**	0.500	0.500
nrh_3	—	**≤0.001**	0.500	0.500
nrh_4	—	**≤0.001**	0.500	0.500
nrh_5	—	**≤0.001**	0.500	0.500

Table 10: Percentage of cases where an algorithm provides the best significant performance compared to the other regarding solution quality and execution time.

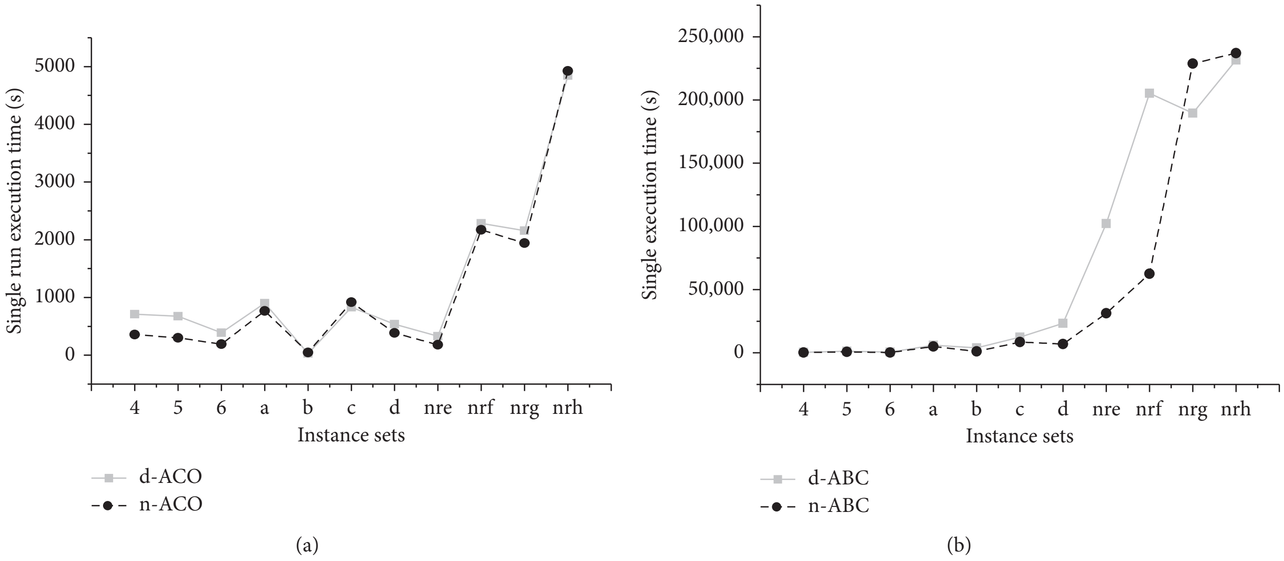
	RPD analysis				Execution time analysis				
	A (%)	B (%)	C (%)	ALL (%)		A (%)	B (%)	C (%)	ALL (%)
First study									
n-ACO > d-ACO	26.15	13.85	12.31	**52.31**	n-ACO > d-ACO	33.85	7.69	13.85	**55.38**
d-ACO > n-ACO	0.00	0.00	0.00	0.00	d-ACO > n-ACO	0.00	15.38	7.69	23.08
n-ACO = d-ACO	12.31	16.92	18.46	47.69	n-ACO = d-ACO	4.62	7.69	9.23	21.54
Percentage	38.46	30.77	30.77	100.00	Percentage	38.46	30.77	30.77	100.00
Second study									
n-ABC > d-ABC	23.64	23.64	16.36	**63.64**	n-ABC > d-ABC	29.23	26.15	13.85	**69.23**
d-ABC > n-ABC	5.45	0.00	1.82	7.27	d-ABC > n-ABC	3.08	0.00	1.54	4.62
n-ABC = d-ABC	16.36	12.73	0.00	29.09	n-ABC = d-ABC	6.15	4.62	15.38	26.15
Percentage	45.45	36.36	18.18	100.00	Percentage	38.46	30.77	30.77	100.00

Through the Wilcoxon–Mann–Whitney test, the differences observed regarding execution time for both algorithms are also analysed. For the first study, we define the hypotheses H_0: $\overline{\text{time (s)}}_a \geq \overline{\text{time (s)}}_b$ and H_1: $\overline{\text{time (s)}}_a < \overline{\text{time (s)}}_b$, $\forall a, b \in \{\text{d} - \text{ACO}, \text{n} - \text{ACO}\}$ with $a \neq b$, where $\overline{\text{time (s)}}_a$ and $\overline{\text{time (s)}}_b$ are the average execution time of the algorithm a and b for a given instance, respectively. The p values obtained for each instance and algorithm are shown in Table 8 under the title *Execution time analysis*, where p values are given in bold with the same criterion as before. For the second study, we consider the Wilcoxon–Mann–Whitney test with similar hypotheses as before H_0: $\overline{\text{time (s)}}_c \geq \overline{\text{time (s)}}_d$ and H_1: $\overline{\text{time (s)}}_c < \overline{\text{time (s)}}_d$, $\forall c, d \in \{\text{d} - \text{ABC}, \text{n} - \text{ABC}\}$ with $c \neq d$. The p values obtained are also shown in Table 9.

Based on the previous statistical analysis, Table 10 shows the percentage of cases where an algorithm provides the best significant performance compared to another for each study in terms of RPD and execution time. Focusing on the first study and RPD values, we verify that n-ACO provides better behaviour than d-ACO in 52.31% of cases. However, it is important to remark that d-ACO never provides better results than n-ACO, meaning that n-ACO clearly outperforms d-ACO. For execution time, n-ACO needs lower execution times than d-ACO in 55.38% of cases and d-ACO needs lower execution times than n-ACO in 23.08% of cases. This fact could mean that the alternative heuristic information needs higher execution times under certain conditions. However, most cases in which d-ACO needs lower execution times correspond with instances whose optimal solution is not reached, and then the 10,000 evaluations are performed for each algorithm, e.g., for instances a_1, a_2, a_3, c_1, c_2, c_3, c_5, nrh_1, nrh_2, nrh_3, and nrh_4. In such unfavourable cases, the differences observed are not of concern as shown in Figure 2(a). On the other hand, most cases in which n-ACO needs lower execution times correspond with instances whose optimal solution is reached. In such cases, a greater difference is observed favoring n-ACO. This fact is because n-ACO reaches the stop condition before d-ACO, e.g., for instance sets 4, 5, 6, d, nre, and nrg. Focusing on the second study, we verify that n-ABC provides better behaviour than d-ABC in 63.64% of cases, where d-ABC outperforms n-ABC in 7.27% of cases. This unfavourable situation occurs in small instances, where the search space is reduced. For execution time, n-ABC needs lower execution times than d-ABC in 69.23% of cases and d-ABC needs lower execution times than n-ABC in 4.62% of cases. As before, this unfavourable situation mainly occurs when n-ABC does not reach the optimal solution, penalizing the additional computation of the gain concept in the alternative SCP formulation. This behaviour is shown in Figure 2(b).

The previous analysis is completed with the landscape study in Tables 11 and 12 for the solutions obtained solving the instances through ACO and ABC, respectively. The metrics in such tables quantify solution quality (QMetric $\in [0, 1]$), the rate of success (SRate $\in [0, 1]$), and speed of reaching a solution (SSpeed $\in [0, 1]$). QMetric follows an exponential formulation which allows distinguishing between the performance of two algorithms, which obtained solutions close to the optimum fitness. SRate is defined as the number of successful runs that the algorithm reaches the optimum fitness divided by the total number of runs. SSpeed quantifies the number of evaluations taken to reach the optimum fitness. For the three metrics, the value 1 indicates the highest quality. More details about the three metrics, as well as formulation, are listed in [71]. Analysing Tables 11 and 12, we check that, in general, both n-ACO and n-ABC provide a higher or equal QMetric than d-ACO and d-ABC approaches. For SSpeed, we check that both d-ACO and d-ABC need a lower or equal number of evaluations than n-ACO and d-ABC to reach the optimum fitness. For SRate, we check that both d-ACO and d-ABC reach the optimum fitness a greater or equal number of times than n-ACO and d-ABC. That means that n-ACO and n-ABC obtained better quality solutions, are better in convergence, and provide a more robust performance than the default approaches.

Up to this point, we know that the concepts included in the alternative formulation positively affect the search process in the first study, where the concepts from the alternative formulation are considered for guiding purposes. A mapping of the solutions visited by n-ACO and d-ACO could help to effectively show how the two approaches explore the search space. To this end, we consider the mapping method (MaM) proposed by Autuori et al. [72], where a mapping function converts a multidimensional space solution in one dimensional space through two steps: (i) a binary conversion is applied to the solution (for SCP,



(a)

(b)

Figure 2: Analysis for the average execution times in each study. (a) Average execution time for each ACO approach and instance set. (b) Average execution time for each ABC approach and instance set.

Table 11: Landscape metrics for each instance and ACO approach.

Inst	d-ACO			n-ACO		
	QMetric	SSpeed	SRate	QMetric	SSpeed	SRate
4_1	0.0008	0.5107	0.1000	**1.0000**	**0.8366**	**0.9667**
4_2	0.1777	0.4248	0.1667	**1.0000**	**0.7190**	**1.0000**
4_3	0.0019	0.0000	0.0000	**0.4467**	**0.5119**	**0.4333**
4_4	0.0027	0.0000	0.0000	**0.0114**	**0.6300**	**0.1000**
4_5	0.5029	0.6151	0.5000	**0.5695**	**0.7979**	**0.5667**
4_6	0.6473	0.5445	0.6333	**1.0000**	**0.6931**	**0.8667**
4_7	0.0669	0.8310	0.0667	**1.0000**	**0.8382**	**0.9333**
4_8	0.6751	0.6144	0.6667	**1.0000**	**0.8425**	**1.0000**
4_9	0.0005	0.0000	0.0000	**0.1045**	**0.4400**	**0.1000**
4_10	**1.0000**	0.8497	**1.0000**	**1.0000**	**0.9080**	**1.0000**
5_1	0.3336	0.4622	0.3333	**0.6335**	**0.5903**	**0.6333**
5_2	0.0000	**0.0000**	**0.0000**	**0.0007**	**0.0000**	**0.0000**
5_3	0.4334	0.5965	0.4333	**1.0000**	**0.8098**	**0.9333**
5_4	**1.0000**	0.6987	**1.0000**	**1.0000**	**0.8650**	**1.0000**
5_5	0.0000	**0.8640**	0.0333	**0.6333**	**0.8066**	**0.6333**
5_6	**1.0000**	0.9183	**1.0000**	**1.0000**	**0.9615**	**1.0000**
5_7	0.0009	0.0000	0.0000	**0.0017**	**0.7800**	**0.2000**
5_8	0.5340	0.5448	0.5333	**1.0000**	**0.8215**	**1.0000**
5_9	0.4017	0.5528	0.4000	**1.0000**	**0.8775**	**1.0000**
5_10	0.4669	0.6416	0.4667	**1.0000**	**0.8216**	**0.9667**
6_1	0.2359	0.4803	0.2333	**0.5696**	**0.7115**	**0.5667**
6_2	**1.0000**	**0.8268**	0.8667	**1.0000**	0.7477	**0.9333**
6_3	**1.0000**	0.7902	0.9667	**1.0000**	**0.9130**	**1.0000**
6_4	**1.0000**	0.8791	**1.0000**	**1.0000**	**0.9645**	**1.0000**
6_5	0.2355	**0.7840**	0.2333	**0.5350**	0.7516	**0.5333**
a_1	0.0037	**0.0000**	**0.0000**	**0.0517**	**0.0000**	**0.0000**
a_2	0.0001	0.0000	0.0000	**0.2202**	**0.7075**	**0.2000**
a_3	**0.0337**	**0.0000**	**0.0000**	**0.0337**	**0.0000**	**0.0000**
a_4	0.0306	0.0610	0.0333	**0.6769**	**0.5440**	**0.6667**
a_5	0.0350	0.5955	0.0667	**0.6772**	**0.6153**	**0.6667**
b_1	**1.0000**	0.9180	**1.0000**	**1.0000**	**0.9510**	**1.0000**
b_2	**1.0000**	**0.9584**	**1.0000**	**1.0000**	0.8810	**1.0000**
b_3	**1.0000**	0.9531	**1.0000**	**1.0000**	**0.9595**	**1.0000**
b_4	**1.0000**	**0.9487**	**1.0000**	**1.0000**	0.9425	**1.0000**
b_5	**1.0000**	**0.9990**	**1.0000**	**1.0000**	0.9850	**1.0000**

Table 11: Continued.

Inst	d-ACO			n-ACO		
	QMetric	SSpeed	SRate	QMetric	SSpeed	SRate
c_1	0.0021	0.0000	0.0000	**0.0086**	**0.9700**	**0.0333**
c_2	**0.0120**	**0.0000**	**0.0000**	**0.0120**	**0.0000**	**0.0000**
c_3	0.0055	**0.0000**	**0.0000**	**0.0785**	**0.0000**	**0.0000**
c_4	**1.0000**	**0.6953**	0.9667	**1.0000**	0.6230	**1.0000**
c_5	**0.0925**	0.6810	**0.0333**	0.0918	**0.8950**	**0.0333**
d_1	0.5069	0.5334	0.3000	**1.0000**	**0.6635**	**0.7667**
d_2	**1.0000**	0.6861	**0.9333**	**1.0000**	**0.7275**	0.8000
d_3	0.3280	**0.8020**	0.1000	**0.5968**	0.6775	**0.4000**
d_4	**1.0000**	0.9768	**1.0000**	**1.0000**	**0.9850**	**1.0000**
d_5	**1.0000**	**0.9288**	**1.0000**	**1.0000**	0.9115	**1.0000**
nre_1	**1.0000**	0.9900	**1.0000**	**1.0000**	**0.9950**	**1.0000**
nre_2	0.9319	0.7268	0.6333	**1.0000**	**0.7906**	**0.8333**
nre_3	**1.0000**	0.8629	0.9333	**1.0000**	**0.8947**	**1.0000**
nre_4	0.9499	**0.8336**	0.7333	**1.0000**	0.8086	**0.9667**
nre_5	**1.0000**	0.9800	**1.0000**	**1.0000**	**0.9877**	**1.0000**
nrf_1	**1.0000**	0.9717	**1.0000**	**1.0000**	**0.9782**	**1.0000**
nrf_2	**1.0000**	0.9847	**1.0000**	**1.0000**	**0.9898**	**1.0000**
nrf_3	**0.9621**	**0.8770**	**0.3333**	**0.9621**	0.7620	**0.3333**
nrf_4	**1.0000**	0.9407	0.9667	**1.0000**	**0.9683**	**1.0000**
nrf_5	**0.9441**	0.6980	**0.1667**	**0.9441**	**0.8650**	0.0667
nrg_1	0.7528	0.3613	0.5000	**1.0000**	**0.4760**	**0.8667**
nrg_2	0.2463	**0.0000**	**0.0000**	**0.3555**	**0.0000**	**0.0000**
nrg_3	0.0808	**0.0000**	**0.0000**	**0.1123**	**0.0000**	**0.0000**
nrg_4	0.1762	**0.0000**	**0.0000**	**0.2026**	**0.0000**	**0.0000**
nrg_5	0.6374	0.2350	0.2000	**0.8550**	**0.3203**	**0.6000**
nrh_1	0.7880	**0.0000**	**0.0000**	**0.8135**	**0.0000**	**0.0000**
nrh_2	0.8136	**0.0000**	**0.0000**	**0.8398**	**0.0000**	**0.0000**
nrh_3	**0.8148**	0.0000	0.0000	**0.8148**	**0.2700**	**0.0333**
nrh_4	0.8392	0.0000	0.0000	**0.8484**	**0.5900**	**0.0333**
nrh_5	**1.0000**	0.5407	0.9667	**1.0000**	**0.6670**	**1.0000**

Table 12: Landscape metrics for each instance and ABC approach.

Inst	d-ABC			n-ABC		
	QMetric	SSpeed	SRate	QMetric	SSpeed	SRate
4_1	0.0001	0.0000	0.0000	**0.0008**	**0.7873**	**0.1333**
4_2	**1.0000**	0.9486	**1.0000**	**1.0000**	**0.9598**	**1.0000**
4_3	0.5791	0.8366	0.5667	**1.0000**	**0.9454**	**1.0000**
4_4	**0.3068**	0.8238	**0.3000**	0.0114	**0.8441**	0.1333
4_5	**1.0000**	**0.9482**	**1.0000**	0.4337	0.8642	0.4333
4_6	0.0846	**0.8985**	0.0667	**0.7432**	0.8656	**0.7333**
4_7	**0.0333**	**0.9187**	**0.0333**	0.0000	0.0000	0.0000
4_8	0.0254	0.0000	0.0000	**0.7076**	**0.8662**	**0.7000**
4_9	0.0335	0.8215	0.0333	**0.0671**	**0.8849**	**0.0667**
4_10	**1.0000**	**0.9050**	**1.0000**	0.5385	0.8514	0.5333
5_1	0.0349	0.8071	0.0333	**0.3338**	**0.8912**	**0.3333**
5_2	0.0000	0.0000	0.0000	**0.1013**	**0.8183**	**0.1000**
5_3	0.0000	0.0000	0.0000	**0.0000**	**0.9069**	**0.1000**
5_4	**1.0000**	0.8848	0.9000	**1.0000**	**0.9674**	**1.0000**
5_5	**1.0000**	0.9134	**1.0000**	**1.0000**	**0.9575**	**1.0000**
5_6	**1.0000**	0.9108	**1.0000**	**1.0000**	**0.9727**	**1.0000**
5_7	0.0011	0.0000	0.0000	**1.0000**	**0.9068**	**0.9667**
5_8	**1.0000**	**0.8952**	**0.9333**	**1.0000**	0.8698	0.8667
5_9	0.0029	0.0000	0.0000	**1.0000**	**0.9517**	**1.0000**
5_10	0.5002	0.8888	0.5000	**1.0000**	**0.9296**	**0.8333**

Table 12: Continued.

Inst	d-ABC			n-ABC		
	QMetric	SSpeed	SRate	QMetric	SSpeed	SRate
6_1	0.2335	**0.9274**	0.2333	**0.5008**	0.8686	**0.5000**
6_2	**1.0000**	0.9509	**1.0000**	**1.0000**	**0.9681**	**1.0000**
6_3	**1.0000**	0.9239	0.9000	**1.0000**	**0.9764**	**1.0000**
6_4	**1.0000**	0.9193	**1.0000**	**1.0000**	**0.9807**	**1.0000**
6_5	**1.0000**	0.9206	**1.0000**	**1.0000**	**0.9778**	**1.0000**
a_1	0.0373	**0.0000**	**0.0000**	**0.0517**	**0.0000**	**0.0000**
a_2	0.0006	0.0000	0.0000	**0.2109**	**0.8496**	**0.2000**
a_3	0.0006	0.0000	0.0000	**0.1476**	**0.8056**	**0.1333**
a_4	**0.0306**	0.8544	**0.2333**	**0.0306**	**0.8577**	0.1667
a_5	0.0350	0.0000	0.0000	**1.0000**	**0.9298**	**1.0000**
b_1	0.5415	0.8758	0.4667	**1.0000**	**0.9792**	**1.0000**
b_2	**1.0000**	0.9646	**1.0000**	**1.0000**	**0.9829**	**1.0000**
b_3	**1.0000**	0.9791	**1.0000**	**1.0000**	**0.9824**	**1.0000**
b_4	**1.0000**	0.9682	**1.0000**	**1.0000**	**0.9796**	**1.0000**
b_5	**1.0000**	0.9843	**1.0000**	**1.0000**	**0.9871**	**1.0000**
c_1	0.0075	0.0000	0.0000	**0.0086**	**0.7975**	**0.0333**
c_2	0.5250	0.7752	0.5000	**1.0000**	**0.8852**	**0.9000**
c_3	0.0094	0.0000	0.0000	**0.1069**	**0.5973**	**0.0667**
c_4	0.1301	0.7586	0.0667	**1.0000**	**0.8601**	**0.7667**
c_5	**1.0000**	0.9049	**1.0000**	**1.0000**	**0.9239**	**1.0000**
d_1	**1.0000**	**0.9653**	**1.0000**	**1.0000**	0.9578	**1.0000**
d_2	0.3285	0.0000	0.0000	**1.0000**	**0.9389**	**1.0000**
d_3	0.2665	0.0000	0.0000	**1.0000**	**0.9094**	**0.8333**
d_4	0.2820	0.5737	0.0667	**1.0000**	**0.9716**	**1.0000**
d_5	0.2963	0.9176	0.0333	**1.0000**	**0.9736**	**1.0000**
nre_1	**1.0000**	0.8874	0.8000	**1.0000**	**0.9818**	**1.0000**
nre_2	0.6684	0.0000	0.0000	**0.9381**	**0.8912**	**0.6667**
nre_3	0.6712	0.0000	0.0000	**1.0000**	**0.8703**	**0.8000**
nre_4	0.5555	0.0000	0.0000	**1.0000**	**0.9228**	**0.9667**
nre_5	0.0000	0.0000	0.0000	**1.0000**	**0.9796**	**1.0000**
nrf_1	0.8880	0.0000	0.0000	**1.0000**	**0.9875**	**1.0000**
nrf_2	0.7632	0.0000	0.0000	**1.0000**	**0.9900**	**1.0000**
nrf_3	0.6578	0.0000	0.0000	**1.0000**	**0.9799**	**1.0000**
nrf_4	0.7354	0.0000	0.0000	**1.0000**	**0.9852**	**1.0000**
nrf_5	0.7624	0.0000	0.0000	**0.9441**	**0.9702**	**0.0333**
nrg_1	0.2134	**0.0000**	**0.0000**	**0.4013**	**0.0000**	**0.0000**
nrg_2	0.2100	**0.0000**	**0.0000**	**0.2295**	**0.0000**	**0.0000**
nrg_3	0.1345	**0.0000**	**0.0000**	**0.2122**	**0.0000**	**0.0000**
nrg_4	0.1234	**0.0000**	**0.0000**	**0.2529**	**0.0000**	**0.0000**
nrg_5	0.1245	**0.0000**	**0.0000**	**0.2343**	**0.0000**	**0.0000**
nrh_1	0.1876	**0.0000**	**0.0000**	**0.2342**	**0.0000**	**0.0000**
nrh_2	0.1984	**0.0000**	**0.0000**	**0.2311**	**0.0000**	**0.0000**
nrh_3	0.1976	**0.0000**	**0.0000**	**0.2542**	**0.0000**	**0.0000**
nrh_4	0.1764	**0.0000**	**0.0000**	**0.2103**	**0.0000**	**0.0000**
nrh_5	0.1648	**0.0000**	**0.0000**	**0.2231**	**0.0000**	**0.0000**

the binary encoding is straightforward, where a column takes the value 1 if it is considered in the solution and 0 otherwise) and (ii) the binary Hamming distance is calculated between such binary solution and a reference solution randomly generated, resulting in a new representation of the solution. This representation is used for identifying the different zones explored by the algorithms. Note that the number of zones corresponds with the number of elements in the binary encoding (the number of columns). To this end, all the binary representations are added, resulting in a frequency diagram, showing how usually the algorithm includes a column in a solution.

Table 13 shows three metrics analysing the frequency diagrams previously generated for each ACO approach and an instance from each group 4, 5, 6, a, b, c, d, nre, nrf, nrg, and nrh. Note that the frequency diagrams were generated using all the solutions built in the 30 runs for each algorithm. The metrics used were also proposed by Autuori et al. [72] and are (i) the number of unexplored zones ("uz"), the number of explored zones ("ez"), and the number of large explored zones ("lez"). The metrics are related as follows. Convergence is considered high if lez is few in number. Diversity is considered good if the number of ez is large. Analysing Table 13, we check that ez is usually higher for n-ACO than for d-ACO, meaning that

Table 13: Mapping analysis of the two ACO approaches, when solving a instance from each group.

Inst	n-ACO			d-ACO		
	uz	ez	lez	uz	ez	lez
4_1	51	122	**41**	50	**123**	43
5_1	52	171	**58**	49	**174**	64
6_1	58	174	**46**	51	**181**	50
a_1	109	**297**	**59**	113	293	71
b_1	329	**235**	76	351	213	**60**
c_1	129	**417**	99	150	396	**97**
d_1	569	**299**	81	586	282	**60**
nre_1	2286	164	**91**	2197	**253**	106
nrf_1	4658	145	**81**	4543	**260**	98
nrg_1	1433	915	**172**	728	**1620**	200
nrh_1	4408	602	**140**	3826	**1184**	163

d-ACO improves diversity during the search compared to the traditional approach. This fact is especially relevant for large instances, as occurs for nrg_1 and nrh_1, where ez metric is significantly large. Focusing on lez, we check that the differences observed between both approaches are not as pronounced as for ez. However, such differences could mean that d-ACO has a lack of convergence during the search, and then future authors should manage this fact.

From these two studies solving the traditional SCP, we verify that (i) the concepts from the alternative formulation are useful in guiding the search process of a metaheuristic and (ii) the concepts from the alternative formulation are useful in updating a usual heuristic feasibility operator from the literature. The improvement in both studies was observed in terms of the solution quality, the rate of success, and the speed of reaching a solution. The conclusion in (ii) is especially interesting because this type of operators is generically applied in metaheuristics (generating unfeasible solutions), and the proposal could be directly incorporated into many solving methods.

7. Conclusions and Future Scope

Traditionally, SCP is formulated without addressing two issues: solution unsatisfiability and set redundancy, meaning that the solving method has to implement mechanisms to control such aspects. In recent years, an alternative SCP formulation was proposed, whose main contribution was that both issues were directly addressed by including penalties in the fitness function.

Reviewing the current scientific literature, we check that the alternative SCP formulation has received limited attention. Hence, we question whether there is any advantage of using this formulation beyond addressing set redundancy and feasibility aspects. This idea led us to propose two studies based on a metaheuristic approach. The aim is to identify if there is any concept in the alternative formulation, which could be considered for enhancing a solving method using the traditional formulation. The first study considers an ACO algorithm in two contexts: (i) solving the problem by addressing the traditional SCP formulation and (ii) solving SCP addressing the traditional formulation but using concepts from the alternative one for guiding the search. The second study considers an ABC algorithm in two contexts: (i) solving SCP addressing the traditional formulation and (ii) solving SCP addressing the traditional formulation but including concepts from the alternative one for updating a usual heuristic feasibility operator from the literature.

As a result of the first study, the authors conclude that it is possible to consider the gain concept from the alternative SCP formulation to successfully guide ACO search addressing the traditional SCP formulation. The benefits of the novel guide are shown in terms of solution quality, convergence, execution time, and diversity. From the second study, the authors conclude that it is possible to consider the gain concept from the alternative SCP formulation to update a feasibility operator from the literature. The benefits of the novel feasibility operator are shown in terms of solution quality, execution time, and convergence.

The first conclusion is interesting for designing novel guide strategies for solving the traditional SCP formulation. The second conclusion is especially interesting because feasibility operators are widely considered in metaheuristic approaches solving the SCP because of the usual generation of unfeasible solutions. This type of operators is integrated into the solving method as black-box methods. That means that it is straightforward to interchange one method for another. This situation implies that the feasibility operator based on the alternative SCP formulation presented here could be integrated with a reduced effort in already published works from the literature, as well as in future works to evaluate each specific use case.

As future lines of research, it would be interesting to consider additional metaheuristics to this study, as well as a larger dataset with bigger problems. Additionally, it could be interesting to extend this work by taking into account the performance of the solving methods and the search space complexity of the instances based on the landscape metrics.

Data Availability

The results shown in this paper were obtained by solving some freely available datasets in the literature. They can be found in http://people.brunel.ac.uk/~mastjjb/jeb/orlib/scpinfo.html.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors would like to thank the following grants: Juan A. Gomez-Pulido is supported by grant IB16002 (Junta Extremadura, Spain), Broderick Crawford is supported by grant CONICYT/FONDECYT/REGULAR/1171243, and Ricardo Soto is supported by grant CONICYT/FONDECYT/REGULAR/1190129. The authors also thank "Proyecto CORFO 14ENI2-26905 Nueva Ingeniería para el 2030."

References

[1] R. M. Karp, *Reducibility among Combinatorial Problems*, Springer, Berlin, Germany, 1972.

[2] L. Liang, D. Cool, N. Kakani, G. Wang, H. Ding, and A. Fenster, "Automatic radiofrequency ablation planning for liver tumors with multiple constraints based on set covering," *IEEE Transactions on Medical Imaging*, vol. 39, no. 5, pp. 1459–1471, 2020.

[3] I.-J. Jeong, "An optimal approach for a set covering version of the refueling-station location problem and its application to a diffusion model," *International Journal of Sustainable Transportation*, vol. 11, no. 2, pp. 86–97, 2017.

[4] H. Ye and H. Kim, "Locating healthcare facilities using a network-based covering location problem," *GeoJournal*, vol. 81, no. 6, pp. 875–890, 2016.

[5] A. O. Adewumi and O. J. Adeleke, "A survey of recent advances in vehicle routing problems," *International Journal of System Assurance Engineering and Management*, vol. 9, no. 1, pp. 155–172, 2018.

[6] S. S. V. Vianna, "The set covering problem applied to optimisation of gas detectors in chemical process plants," *Computers & Chemical Engineering*, vol. 121, pp. 388–395, 2019.

[7] V. Chvatal, "A greedy heuristic for the set-covering problem," *Mathematics of Operations Research*, vol. 4, no. 3, pp. 233–235, 1979.

[8] N. Bilal, P. Galinier, and F. Guibault, "A new formulation of the set covering problem for metaheuristic approaches," *ISRN Operations Research*, vol. 2013, pp. 1–10, 2013.

[9] F. J. Vasko, Y. Lu, and K. Zyma, "What is the best greedy-like heuristic for the weighted set covering problem?" *Operations Research Letters*, vol. 44, no. 3, pp. 366–369, 2016.

[10] J. E. Beasley and P. C. Chu, "A genetic algorithm for the set covering problem," *European Journal of Operational Research*, vol. 94, no. 2, pp. 392–404, 1996.

[11] J. E. Beasley, *Or-library*, http://people.brunel.ac.uk/mastjjb/jeb/orlib/scpinfo.html, 2016.

[12] M. Birattari, Z. Yuan, P. Balaprakash, and T. Stützle, "F-race and iterated f-race: an overview," *Experimental Methods for the Analysis of Optimization Algorithms*, Springer, Berlin, Germany, 2010.

[13] B. Crawford, R. Soto, R. Cuesta, and F. Paredes, "Application of the artificial bee colony algorithm for solving the set covering problem," *The Scientific World Journal*, vol. 2014, Article ID 189164, 8 pages, 2014.

[14] M. L. Fisher and P. Kedia, "Optimal solution of set covering/partitioning problems using dual heuristics," *Management Science*, vol. 36, no. 6, pp. 674–688, 1990.

[15] J. E. Beasley and K. Jørnsten, "Enhancing an algorithm for set covering problems," *European Journal of Operational Research*, vol. 58, no. 2, pp. 293–300, 1992.

[16] E. Balas and M. C. Carrera, "A dynamic subgradient-based branch-and-bound procedure for set covering," *Operations Research*, vol. 44, no. 6, pp. 875–890, 1996.

[17] R. Bar-Yejuda and S. Even, "A linear-time approximation algorithm for the weighted vertex cover problem," *Journal of Algorithms*, vol. 2, no. 2, pp. 198–203, 1981.

[18] J. E. Beasley, "An algorithm for set covering problem," *European Journal of Operational Research*, vol. 31, no. 1, pp. 85–93, 1987.

[19] E. El-Darzi and G. Mitra, "Set covering and set partitioning: a collection of test problems," *Omega*, vol. 18, no. 2, pp. 195–201, 1990.

[20] A. Caprara, P. Toth, and M. Fischetti, "Algorithms for the set covering problem," *Annals of Operations Research*, vol. 98, no. 1, pp. 353–371, 2000.

[21] F. J. Vasko, "An efficient heuristic for large set covering problems," *Naval Research Logistics Quarterly*, vol. 31, no. 1, pp. 163–171, 1984.

[22] T. A. Feo and M. G. C. Resende, "A probabilistic heuristic for a computationally difficult set covering problem," *Operations Research Letters*, vol. 8, no. 2, pp. 67–71, 1989.

[23] M. Haouari and J. S. Chaouachi, "A probabilistic greedy search algorithm for combinatorial optimisation with application to the set covering problem," *Journal of the Operational Research Society*, vol. 53, no. 7, pp. 792–799, 2002.

[24] J. H. Ablanedo-Rosas and C. Rego, "Surrogate constraint normalization for the set covering problem," *European Journal of Operational Research*, vol. 205, no. 3, pp. 540–551, 2010.

[25] J. E. Beasley, "A Lagrangian heuristic for set-covering problems," *Naval Research Logistics*, vol. 37, no. 1, pp. 151–164, 1990.

[26] S. Ceria, P. Nobili, and A. Sassano, "A Lagrangian-based heuristic for large-scale set covering problems," *Mathematical Programming*, vol. 81, no. 2, pp. 215–228, 1998.

[27] A. Caprara, M. Fischetti, and P. Toth, "A heuristic method for the set covering problem," *Operations Research*, vol. 47, no. 5, pp. 730–743, 1999.

[28] P. Du and Y. Zhang, "A new distributed approximation algorithm for the maximum weight independent set problem," *Mathematical Problems in Engineering*, vol. 2016, Article ID 9790629, 10 pages, 2016.

[29] R. Soto, E. Rodriguez-Tello, and E. Monfroy, "Recent advances on swarm intelligence for solving complex engineering problems," *Mathematical Problems in Engineering*, vol. 2018, Article ID 5642786, 1 pages, 2018.

[30] X. Zhao, R. Li, and X. Zuo, "Advances on qos-aware web service selection and composition with nature-inspired computing," *CAAI Transactions on Intelligence Technology*, vol. 4, no. 3, pp. 159–174, 2019.

[31] X. Xue, J. Lu, and J. Chen, "Using nsga-iii for optimising biomedical ontology alignment," *CAAI Transactions on Intelligence Technology*, vol. 4, no. 3, pp. 135–141, 2019.

[32] L. R. Rodrigues and J. P. P. Gomes, "Tlbo with variable weights applied to shop scheduling problems," *CAAI Transactions on Intelligence Technology*, vol. 4, no. 3, pp. 148–158, 2019.

[33] H. S. Pannu, D. Singh, and A. K. Malhi, "Improved particle swarm optimization based adaptive neuro-fuzzy inference system for benzene detection," *CLEAN-Soil, Air, Water*, vol. 46, no. 5, Article ID 1700162, 2018.

[34] M. Kaur, V. Kumar, and L. Li, "Color image encryption approach based on memetic differential evolution," *Neural Computing and Applications*, vol. 31, no. 11, pp. 7975–7987, 2019.

[35] P. C. Chu and J. E. Beasley, "A genetic algorithm for the generalised assignment problem," *Computers & Operations Research*, vol. 24, no. 1, pp. 17–23, 1997.

[36] U. Aickelin, "An indirect genetic algorithm for set covering problems," *Computers & Operations Research*, vol. 31, no. 5, pp. 1118–1126, 2004.

[37] M. Deveci and N. Ç. Demirel, "Evolutionary algorithms for solving the airline crew pairing problem," *Computers & Industrial Engineering*, vol. 115, pp. 389–406, 2018.

[38] T. Lust and D. Tuyttens, "Variable and large neighborhood search to solve the multiobjective set covering problem," *Journal of Heuristics*, vol. 20, no. 2, pp. 165–188, 2014.

[39] F. Colombo, R. Cordone, and G. Lulli, "A variable neighborhood search algorithm for the multimode set covering problem," *Journal of Global Optimization*, vol. 63, no. 3, pp. 461–480, 2015.

[40] S. Sundar and A. Singh, "A hybrid heuristic for the set covering problem," *Operational Research*, vol. 12, no. 3, pp. 345–365, 2012.

[41] R. Z. Farahani, A. Hassani, S. M. Mousavi, and M. B. Baygi, "A hybrid artificial bee colony for disruption in a hierarchical maximal covering location problem," *Computers & Industrial Engineering*, vol. 75, pp. 129–141, 2014.

[42] L. Lessing, I. Dumitrescu, and T. Stützle, "A comparison between aco algorithms for the set covering problem," in *Ant Colony Optimization and Swarm Intelligence*, pp. 1–12, Springer, Berlin, Germany, 2004.

[43] Z.-G. Ren, Z.-R. Feng, L.-J. Ke, and Z.-J. Zhang, "New ideas for applying ant colony optimization to the set covering problem," *Computers & Industrial Engineering*, vol. 58, no. 4, pp. 774–784, 2010.

[44] B. Crawford, R. Soto, E. Monfroy, F. Paredes, and W. Palma, "A hybrid ant algorithm for the set covering problem," *International Journal of Physical Science*, vol. 6, no. 19, pp. 4667–4673, 2011.

[45] B. Crawford, R. Soto, M. Olivares-Suarez et al., "A binary coded firefly algorithm that solves the set covering problem," *Romanian Journal of Information Science and Technology*, vol. 17, no. 3, pp. 252–264, 2014.

[46] B. Crawford, R. Soto, M. Riquelme-Leiva et al., "Modified binary firefly algorithms with different transfer functions for solving set covering problems," in *Software Engineering in Intelligent Systems*, pp. 307–315, Springer, Berlin, Germany, 2015.

[47] B. Crawford, R. Soto, F. Aballay, S. Misra, F. Johnson, and F. Paredes, "A teaching-learning-based optimization algorithm for solving set covering problems," in *Computational Science and its Applications-ICCSA 2015*, Springer, Berlin, Germany, 2015.

[48] Y. Lu and F. J. Vasko, "An OR practitioner's solution approach for the set covering problem," *International Journal of Applied Metaheuristic Computing*, vol. 6, no. 4, pp. 1–13, 2015.

[49] Z. Naji-Azimi, P. Toth, and L. Galli, "An electromagnetism metaheuristic for the unicost set covering problem," *European Journal of Operational Research*, vol. 205, no. 2, pp. 290–300, 2010.

[50] R. Soto, B. Crawford, A. Muñoz, F. Johnson, and F. Paredes, "Pre-processing, repairing and transfer functions can help binary electromagnetism-like algorithms," in *Artificial Intelligence Perspectives and Applications*, Springer, Berlin, Germany, 2015a.

[51] B. Crawford, R. Soto, C. Peña et al., "Binarization methods for shuffled frog leaping algorithms that solve set covering problems," in *Software Engineering in Intelligent Systems*, pp. 317–326, Springer, Berlin, Germany, 2015.

[52] B. Crawford, R. Soto, C. Torres-Rojas et al., "A binary fruit fly optimization algorithm to solve the set covering problem," in *Proceedings of the International Conference on Computational Science and its Applications-ICCSA 2015*, pp. 411–420, Saint Petersburg, Russia, July 2015.

[53] R. Soto, B. Crawford, J. Barraza, F. Johnson, and F. Paredes, "Solving pre-processed set covering problems via cuckoo search and lévy flights," in *Proceedings of the 10th Iberian Conference on Information Systems and Technologies (CISTI)*, pp. 1–6, Agueda, Aveiro, Portugal, June 2015.

[54] R. Soto, B. Crawford, R. Olivares et al., "Solving the non-unicost set covering problem by using cuckoo search and black hole optimization," *Natural Computing*, vol. 16, no. 2, pp. 213–229, 2017.

[55] B. Crawford, R. Soto, N. Berríos et al., "A binary cat swarm optimization algorithm for the non-unicost set covering problem," *Mathematical Problems in Engineering*, vol. 2015, Article ID 578541, 8 pages, 2015.

[56] J. M. Lanza-Gutierrez, B. Crawford, R. Soto, N. Berrios, J. A. Gomez-Pulido, and F. Paredes, "Analyzing the effects of binarization techniques when solving the set covering problem through swarm optimization," *Expert Systems with Applications*, vol. 70, pp. 67–82, 2017.

[57] S. Balaji and N. Revathi, "A new approach for solving set covering problem using jumping particle swarm optimization method," *Natural Computing*, vol. 15, no. 3, pp. 503–517, 2016.

[58] R. Soto, B. Crawford, R. Olivares et al., "Adaptive black hole algorithm for solving the set covering problem," *Mathematical Problems in Engineering*, vol. 2018, Article ID 2183214, 23 pages, 2018.

[59] B. Crawford, R. Soto, R. Olivares et al., "A binary monkey search algorithm variation for solving the set covering problem," *Natural Computing*, pp. 1–17, 2019.

[60] N. Bilal, P. Galinier, and F. Guibault, "An iterated-tabu-search heuristic for a variant of the partial set covering problem," *Journal of Heuristics*, vol. 20, no. 2, pp. 143–164, 2014.

[61] B. Crawford, R. Soto, W. Palma, F. Paredes, F. Johnson, and E. Norero, "The impact of a new formulation when solving the set covering problem using the aco metaheuristic," in *Modelling, Computation and Optimization in Information Systems and Management Sciences*, pp. 209–218, Springer, Berlin, Germany, 2015.

[62] B. Crawford and C. Castro, "Integrating lookahead and post processing procedures with aco for solving set partitioning and covering problems," in *Artificial Intelligence and Soft Computing–ICAISC, 2006*, pp. 1082–1090, Springer, Berlin, Germany, 2006.

[63] M. Dorigo and L. M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53–66, 1997.

[64] T. Stützle and H. H. Hoos, "Ant system," *Future Generation Computer Systems*, vol. 16, no. 8, pp. 889–914, 2000.

[65] E. Balas, "Cutting planes from conditional bounds: a new approach to set covering," in *Combinatorial Optimization*, pp. 19–36, Springer, Berlin, Germany, 1980.

[66] M. Finger, T. Stützle, and H. Lourenço, "Exploiting fitness distance correlation of set covering problems," in *Workshops on Applications of Evolutionary Computation*, pp. 61–71, Springer, Berlin, Germany, 2002.

[67] M. López-Ibáñez, J. Dubois-Lacoste, L. Pérez Cáceres, M. Birattari, and T. Stützle, "The irace package: iterated racing for automatic algorithm configuration," *Operations Research Perspectives*, vol. 3, pp. 43–58, 2016.

[68] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *The Annals of Mathematical Statistics*, vol. 18, no. 1, pp. 50–60, 1947.

[69] J. Knowles, L. Thiele, and E. Zitzler, "A tutorial on the performance assessment of stochastic multiobjective optimizers," *Computer Engineering and Networks Laboratory (TIK)*, ETH Zurich, Switzerland, 2006.

[70] C. Fonseca, J. Knowles, L. Thiele, and E. Zitzler, "Performance assessment package," 2006, https://sop.tik.ee.ethz.ch/pisa/?page=assessment.php.

[71] K. M. Malan and A. P. Engelbrecht, "Fitness landscape analysis for metaheuristic performance prediction," in *Recent Advances in the Theory and Application of Fitness Landscapes*, pp. 103–132, Springer, Berlin, Germany, 2014.

[72] J. Autuori, F. Hnaien, and F. Yalaoui, "A mapping technique for better solution exploration: NSGA-II adaptation," *Journal of Heuristics*, vol. 22, no. 1, pp. 89–123, 2016.

Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 5716985, 15 pages
https://doi.org/10.1155/2020/5716985

Research Article

The Modified Particle Swarm Optimization for a Special Case of the Assignment Problem: A Case Study in Chicken Transportation

Naratip Supattananon and Raknoi Akararungruangkul

Department of Industrial Engineering, Faculty of Engineering, KhonKaen University, KhonKaen 40000, Thailand

Correspondence should be addressed to Raknoi Akararungruangkul; raxaka@kku.ac.th

Received 28 February 2020; Accepted 20 May 2020; Published 8 July 2020

Guest Editor: Dilbag Singh

This research aims at solving the special case of multistate assignment problem. The problem includes many special characteristics which are not normally included in the assignment problem. There are many types and conditions of vehicles included in the planning and different road conditions of traveling, which would have different effects on fuel consumption, which is the objective function of the study. The proposed problem is determined as a large and complicated problem making the optimization software unable to find an optimal solution within the proper time. Therefore, the researchers had developed a method for determining the optimal solution by using particle swarm optimization (PSO) in which the methods were developed for solving the proposed problem. This method is called the modified particle swarm optimization (modified PSO). The proposed method was tested with three groups of tested instances, i.e., small, medium, and large groups. The computational result shows that, in small-sized and medium-sized problems, the proposed method performed not significantly different from the optimization software, and in the large-sized problems, the modified PSO method gave 3.61% lower cost than the cost generated from best solution generated from optimization software within 72 hours and it gave 11.03% better solution than that of the best existing heuristics published so far (differential evolution algorithm).

1. Introduction

The egg is very important to the Thai economy and its people. Since it is a high nutrition fact food and can be cooked in many dishes, both main courses and desserts, it is therefore widely consumed, resulting in high demand for hen eggs throughout the year and it is likely to increase steadily. During 2012–2018, the average domestic egg consumption tended to increase at a rate of 6.13% per year. In 2018, the consumption of eggs was equal to 14,823.24 million, which is increased from 13,534.98 million eggs at a rate of 9.52% from the year 2017. Egg is cheap compared to other protein resources and easy for cooking. In addition, the government and private sectors have campaigned on the egg consumption suitable for all ages. In 2012–2018, Thai egg production increased at a rate of 5.98% per year, according to the increase in consumption demand. In 2018, the eggs were produced in the amount of 14,915.82 million, which is increased from 13,724.42 million eggs at a rate of 8.68% from the year 2017. Since there was the efficient management of egg farms, resulting in increased productivity, the export value of fresh eggs in 2018 was 92.58 million eggs or worth 319.42 million baht, which tended to increase at the rate of 6.82% per year. In 2018, 4,356.92 tons of chicken eggs were exported which is equivalent to 425.41 million baht. Japan was the main export market.

This research studied the problem of chicken transportation, which was the multistage assignment problem. For the case study, an appropriate vehicle was assigned to transport the chickens directly from its farms to the egg farms, for the purpose of finding the answer with the lowest assignment cost. The mathematical models for the multistage assignment problem were developed which are suitable for the case study. Then, the estimating methods to find the optimal answer were also developed by employing the particle swarm optimization (PSO). When transporting chickens to the purchased farms, some factors must be considered, such as transportation standards, time, and temperature, and also the chickens apart from multiple sources must not blend up. These factors may affect the quality of the chicken. Therefore, if the assigned vehicle is appropriate and meets the needs of the chicken farms, it shall be used to send the chickens directly to the egg farms, without transporting chickens from other farms. The resulting cost of assignments or production costs is at the lowest value, which benefits the chicken farm concerning the decrease in production costs. Since each farm is responsible for all the costs incurred for transportation, they must be managed correctly and efficiently by establishing production schedules and assignments on farms, in both the allocation of chickens and the size of trucks that are used to deliver efficiently and quickly. It shall create the highest possible operating profit and the overall limitation of the chicken production will be raised. As the cost is decreased, chicken and egg farms can employ the time for doing other activities such as feeding, vaccination, or research and development for their farms.

Road transport is an essential means and widely used in the transshipment of agricultural goods in Thailand. However, this type of transportation depends on energy from fuels, which has high energy consumption. Regarding the costs of transportation in 2018, the overall fuel usage of road transportation increased by 0.37% when compared with the previous year. When considering diesel and gasoline usage, it increased by 2.68% and 3.43%, respectively. Besides, the data showed the trends in increasing road transport costs each year (Energy Policy and Planning Office) [1]. Therefore, it is shown that the more the development in logistics, the more the fuel consumption. Moreover, the combustion of fuels in transportation causes a lot of air pollution, such as nitrogen oxides (N_2O), carbon dioxides (CO_2), and particulate matter (PM). Fuel conservation, therefore, is necessary to be considered together with the growth of various industries at the same time to prevent the fuel shortage in the future as well as to reduce the impact on the environment.

Previously, Srivarapongse and Pijitbanjong [2] studied the assignment problem in the agricultural industry, which presented the generalized assignment problem (S-GAP) model. In the first step, the driver was assigned to the truck. Then, in the second step, the truck was assigned to the harvesting of sugarcane with the objective to create the maximum land size that could be employed for harvesting in a day. However, this research determined the most appropriate amount of chickens and the most fitted trucks to transport chickens to the factories. Each type of truck had a different capacity to carry chickens and had a different fuel consumption rate, causing the assignment of each type of truck to have different costs. And in this case, the objective was to obtain the lowest total cost. Nevertheless, the study of Srivarapongse and Pijitbanjong never mentioned this matter. In addition, this research also considered different road conditions for transportation, and their difference would affect the rate of fuel consumption as well. Therefore, this research had limitations closer to real-life working.

The research motivation is to increase the profitability of farmers and all stakeholders related to the broiler industry, by reducing operational costs from the current situation. The contributions of this paper are as follows: (1) This research combines environmental care in the assignment problem by considering road categories that impact fuel consumption. This has been a new feature for this kind of problem. (2) The proposed method is a modified metaheuristic method which was developed for solving only this problem. (3) The case study, which is a real-world problem, occurred in Thailand. This paper fulfills the gap in the literature by determining the most appropriate amount of chickens and the most fitted trucks to transport chickens to the factories. Each type of truck had a different capacity to carry chickens.

This research consisted of the following structures. Section 2 provides the survey of the previous literature, Section 3 presents the problem statement, the methodology is presented in Section 4, Section 5 reports the computational result, and Section 6 provides conclusion and suggestion.

2. Literature Review

Assignment problem (AP) means the problem of the task allocation to an agent. Each job is different and each employee has different expertise, resulting in unequal time spent in working, and the cost of assigning jobs to each one is different as well. Therefore, the problem is how to assign the task so that the total cost shall be the lowest, with an important condition that the assignment must be one on one basis. In other words, once an assignment has been assigned to an agent, it cannot be assigned to another. On the other hand, if an agent gets assigned a task, he/she does not get assigned another task.

The generalized assignment problem (GAP) is an extended type of the assignment problem (AP), which can assign multiple jobs to an employee, whereby the different assignments might require different resources. Ross and Soland [3] first presented GAPs and proved that GAPs were NP-hard problems [4]. Later, it was proven to be complete NP by Chu and Beasley et al. [5]. The exact procedures were presented and executed with the generated dataset without more added constraints.

GAP has been revealed extensively by plenty of researchers who are trying to solve practical problems. Similarly, Osorio and Laguna [6] resolved GAP issues by considering work availability and rotation on a working day. Alfares [7] and Elshafei [8] also studied the same problem but considered additional working days in other GAP extensions in order to consider the assignment more than once, which is called multilevel GAP. Moreover, searching for a location and task allocation were considered together with the GAP solution [9].

Dantzig[10] proposed a problem solution of simplex assignment by presenting assignment problems in linear programming problems and was able to use the simplex methods to execute problems. However, it has limitations to the range of the tested instances, a value of decision variables, equations, and limitations, including tools used to find the answers (computer). In other words, if the limitations were too much or the computer had not got enough capabilities, then the answer could not be found by using the simplex method. Therefore, the Hungarian method was a method proposed by Kuhn [11], which was another method used for resolving the assignments quickly. Furthermore, Ford and Fulkerson (1956) said that if the assignment problem with a size of 20×20 was solved with the simplex method, it would take at least an hour. However, if the Hungarian method was used, it would take about 30 minutes. It was the optimal manual calculation, which was considered much faster.

The metaheuristic method is necessary to solve GAP problems. Its well-known methods were variable neighborhood search [12,13], colony optimization [14], adaptive large neighborhood search [15], differential evolution (DE) [16,17], and genetic algorithm (GA) [18]. DE and GA were also applied to image encryption application for classification of COVID-19 patients from chest CT images and drug interaction prediction efficiency [19–27].

Particle swarm optimization (PSO) is a method that used natural imitation behavior by relying on the foraging role of animals, such as birds, fish, or other animals that have the behavior of finding food together. Each animal or particle shall find food by moving from the current point to the new point by using the direction and speed from particle best (P_{best}) and the global best (G_{best}) to search for the best food resources. This is the work to benefit the herd mainly. The PSO was first published by Kennedy and Eberhart (1995) [28]. Since then, this method has been discussed and used to execute various problems. Besides, Rapeepan and Kanchana (2016) [29] presented the particle swarm optimization (PSO) which was applied to execute the vehicle routing problem (VRP) with the service points and demands that could be changed.

PSO methods have been extensively used to solve various problems, for instance, the assignment problems and multilevel location-allocation problems [30]. Besides, VRP with time window problems were solved by using the hybrid PSO method. In other words, other solutions were assembled into existing problems [31]. Solutions for pricing and production quantities were found by employing the hybrid particle swarm optimization and the differential evolution (DE) [32]. The hybrid particle swarm optimization was improved by employing particle swarm optimization and other methods [33]. Inventory management problems were solved by using the PSO method [34]. The logistics problems of distribution centers were resolved by applying the PSO method [35]. University and polytechnic exam scheduling was modified by employing the PSO method [36].

Green logistics have received attention from business organizations in terms of environmental and ecological factors. When making logistics decisions aside from general economic costs, these also included pollution, accidents, resource use, and the risk of climate change [37]. Green logistics and green transportation were becoming part of supply chain management, which stimulated environmental awareness in transportation decisions, in addition to transportation costs as in the past [38].

Nowadays, customers and business organizations place importance on environmental impacts due to the transportation of agricultural products which is a large type of energy consumption and greenhouse gas emissions (GHG). Many organizations are thus aware of the need to assess and reduce the environmental impacts of their activities and services. However, society is still concerned about the impact of human activities and the carelessness of using resources. There are a lot of research studies aiming to reduce the negative effects (i.e., fuel consumption and greenhouse gas emissions) from logistics activities to the environment such as pollution-routing problem (PRP) [39–41], green-VRP (G-VRP) [42], and the green-VRP pickup and delivery problem [43].

There were previous research studies on the death of chickens, its diets, and growth periods, which could increase production rates or reduce death and weight loss [44–46]. Besides, production planning in chicken had been studied by Mohaddes [47], who has taken efforts to decrease the cost of raising chickens by revealing the most appropriate type of food and the number of chickens in all farms. Furthermore, Demircan et al. [48] sought to maximize the profit by considering the appropriate size of farm in order to provide the optimum feed consumption, including production costs and profits. In this study, appropriate farm sizes and parameters gained from former research studies were used for manufacturing planning. Therefore, each farm had to have a good production plan and the production plant would determine the chicken needs for each period, and the chicken produced from the farm had to meet the requirement of the production factory. As each farm had different sizes, it has to make a good assignment, so that the chicken demand of the factory was properly met. The highest possible profit will be achieved when all the needs are fulfilled. The profit of production planning is revenue, minus costs, which are probably the operating costs and transportation costs of the farm. All farms are considered to have the same production costs, but the delivery cost is different because they are located in different areas. The models presented in this research would determine the optimal number of chickens and the most fitted truck to transport the chicken to the production plant. Each type of truck has a different capacity to transport chickens and a different fuel consumption rate, resulting in different assignment costs as well. In addition, different road conditions are considered, and their difference would affect the rate of fuel consumption. Nevertheless, there has never been any research on GAP problems since GAP is a difficult problem. If there are many farms, it is not possible to execute problems using the exact procedure. Therefore, the particle swarm optimization (PSO) has been presented to execute the problems in this research.

3. Problem Statement

The case study is the multistage assignment problem, which is used to assign the appropriate vehicle type suitable for chicken transportation directly from the chicken farms to the egg farms. There are 4 categories of vehicles, which are truck that has ten, six, and four wheels and the modified version of the four-wheel truck aiming to keep the total cost minimum. The cost of the assignment in this case study consists of 3 parts, which are (1) the cost of transshipment depending on the category of vehicle with different fuel consumption rate and the distance in transportation, (2) the cost of transshipment depending on the category of road condition and the distance to travel, and (3) the opportunity cost.

Multistage assignment problem was studied by considering the appropriate vehicle type suitable for chicken transportation straight from the chicken farms to the egg farms, and the limitations are listed as follows:

(1) It is direct transportation in which there was no picking the chickens up from different farms and not being transport to other egg farms. The egg farms require quality control and good breed, to protect against communicable diseases.

(2) A chicken ranch may transport to many egg ranches.

(3) Once the chickens have been produced, all of them can be sold.

(4) The egg ranch may obtain chickens from many different farms, but must not over the capabilities of such a farm.

(5) Egg ranch shall get not less than 50 percent of the demand of their farm.

(6) The time required for transportation should not exceed 8 hours, beginning with loading chicken to the vehicle, transporting, and taking them down.

(7) The vehicles employed for transportation are acceptable for needs.

(8) Chicken ranch can employ more than one category of vehicles.

This assignment is the multistage assignment problem beginning by assigning the truck type (4 types). Each truck has a different capacity to transport chickens and a different fuel consumption rate, which causes different assignment costs. Therefore, the researcher aims to study the multistage assignment problem by considering the appropriate vehicle type to transport chickens straight from the chicken farms to the egg farms for the cheapest total cost.

The multistage assignment problem of the case study is to assign the layer hen farming to feed chickens starting from hatching and then raise them until they can be sold to the egg farming. Hatching and feeding of layer chickens require different technologies compared to raising chickens to lay eggs. The chicken farm consists of 40 farms, and all are capable of producing chickens differently, as shown in Table 1. In addition, this assignment is the multilevel assignment beginning by assigning the truck type, which consists of 4 types as shown in Table 2. Each type of truck has a different capacity to transport chickens and different fuel consumption rate, resulting in different assignment costs. However, the assignment must be under the conditions or restrictions specified.

Table 1: Details of 40 chicken farms.

Types of chicken farms	Production capability (chickens/farm)	Number of farms	Total amount (chickens)
Large farm	20,000	8	160,000
Medium farm	10,000	12	120,000
Small farm	5,000	20	100,000
Total	35,000	40	380,000

Egg production can be obtained by using layer chickens that are obtained from the chicken farms in which there are 60 egg farms. Each farm has a different demand for chickens, but in total there is a demand for 388,000 chickens, as shown in Table 3.

The condition of the road used to transport shall affect the speed of the truck and its speed influences the rate of fuel consumption as well. The speed varies according to road conditions. For example, the main roads connecting the province are usually large with 4–6 traffic lanes. The vehicles can speed up than the roads with narrower lanes. The road surface also affects the speed, and the roads with a smooth surface, such as paved roads, can be driven faster than the roads with a rough surface (a concrete road, a damaged road, and bumpy road surface). In this research, the road condition is divided into 5 types [49], with each road having different average speeds and fuel consumption rates as shown in Table 4.

In general, the cost of transporting goods varies with distance, so the mathematical models shall try to find the shortest route as the answer to the problem, resulting in the lowest total cost. However, this research presents different perspectives with the purpose of finding the lowest grand fuel transportation cost. Therefore, the mathematical model shall try to choose the transportation route with the lowest fuel consumption first regardless of the distance. Calculation examples are shown for a better understanding of the pattern of the problem in this research as follows.

The road types among 6 farms are specified in Table 5 and its distance is revealed in Table 6. When the fuel usage rate is multiplied by the traveling distance, the result shows the amount of fuel used to travel among 6 farms. For example, traveling from Farm 1 to Farm 6 has a distance of 23 kilometers, which is the road type C (fuel consumption rate of 0.098 liter/kilometer). Therefore, the fuel consumption on this route is $23 \times 0.098 = 2.254$ liters, as shown in Table 7.

Transportation solutions generally focus on finding the shortest route as it leads to the lowest transportation costs. Nevertheless, shorter routes may consume more fuel. For instance, travel distances from Farm 6 to Farm 2 and Farm 6 to Farm 3 are equal 24 kilometers and 26 kilometers, respectively. When considering the fuel used in both directions, which is 2.688 liters and 2.548 liters, it can be seen that the route from Farms 6–3 is longer than the route from Farms 2–6, but less fuel is used.

Table 2: Types of vehicles used in transportation.

Types of vehicles	Number of chickens	Rate of fuel consumption (liter/kilometer)
Ten-wheel truck	12,000	3.2
Six-wheel truck	8,000	5.0
Four-wheel truck	4,000	8.0
Modified four-wheel truck	2,000	10.0

Table 3: Details of egg farms.

Types of egg farms	Number of laying chickens	Number of farms	Total amount (chickens)
A	10,000	18	180,000
B	8,000	14	112,000
C	5,000	10	50,000
D	3,000	10	30,000
E	2,000	8	16,000
Total	28,000	60	388,000

Table 4: Road types and fuel consumption rates.

Road types	Average speed (km/hr)	Fuel consumption rate (liter/kilometer)
A	<50	0.112
B	51–60	0.090
C	61–70	0.098
D	71–80	0.098
E	81–90	0.102

Table 5: Example of road type metrics among 6 farms.

Farm	1	2	3	4	5	6
1	—	A	C	B	B	C
2	A	—	B	C	A	A
3	C	B	—	B	B	C
4	B	C	B	—	D	A
5	B	A	B	D	—	C
6	C	A	C	A	C	—

Table 6: Example of distance metrics among 6 farms.

Farm	1	2	3	4	5	6
1	—	17	39	37	27	23
2	17	—	28	31	19	24
3	39	28	—	42	16	26
4	37	31	42	—	29	18
5	27	19	16	29	—	39
6	23	24	26	18	39	—

Table 7: Example of fuel metrics for traveling among 6 farms.

Farm	1	2	3	4	5	6
1	—	1.904	3.822	3.330	2.430	2.254
2	1.904	—	2.520	3.038	2.128	2.688
3	3.822	2.520	—	3.780	1.440	2.548
4	3.330	3.038	3.780	—	2.842	2.016
5	2.430	2.128	1.440	2.842	—	3.822
6	2.254	2.688	2.548	2.016	3.822	—

4. Mathematical Model Formulation

Indices

$i = 1, 2, 3, \ldots, I$ (types of trucks from 1 to I)
$j = 1, 2, 3, \ldots, J$ (chicken farms 1 to J)
$k = 1, 2, 3, \ldots, K$ (transportation rounds 1 to K)
$l = 1, 2, 3, \ldots, L$ (egg farms 1 to L)

Decision variables

$x_{ijkl} = 1$, chickens are assigned to the truck i to ship the chickens from the farm j in the round k to deliver them to the egg farm l
$= 0$, other cases
$y_{ijkl} = 1$, there is the vehicle type i to deliver the chickens from the farm j in the round k to transport to the egg farm l
$= 0$, other cases
q_{ijkl} = the amount of chickens shipped by vehicle i to deliver chickens from the farm j in the round k to the egg ranch l

Parameters

I = category of the truck
J = amount of chicken ranch
K = amount of rounds of the transportation of the truck
L = number of egg farms
c_{ijl} = cost of assigning truck i to transport the chickens apart from the farm j to the egg farm l
e_{ijl} = cost of transportation on the truck i to transport the chickens apart from the farm j to the egg farm l

t_i = the limitation to deliver chickens of the vehicle type i

O_{ijl} = the opportunity cost of inefficient transportation by truck i to deliver chickens from the farm j to the egg ranch l (unit: baht per chicken)

d_l = the chicken demand from the egg farm l

s_j = the capacity to produce chickens from the farm j

t^1_{jikl} = the time taken to load the chickens up the vehicle i from chicken ranch j in the round k to egg farm l

p^1_{ji} = the staff ability of the chicken farm j to load up the chickens to the truck i

t^2_{ijkl} = the time taken to bring the chickens down from the truck i from chicken ranch j in the round k at the egg ranch l

p^2_{li} = the staff ability of the egg ranch l to bring chickens down from the truck type i.

t^3_{ijkl} = traveling time of vehicle type i from the service center to the chicken farm j to transport the chicken in the round k to the egg ranch l

t^d_{lj} = the time taken for transporting the chicken from the service center to the chicken ranch j, and the time spent to travel from the chicken ranch j to the egg ranch l

t^4_{ijkl} = the total operating time of the vehicle type i to deliver chicken from ranch j in the round k to egg farm l

t^5_i = the limited predefined working time of truck i

Objective function:

$$\min \sum_{i}^{I}\sum_{j}^{J}\sum_{k}^{K}\sum_{l}^{L}\left(c_{ijl} \times x_{ijkl}\right) + \sum_{i}^{I}\sum_{j}^{J}\sum_{k}^{K}\sum_{l}^{L}\left(e_{ijl} \times y_{ijkl}\right)$$
$$\sum_{i}^{I}\sum_{j}^{J}\sum_{k}^{K}\sum_{l}^{L}\left[\left(\left(x_{ijkl} \times t_i\right) - q_{ijkl}\right) \times O_{ijl}\right]. \tag{1}$$

Constraints:

$$x_{ijkl} \in \{0, 1\} \quad \forall ijkl, \tag{2}$$

$$q_{ijkl} \in 0 \quad \forall ijkl, \tag{3}$$

$$\sum_{l}^{L} x_{ijkl} \le 1 \quad \forall ijk, \tag{4}$$

$$\sum_{i}^{I}\sum_{j}^{J}\sum_{k}^{K} q_{ijkl} \le d_l \quad \forall l, \tag{5}$$

$$\sum_{i}^{I}\sum_{j}^{J}\sum_{k}^{K} q_{ijkl} \ge 0.5 d_l \quad \forall l, \tag{6}$$

$$\sum_{i}^{I}\sum_{j}^{J}\sum_{k}^{K} q_{ijkl} = s_j \quad \forall j, \tag{7}$$

$$q_{ijkl} \le M \times x_{ijkl} \quad \forall ijkl, \tag{8}$$

$$q_{ijkl} \le t_i \quad \forall ijkl, \tag{9}$$

$$t^1_{ijkl} = p^1_{ji_{ij}} \times q_{ijkl}, \quad \forall ijkl, \tag{10}$$

$$t^2_{ijkl} = p^2_{li} \times q_{ijkl} \quad \forall ijkl, \tag{11}$$

$$t^3_{ijkl} = t^d_{lj} \times x_{ijkl} \quad \forall ijkl, \tag{12}$$

$$t^4_{ijkl} = t^1_{ijkl} + t^2_{ijkl} + t^3_{ijkl} \quad \forall ijkl, \tag{13}$$

$$t^4_{ijkl} \le t^5_i \quad \forall ijkl, \tag{14}$$

$$x_{ijkl} \ge x_{ij(k+1)l}, \tag{15}$$

This mathematical model was formulated to execute the multistage assignment problem. The purpose function consists of 3 cost terms: (1) the cost of transshipment depending on the category of vehicle with different fuel consumption rate and the distance in transportation, (2) the cost of transshipment depending on the type of road condition and the distance to travel, and (3) the opportunity cost occurred due to not full capacity of transporting.

Various limitations relating to the decision variables are as follows: X_{ijkl} is a positive integer and has a value of 0 or 1 only (equation (2)). The amount of chickens to be transported (q_{ijkl}) must be a positive integer (equation (3)). The egg ranch (l) can only get chickens apart from the chicken farm (j) by employing the truck (i) in the round (k) only a single time (equation (4)). Equation 4 is also used avoid the shipping of the chickens from different sources to the same egg farm. Egg farm (l) might not receive chickens as its demand (equation (5)) and each egg farm (l) shall get not less than 50 percent of the chickens, according to their requirement (equation (6)). Each chicken ranch (j) is able to send chickens to all egg farms with no leftover at the farm (equation (7)). If the assignment is not occurred ($X_{ijkl} = 0$), the amount to be delivered must be equal to 0 ($q_{ijkl} = 0$) (equation (8)). The amount to be delivered (q) in each cycle must not be over the capacity of the vehicle (equation (9)). The time consumed on storing up chickens to the vehicle is shown in equation (10). Equation (11) is used to determine the amount of time spent to load out the chicken that is transported to egg farm l. Equation (12) determines the total traveling time of truck i to deliver the assigned egg farm. Equations (13) and (14) determine the total time that truck i spent to deliver the chicken to the egg farm and this time should not exceed the predefined period of time (working time), and equation (15) is used to control that round k must be executed before round $k + 1$.

5. Methodology

The multistage assignment problem of the case study is to assign the fitted vehicle type for the chickens transporting straight from the chicken ranch to the egg ranch by employing the cheapest total cost. Therefore, the author has applied and developed particle swarm optimization (PSO) and modified particle swarm optimization (modified PSO) in Sections 5.1 and 5.2, respectively.

5.1. Particle Swarm Optimization (PSO). Particle swarm optimization is one of the most widely used methods, which was first mentioned by Kennedy and Eberhart in 1995. It can find the answer by using the cooperation between the particle and its swarm and each particle searches for the appropriate value from the current location. The direction and velocity for the next position are known by considering the former direction and speed from the particle best (P_{best}) and the global best (G_{best}). The relationship can be shown as follows:

$$V_{t+1} = c_0 V_t + c_1 r_1 \left(P_{\text{best}} - X_t\right) + c_2 r_2 \left(G_{\text{best}} - X_t\right), \quad (16)$$

$$X_{t+1} = X_t + V_{t+1}, \quad (17)$$

where V_{t+1} is the speed of each particle for traveling to a new position, V_t is the speed of each particle for the existing position, r_1 and r_2 are random number values between 0 and 1, c_0, c_1, and c_2 are the learning coefficient constants, X_t is the existing position, X_{t+1} is the new position, P_{best} is the particle best, and G_{best} is the global best.

When the particle recognizes the new velocity (V_{t+1}), such particle shall change from its existing position (X_t) by using the said velocity to the new position (X_{t+1}). When each particle changes from its existing position to the new position, it must use its particle best and global best (equation (17)). Therefore, when the particle best is found, the whole changes to that position. Such a position might be the only local optimal solution. In order to find the solution, the particle swarm optimization method is detailed as follows.

5.1.1. Encoding Method. The encoding method uses the same principle with the differential evolution (DE), which assigns a random number between 0 and 1 for every particle in each particle. Then the random number of each particle is sorted in ascending order. Particles with the lowest random number shall be chosen first. Random numbers of truck types are shown in Table 8. Random numbers of the chicken farm are revealed in Table 9 and random numbers of egg farms are revealed in Table 10. After that, the lowest random numbers will be considered first, and then the highest will be chosen last. Then the sequence from the said guidelines shall be decoded.

5.1.2. Decoding Method. The decoding method uses the same principle with the differential evolution (DE), where each cycle begins with determining the quantity to be transported by comparing the number of chickens in the first chicken farm with the demand of the first egg farm. If any amount is less, such amount shall be transported. To comply with the constraints of the case study, each egg farm shall receive at least 50 percent according to the chicken demand. However, the egg farms in the last rank may not gain the chickens as they need:

Table 8: Initial particle of the truck type.

Particle	Initial particle of the truck type				
	1	2	3	4	5
1	0.3662	0.0330	0.8662	0.5309	0.4304
2	0.3738	0.2698	0.5456	0.9380	0.4344
3	0.6267	0.6280	0.5089	0.0926	0.3759
4	0.6083	0.7464	0.0687	0.8766	0.5045

Table 9: Initial particle of the chicken farm.

Particle	Initial particle of the chicken farm				
	1	2	3	4	5
1	0.2597	0.1471	0.8420	0.4044	0.7124
2	0.3184	0.2953	0.4526	0.8481	0.1213
3	0.1232	0.0995	0.3469	0.3218	0.0456
4	0.9581	0.2122	0.5713	0.8034	0.9386

Table 10: Initial particle of the egg farm.

Particle	Initial particle of the egg farm				
	1	2	3	4	5
1	0.0472	0.6264	0.8370	0.8489	0.2479
2	0.3940	0.7509	0.1076	0.5824	0.5417
3	0.9313	0.2053	0.1851	0.4747	0.0456
4	0.6833	0.5825	0.2744	0.0746	0.5546

$$A^1 = \begin{cases} Q_d^A, & \text{if } Q_d^A \leq Q_s, \\ Q_s, & \text{otherwise,} \end{cases} \quad (18)$$

where A^1 means the quantity to be transported, Q_d^A means the chicken demand (50 percent of the total demand), and Q_s means the number of chickens produced by the chicken farms.

Then the appropriate truck is chosen by considering its capacity that is greater than the quantity to be transported and must be the truck with the capacity closest to the delivered amount.

Once the transportation has been completed, adjustments and validity checks must be recorded to meet the constraints of the case study. If the chicken farm still has chickens left, it will be transported next round until there are no remaining chickens from this farm, and then the next farm will be chosen for further transportation.

The decoding process begins with determining the transport quantity and choosing the appropriate truck type to avoid transportation many times, which results in higher production costs. The chicken farm is able to transport all the chickens without remaining. Egg farms also get the chickens at least 50 percent of the demand. The egg farms in the first particle shall receive all the chickens as they need, while the egg farms in the last order may not receive all the chickens as they need. However, chicken transportation in each round must not be over the capacity of each vehicle and must be used not more than the specified usage hours. When decoding the initial particle, the answer is given in Table 11.

Once the initial particle has been processed (equation (17)), it provides the particle of the truck type according to Table 12, the particle of chicken farms as per Table 13, and

Table 11: The assignment costs from initial particle decoding.

Particle sequences	Transportation cost	Opportunity cost	Assignment cost
1	13,562	3,845	17,407
2	14,610	4,846	19,456
3	14,088	4,811	18,899
4	14,951	4,158	19,109
5	13,433	4,152	17,585

Table 12: The particle of the truck type after past the process of particle swarm optimization (PSO).

Particle	The particle of the truck type				
	1	2	3	4	5
1	0.4820	0.3197	−0.2520	0.1928	0.4129
2	0.5733	0.8521	0.3194	0.5352	0.6262
3	0.3993	0.4981	0.5763	0.4341	1.7402
4	0.2681	−0.1215	0.6809	0.3933	0.1290

Table 13: The particle of chicken farms after past the process of particle swarm optimization (PSO).

Particle	The particle of the chicken farm				
	1	2	3	4	5
1	0.8233	0.4177	0.6567	−0.2485	0.1032
2	0.3865	0.2223	0.6617	0.8264	0.1277
3	0.8358	1.6121	0.1989	0.4907	0.7508
4	0.8178	0.0477	0.6120	−0.5619	−0.2719

Table 14: The particle of egg farms after past the process of particle swarm optimization (PSO).

Particle	The particle of the egg farm				
	1	2	3	4	5
1	0.6685	0.8553	0.5730	0.8024	0.4196
2	1.2680	0.7375	0.9217	0.6807	0.3372
3	−0.1930	0.3455	0.4981	1.0905	1.0754
4	0.1991	−0.3728	0.3575	1.3476	0.5766

Table 15: The particle sequences of the truck type chosen for the assignment.

Particle	The particle of the truck type				
	1	2	3	4	5
1	3	2	1	1	2
2	4	4	2	4	3
3	2	3	3	3	4
4	1	1	4	2	1

Table 16: The particle sequences of the chicken farm chosen for the assignment.

Particle	The particle of the chicken farm				
	1	2	3	4	5
1	3	3	3	2	2
2	1	2	4	4	3
3	4	4	1	3	4
4	2	1	2	1	1

Table 17: The particle sequences of the egg farm chosen for the assignment.

Particle	The particle of the egg farm				
	1	2	3	4	5
1	3	4	3	2	2
2	4	3	4	1	1
3	1	2	2	3	4
4	2	1	1	4	3

Table 18: The particle in the first order to be assigned.

Particle	The particle of the truck type	The particle of the chicken farm	The particle of the egg farm
1	3	3	3
2	4	1	4
3	2	4	1
4	1	2	2

the particle of the egg farm as per Table 14. After that, the values obtained shall be arranged in order to be assigned before-after as in Tables 15–17, respectively.

The decoding can be conducted by taking the 1st particle of the truck category, the chicken farm, and the egg farm in order to be used to arrange the assignment, which can be put in order as in Table 16 and the data in Table 18 are taken for decoding. The assignment of the truck type, suitable for the number of chickens delivered each time, is detailed in Table 19.

There are differences in the assignment of chicken farms and egg farms: (1) the chicken farm shall be assigned continuously to run out the chickens and (2) egg farm, where the chicken demand is divided into two equal parts according to the constraint of the case study.

In the decoding process, the conditions must be checked, such as the remaining working hours of each type of vehicle and the time spent on each shipment. Each chicken farm is able to send out most of the chickens and egg farms shall get at least 50% of all chickens as their requirement. The assignment for small-size samples, a particular type of truck usage, does not over the specified number of hours. The details are as in Table 20. The assignment cost consists of the main cost which is the cost incurred due to transportation, including the distance to transport, fuel consumption rate, and fuel prices. If the transportation capacity is not full, it causes an increase in the opportunity cost. If there is no free space, the opportunity cost shall be 0, as in Table 21.

Table 19: The particle in the first order to be assigned.

No.	Truck		Cycle	Chicken farm				Egg farm			
	Type	Space		Farm	Supply	Assign	Remaining	Farm	Demand (50%)	Assign	Remaining
1	2	3,000	1	3	10,000	5,000	5,000	3	5,000	5,000	0
2	3	1,500	1	3	5,000	2,500	2,500	4	2,500	2,500	0
3	3	1,500	1	3	2,500	2,500	0	1	5,000	2,500	2,500
4	3	1,500	1	1	10,000	2,500	7,500	1	2,500	2,500	0
5	3	0	1	1	7,500	4,000	3,500	2	4,000	4,000	0
6	3	500	1	1	3,500	3,500	0	3	5,000	3,500	1,500
7	4	500	1	4	5,000	1,500	3,500	3	1,500	1,500	0
8	3	1,500	1	4	3,500	2,500	1,000	4	2,500	2,500	0
9	4	1,000	1	4	1,000	1,000	0	1	5,000	1,000	4,000
10	3	0	1	2	5,000	4,000	1,000	1	4,000	4,000	0
11	4	1,000	1	2	1,000	1,000	0	2	4,000	1,000	3,000

Table 20: Time for assignment (time unit: hours).

No.	Truck type	Cycle	Chicken farm	Egg farm	Assignment	Time to load up	Time to bring down	Time transport	Total time
1	2	1	2	3	5,000	0.45	0.5	2.1	4.8
2	3	1	4	4	2,500	0.175	0.2	1.938	4.038
3	3	1	4	1	2,500	0.175	0.225	0.788	2.988
4	3	1	1	1	2,500	0.25	0.225	2.05	4.813
5	3	1	1	2	4,000	0.4	0.28	1.175	4.25
6	3	1	1	3	3,500	0.35	0.28	2.175	3.9
7	4	1	3	3	1,500	0.15	0.105	0.263	2.4
8	3	1	3	4	2,500	0.2	0.175	2.388	5.463
9	2	1	3	1	5,000	0.45	0.45	2.388	5.563
10	4	1	3	2	1,000	0.1	0.1	2.45	5.938

Table 21: Cost of assignment.

No.	Truck		Cycle	Chicken farm	Egg farm	Transportation cost	Opportunity cost	Total cost
	Type	Space						
1	2	3,000	1	2	3	2,304	864	3,168
2	3	1,500	1	4	4	1,211	454	1,665
3	3	1,500	1	4	1	896	336	1,232
4	3	1,500	1	1	1	1,444	541	1,985
5	3	0	1	1	2	1,275	0	1,275
6	3	500	1	1	3	1,170	146	1,316
7	4	500	1	3	3	576	144	720
8	3	1,500	1	3	4	1,639	615	2,254
9	2	3,000	1	3	1	2,670	1,032	3,702
10	4	1,000	1	3	2	1,425	713	2,138
Total						14,610	4,846	19,456

5.2. Modified Particle Swarm Optimization (Modified PSO).

The update of the particle position has been modified; therefore, we called our modification method as modified PSO. Instead of using formula (17), the following formula is used to update the particles' position to enhance the search performance of the PSO:

$$X_{t+1} = \begin{cases} X_t + V_{t+1}, & \text{if } \mathrm{rand}_t \le CR, \\ R_t, & \text{otherwise,} \end{cases} \tag{19}$$

Particle swarm optimization (PSO) is often trapped on the answer which is the local optimal answer. It may lead to not the most appropriate answer. In order to expand, therefore, the answer is improved and there is no change in the answer after processing 200 iterations, by randomly sampling the number of particles that might change in each particle of the truck types, the number of cycles, chicken farms, and egg farms. Then a new random number (R_t) is assigned to change the location to find the answer as in equation (19). A constant (CR) has a value between 0 and 1, and then a random number (rand_t) is assigned to compare with the CR. If the random value is less than or equal to CR, it shall require to change to the new position as before $(X_t + V_{t+1})$. However, if the random value is greater than the CR value, the new position shall be assigned to the random value between 0 and 1, and the new number (R_t) due to the new position needs to be changed for only certain particles, which no need many changes from the existing searches. From the preliminary experiment, we found that the suitable value of CR is 0.8. The procedure of modified PSO is illustrated as a flowchart in Figure 1.

Table 22 compares the features between modified PSO and DE, and this shows the advantage of the algorithm in this paper. The modified PSO has a lower number of parameters, steps, and processing time used in the process. Moreover, it suites for the complex problem as well.

6. Computational Framework and Result

The proposed method is encoded and processed by Visual Studio C# with a mathematical model by Lingo v.11 via Intel Core ™ i5-2450 M CPU 2.50 GHz Ram, 6 GB, compared with solutions provided by Lingo v. 11 software. The particle swarm optimization (PSO) method has been tested with 3 groups of problems (Chicken Farm × Egg Farm): small size (5 × 5), medium size (10 × 10), and large size (20 × 20). The problem of the case study is that the sample has been run 5 times, and the optimal result is recorded. The details are shown in Table 23.

The proposed algorithm consists of two methods, i.e., particle swarm optimization (PSO) and modified particle swarm optimization (modified PSO) and the presented problem is compared with the best solution gained from Lingo v.11 (Lingo best solution (LBS)) and the differential evolution algorithm given in [50]. The details are presented in Table 24.

6.1. Datasets.

6.1. Datasets. Datasets consist of 3 groups of problems (Chicken Farm × Egg Farm): small-size groups (5 × 5), medium-size groups (10 × 10), and large-size groups (20 × 20), including the problem of the case study, by using the data from Table 23, which are tested and compared with Lingo and DE [50].

For small- and medium-sized problems, the stopping criterion of the Lingo program is set to run until finding the optimal solution. The stopping criterion of PSO and modified PSO is set to run for 5 minutes to be fairly compared with DE proposed by Kaewman et al. [50].

$$\%\text{gap} = \left(\frac{S_V - S_L}{S_V}\right) \times 100\%, \tag{20}$$

where S_V is the answer obtained from the metaheuristic method and S_L is the answer provided by the Lingo program, and the %gap is the percentage difference of answers from both methods.

From Table 25, it is found that there are some answers equal to the exact method of the Lingo program, which shows that the PSO and modified PSO methods are reliable and can be used for further analysis and the %gap of all 3 methods is close to the answer obtained from the Lingo program.

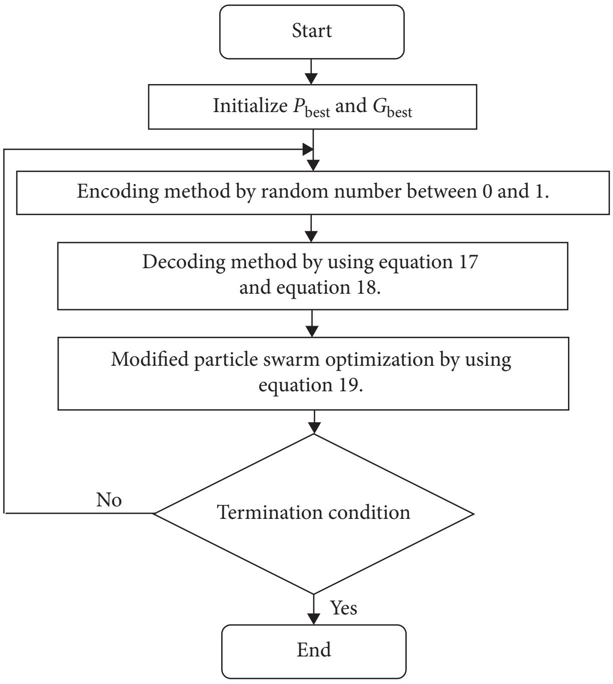

Start
Initialize P_{best} and G_{best}
Encoding method by random number between 0 and 1.
Decoding method by using equation 17 and equation 18.
Modified particle swarm optimization by using equation 19.
Termination condition
No
Yes
End

Figure 1: Flowchart of modified PSO methodology.

The second experiment has been executed with the medium size of test instances. In this dataset, 15 minutes is used to be the stopping criteria of PS and modified PSO which is equal to that of DE proposed by Kaewman [50], and the computational result is presented in Table 26.

According to Table 26, the experimental results in a medium-size sample found that there is an average of the exact method by using the Lingo program in which the average cost of the assignment is 12,431.58 baht. Whereby, the PSO method gives an answer of 12,438.75 baht and the modified PSO method provides an answer of 12,437.08 baht; the answer is close to the means of the exact method. The %gap of all 3 methods is also similar to the answer obtained from the Lingo program. %gap is calculated using formula (20).

The last experiment has been executed with the large size of test instances. In this dataset, the computational time of 30 minutes is used to be the stopping criteria of PS and modified PSO which is equal to that of DE proposed by Kaewman [50]. The results are compared with the best result of optimization software (Lingo v.11) that was found within 72 hours and the result is depicted in Table 27.

Table 27, shows the experimental results of the large-sized of problem instances, the best solution obtained from Lingo program using 72 hours computational time is recorded (best solution within 72 hours). The solution of Lingo programm is used to compare with solution obtained from DE, PSO, and Modified PSO. The DE method provided the %gap different from Lingo −3.52%, indicating that the answer was better than Lingo in a limited time. The

Table 22: Comparison in terms of features of DE and modified PSO.

Features	DE	Modified PSO
Principle	Using the difference among vectors to expand the searching area	Using cooperation among particles to find the best area which contained the best solution
Number of parameters	High	Low
Procedure	4 main steps: (1) Initial population (2) Mutation (3) Recombination (4) Selection	3 main steps: (1) Encoding (2) Decoding (3) Modified particle swarm optimization
Suit for complex problem	Medium	High
Processing time	Long	Short

Table 23: Defining the sample sizes.

Sample	Number of the datasets	Truck type (unit: type)	Round transportation (unit: round)	Chicken farm (unit: farm)	Egg farm (unit: farm)
Small size	12	4	4	5	5
Medium size	12	4	4	10	10
Large size	12	4	4	20	20
The case study	1	4	4	40	60

Table 24: Explanation of the proposed method.

Algorithms	Definition of the proposed algorithm
PSO	Particle swarm optimization
Modified PSO	Modified particle swarm optimization
LBS	Lingo v.11 best solution obtained within predefined time
DE	Differential evolution algorithm obtained by Kaewman et al. [50]

Table 25: Test results of small-size sample groups (5 × 5).

Dataset	LBS	DE	%gap	PSO	%gap	Modified PSO	%gap
1	9,723	9,723	0.00%	9,723	0.00%	9,723	0.00%
2	8,753	8,753	0.00%	8,753	0.00%	8,753	0.00%
3	5,330	5,330	0.00%	5,330	0.00%	5,330	0.00%
4	7,056	7,059	0.04%	7,059	0.04%	7,059	0.04%
5	7,317	7,317	0.00%	7,317	0.00%	7,317	0.00%
6	6,098	6,107	0.15%	6,107	0.15%	6,102	0.07%
7	7,004	7,004	0.00%	7,004	0.00%	7,004	0.00%
8	7,649	7,649	0.00%	7,649	0.00%	7,649	0.00%
9	7,761	7,761	0.00%	7,761	0.00%	7,761	0.00%
10	7,894	7,894	0.00%	7,894	0.00%	7,894	0.00%
11	7,566	7,575	0.12%	7,575	0.12%	7,575	0.12%
12	7,683	7,683	0.00%	7,683	0.00%	7,683	0.00%
Average	7,486.17	7,487.92	0.03%	7,487.92	0.03%	7,487.50	0.02%

PSO and modified PSO methods gave the %gap equal to −3.49% and −3.61%, respectively.

According to Table 28, the statistical test results at a significant level of 0.05 in the large-sized problem groups indicated that the answers of the 3 heuristic methods were not different from the answers obtained from the Lingo program and modified PSO is significantly different from the answer obtained from Lingo and DE method. From the result obtained in Table 28, the best existing heuristics to solve this problem is differential evolution algorithm; it can reduce the total cost by 11.03%.

Table 26: Test results of medium-size sample groups (10 × 10).

Dataset	LBS	DE	%gap	PSO	%gap	Modified PSO	%gap
1	11,989	11,989	0.00%	11,989	0.00%	11,989	0.00%
2	11,397	11,401	0.04%	11,401	0.04%	11,401	0.04%
3	11,572	11,572	0.00%	11,574	0.02%	11,572	0.00%
4	12,898	12,898	0.00%	12,898	0.00%	12,898	0.00%
5	12,184	12,184	0.00%	12,184	0.00%	12,184	0.00%
6	11,315	11,315	0.00%	11,315	0.00%	11,311	−0.04%
7	14,508	14,508	0.00%	14,508	0.00%	14,505	−0.02%
8	12,613	12,613	0.00%	12,616	0.02%	12,613	0.00%
9	10,902	10,921	0.17%	10,921	0.17%	10,921	0.17%
10	11,870	11,890	0.17%	11,893	0.19%	11,890	0.17%
11	15,817	15,849	0.20%	15,849	0.20%	15,847	0.19%
12	12,114	12,114	0.00%	12,117	0.02%	12,114	0.00%
Average	12,431.58	12,437.83	0.05%	12,438.75	0.06%	12,437.08	0.04%

Table 27: Computational result of large-size sample groups (20 × 20).

Dataset	LBS	DE	%gap	PSO	%gap	Modified PSO	%gap
1	33,249	32,716	−1.63%	32,716	−1.63%	32,716	−1.63%
2	29,943	29,094	−2.92%	29,121	−2.82%	29,094	−2.92%
3	37,672	36,128	−4.27%	36,146	−4.22%	36,106	−4.34%
4	38,891	37,781	−2.94%	37,781	−2.94%	37,762	−2.99%
5	39,781	37,895	−4.98%	37,898	−4.97%	37,795	−5.25%
6	31,480	30,084	−4.64%	30,084	−4.64%	30,084	−4.64%
7	58,984	57,738	−2.16%	57,718	−2.19%	57,706	−2.21%
8	89,872	87,573	−2.63%	87,586	−2.61%	87,573	−2.63%
9	90,164	89,079	−1.22%	89,104	−1.19%	89,007	−1.30%
10	35,878	34,871	−2.89%	34,902	−2.80%	34,812	−3.06%
11	29,095	28,049	−3.73%	28,049	−3.73%	28,006	−3.89%
12	29,892	29,152	−2.54%	29,198	−2.38%	29,110	−2.69%
Case study	125,593	114,932	−9.28%	114,968	−9.24%	114,886	−9.32%
Average	51,576.46	49,622.46	−3.52%	44,191.92	−3.49%	44,147.58	−3.61%

Table 28: Statistical test results of answers in large-sized problem groups.

	DE	PSO	Modified PSO
LBS	0.021	0.022	0.020
DE	—	0.021	0.002
PSO	—	—	0.001
Modified PSO	—	—	—

Table 29: Computational results of average of %gap the sample sizes.

No.	Sample	DE (%)	PSO (%)	Modified PSO (%)
1	Small size	0.03	0.03	0.02
2	Medium size	0.05	0.06	0.04
3	Large size	−3.05	−3.01	−3.13
4	Case study	−9.28	−9.24	−9.32
Average (%)		−3.06	−3.04	−3.10

Table 29 compares the average of %gap between modified PSO, PSO, and DE from the literature. The results show that the solutions from modified PSO were better than those obtained by DE and PSO. The modified PSO methods provided the %gap equal to −3.10%. Therefore, the modified PSO method is efficient for the application in solving the assignment problem.

7. Conclusion and Suggestions

The case study problem was the assignment consisting of the main costs incurred from chicken transportation and depended on the difference of truck types using to deliver throughout the road conditions, which affected the fuel costs as well. Besides, the opportunity cost incurred as there was free space on the transporter truck. In which both costs had to be the lowest and the assignment needed to comply with various conditions as specified, making the problem-solving in this case study more complicated.

The nature of the multistage assignment problem and various conditions caused a complicated problem. In which problem-solving with the exact method either the branch and bound method or the Lingo program could not be conducted in a short time. Therefore, problem-solving using alternative methods or the heuristics method was the appropriate choice for resolving this problem. The heuristics used in this study would use the particle swarm optimization (PSO) method.

The particle swarm optimization (PSO) was the method having a small number of parameters, including a short procedure, which saves time to search for answers, but the appropriate answer was likely to not change to find the answer in other locations. This study added additional steps in order to find answers in other areas which was likely that the answer would be the optimal answer (global optimal). The results showed that the modified particle swarm optimization method provided a better answer than the particle swarm optimization method. This method might be suitable for complicated problems than the exact method of the Lingo program. There were also a small number of procedures and parameters, which make it easier to find the lowest cost as well as the amount of fuel used in operations. From the computational result, we found that modified PSO statistically outperforms the best existing heuristics which is DE proposed by Kaewman [50]. It can find 11.03% lower cost than that the DE. It means the modified PSO that we have been developed generates the same result compared with the optimal solution generated from Lingo v.11 (optimization software) in small and medium size of test problems, and when the problem size increases, the optimization software cannot solve the problem to the optimal solution and modified PSO can still find the good result compared with the lower bound generated from the optimization software and perform better than that the best existing heuristics like DE.

Future research should study the problem of more complicated assignments and to comply with the conditions in the real-world (Realistic) or study other metaheuristic methods to solve problems more efficiently, by using hybrid methodologies and developing the exact method to find the answer of the assignment problem. This is another interesting way for further study. The others additional factors should be considered, for instance, the study of capability and performance of each type of vehicle and study of driver skills of driving for each type of road.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This paper was supported by KKU-En Grad Camp 2016, Faculty of Engineering, Khon Kaen University, Thailand. This project was funded by the Faculty of Engineering, Khon Kaen University, Thailand.

References

[1] Oil and Electricity Consumption Situation in 2018. http://www.eppo.go.th/index.php/thenergy-information/situation-oil-electric?orders%5bpublishUp%5d=publishUp&issearch=1.

[2] T. Srivarapongse and P. Pijitbanjong, "Solving a special case of the generalized assignment problem using the modified differential evolution algorithms: a case study in sugarcane harvesting," *Journal of Open Innovation: Technology, Market, and Complexity*, vol. 5, no. 1, p. 5, 2019.

[3] G. T. Ross and R. M. Soland, "A branch and bound algorithm for the generalized assignment problem," *Mathematical Programming*, vol. 8, no. 1, pp. 91–103, 1975.

[4] M. L. Fisher and R. Jaikumar, "A generalized assignment heuristic for vehicle routing," *Networks*, vol. 11, no. 2, pp. 109–124, 1981.

[5] P. C. Chu and J. E. Beasley, "A genetic algorithm for the generalised assignment problem," *Computers & Operations Research*, vol. 24, no. 1, pp. 17–23, 1997.

[6] M. A. Osorio and M. Laguna, "Logic cuts for multilevel generalized assignment problems," *European Journal of Operational Research*, vol. 151, no. 1, pp. 238–246, 2003.

[7] H. K. Alfares, "Optimum workforce scheduling under the (14, 21) days-off timetable," *Journal of Applied Mathematics and Decision Sciences*, vol. 6, no. 3, pp. 191–199, 2002.

[8] M. Elshafei and H. K. Alfares, "A dynamic programming algorithm for days-off scheduling with sequence dependent labor costs," *Journal of Scheduling*, vol. 11, no. 2, pp. 85–93, 2008.

[9] M. Laguna, J. P. Kelly, J. L. Gonzalez Velarde, and F. Glover, "Tabu search for the multilevel generalized assignment problem," *European Journal of Operational Research*, vol. 82, pp. 176–189, 1995.

[10] G. B. Dantzig, "Application of the simplex method to a transportation problem activity analysis of production and allocation," in *Proceedings of the Conference on Linear Programming*, John Wiley and Sons, Inc., Chicago, IL, USA, pp. 359–373, 1951.

[11] H. W. Kuhn, "The hungarian method for the assingnment problem," *Naval Research Logistic Quarterly*, vol. 2, no. 1-2, pp. 83–97, 1956.

[12] B. Maenhout and M. Vanhoucke, "A perturbation matheuristic for the integrated personnel shift and task rescheduling problem," *European Journal of Operational Research*, vol. 269, no. 3, pp. 806–823, 2018.

[13] D. Aksen, O. Kaya, F. Sibel Salman, and Ö. Tüncel, "An adaptive large neighborhood search algorithm for a selective and periodic inventory routing problem," *European Journal of Operational Research*, vol. 239, no. 2, pp. 413–426, 2014.

[14] W. J. Gutjahr and M. S. Rauner, "An ACO algorithm for a dynamic regional nurse scheduling problem in Austria," *Computers & Operations Research*, vol. 3, pp. 66–642, 2007.

[15] P. Shaw, "Using constraint programming and local search methods to solve vehicle routing problems," *Principles and Practice of Constraint Programming-CP98*, vol. 1520, pp. 417–431, 1998.

[16] R. Storn and K. Price, “Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces,” *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[17] C. Şahin and Y. Kuvvetli, “Differential evolution based meta-heuristic algorithm for dynamic continuous berth allocation problem,” *Applied Mathematical Modelling*, vol. 40, pp. 10679–10688, 2016.

[18] J. H. Holland, *Adaptation in Natural and Artificial Systems*, MIT Press, Cambridge, MA, USA, 1975.

[19] D. Singh, V. Chahar, Vaishali, and M. Kaur, “Classification of COVID-19 patients from chest CT images using multi-objective differential evolution-based convolutional neural networks,” *European Journal of Clinical Microbiology & Infectious Diseases*, 2020.

[20] M. Kaur, V. Chahar, and L. Li, “Color image encryption approach based on memetic differential evolution,” *Neural Computing And Applications*, vol. 31, no. 11, pp. 7975–7987, 2019.

[21] M. Kaur and V. Kumar, “Adaptive differential evolution-based lorenz chaotic system for image encryption,” *Arabian Journal for Science and Engineering*, vol. 43, no. 12, pp. 8127–8144, 2018.

[22] P. Shukla, P. Shukla, P. Sharma et al., “Efficient prediction of drug-drug interaction using deep learning models,” *IET Systems Biology*, pp. 1751–8857, 2020.

[23] M. Kaur, D. Singh, and R. Singh Uppal, “Parallel strength pareto evolutionary algorithm-II based image encryption,” *IET Image Processing*, vol. 14, no. 6, pp. 1015–1026, 2020.

[24] A. Gupta, D. Singh, and M. Kaur, “An efficient image encryption using non-dominated sorting genetic algorithm-III based 4-D chaotic maps,” *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 3, pp. 1309–1324, 2019.

[25] M. Kaur and V. Kumar, “Parallel non-dominated sorting genetic algorithm-II-based image encryption technique,” *The Imaging Science Journal*, vol. 66, no. 8, pp. 453–462, 2018.

[26] M. Kaur and V. Kumar, “Beta chaotic map based image encryption using genetic algorithm,” *International Journal of Bifurcation and Chaos*, vol. 28, no. 11, pp. 1850132–11, 2018.

[27] M. Kaur, D. Singh, K. Sun, and U. Rawat, “Color image encryption using non-dominated sorting genetic algorithm with local chaotic search based 5D chaotic map,” *Future Generation Computer Systems*, vol. 107, pp. 333–350, 2020.

[28] J. Kennedy and R. C. Eberhart, “Particle swarm optimization,” in *Proceedings of IEEE International Conference on Neural Networks*, IEEE, Perth, Australia, pp. 1942–1948, November 1995.

[29] P. Rapeepan and S. Kanchana, “Particle swarm optimization for the heterogeneous fleet capacitated vehicle routing problem when number of service points and demand are varying,” in *Proceedings of the International Conference of Logistic and Supply Chain Management System*, Arizona State University, Tempe, AZ, USA, pp. 120–129, June 2016.

[30] C. Anurak and P. Rapeepan, “Particle swam optimization for multi-level location allocation problem under supplier evaluation,” in *Proceedings of the Institute of Industrial Engineers Asian Conference*, National Taiwan University of Science and Technology (NTUST), Taiwan, China, pp. 1237–1250, 2013.

[31] T. M. Stehling, S. R. De Souza, and F. Moacir, “A hybrid particle swarm optimization for solving vehicle routing problem with time window,” in *Proceedings of the GECCO Companion ’15 Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation*, pp. 1489-1490, Spain, July 2015.

[32] W. Ma, M. Wang, and X. Zhu, “Hybrid particle swarm optimization and differential evolution algorithm for bi-level programming problem and its application to pricing and lot-sizing decisions,” *Journal of Intelligent Manufacturing*, vol. 26, no. 3, pp. 471–483, 2012.

[33] A. Khashei-siuki, I. Tadayoni Navaei, and B. Ghahraman, “An improved hybrid optimization algorithm based on particle swarm, ant colony and elitist mutation algorithms,” *Iranian Journal of Science and Technology Transactions of Civil Engineering*, vol. 37, no. 1, pp. 491–501, 2013.

[34] S. M. Orand, A. Mirzazadeh, F. Ahmadzadeh, and F. Talebloo, “Optimization of the inflationary inventory control model under stochastic conditions with simpson approximation: particle swarm optimization approach,” *Iranian Journal of Management Studies*, vol. 8, no. 4, pp. 2203–2220, 2015.

[35] X. Hua, X. Hu, and W. Yuan, “Research optimization on logistics distribution center location based on adaptive particle swarm algorithm,” *Optik*, vol. 127, no. 20, pp. 8443–8450, 2016.

[36] A. Ahmad and F. Shaari, “Solving university/polytechnics exam timetable problem using particle swarm optimization,” in *Proceedings of the 10th International Conference on Ubiquitous Information and Communication*, Trier University, Danang, Vietnam, pp. 1–4, January 2016.

[37] H. Essen, A. Schroten, M. Otten et al., “External costs of transport in europe, update study for 2008,” 2011.

[38] C. Lin, K. L. Choy, G. T. S. Ho, S. H. Chung, and H. Y. Lam, “Survey of green vehicle routing problem: past and future trends,” *Expert Systems with Applications*, vol. 41, no. 4, pp. 1118–1138, 2014.

[39] E. Demir, T. Bektaş, and G. Laporte, “An adaptive large neighborhood search heuristic for the Pollution-Routing Problem,” *European Journal of Operational Research*, vol. 223, no. 2, pp. 346–359, 2012.

[40] M. Barth and K. Boriboonsomsin, *Real-World CO_2 Impacts of Traffic Congestion*, Transportation Research Record, 2008.

[41] E. Demir, T. Bektaş, and G. Laporte, “The bi-objective pollution-routing problem,” *European Journal of Operational Research*, vol. 232, no. 3, pp. 464–478, 2014.

[42] Ç. Koç and I. Karaoglan, “The green vehicle routing problem: a heuristic based exact solution approach,” *Applied Soft Computing*, vol. 39, pp. 154–164, 2016.

[43] M. Soysal, M. Çimen, and E. Demir, “On the mathematical modeling of green one-to-one pickup and delivery problem with road segmentation,” *Journal of Cleaner Production*, vol. 174, pp. 1664–1678, 2018.

[44] C. Lokhorst and E. J. J. Lamaker, “An expert system for monitoring the daily production process in aviary systems for laying hens,” *Computers and Electronics in Agriculture*, vol. 15, no. 3, pp. 215–231, 1996.

[45] V. C. Patel, R. W. McClendon, and J. W. Goodrum, “Development and evaluation of an expert system for egg sorting,” *Computers and Electronics in Agriculture*, vol. 20, no. 2, pp. 97–116, 1998.

[46] K. Mertens, I. Vaesen, J. Löffel et al., “Data-based design of an intelligent control chart for the daily monitoring of the average egg weight,” *Computers and Electronics in Agriculture*, vol. 61, no. 2, pp. 222–232, 2008.

[47] S. A. Mohaddes, “Productivity analysis of eggs production in khorasan razavi province, Iran,” *International Journal of Poultry Science*, vol. 8, no. 12, pp. 1209–1213, 2009.

[48] V. Demircan, H. Yilmaz, Z. Dernek, T. Bal, M. Gül, and H. Koknaroglu, “Economic analysis of different laying hen

farm capacities in Turkey," *Agricultural Economics (Zemědělská Ekonomika)*, vol. 56, no. 10, pp. 489–497, 2010.

[49] R. Akararungruangkul and S. Kaewman, "Modified differential evolution algorithm solving the special case of location routing problem," *Mathematical and Computational Applications*, vol. 23, no. 3, p. 34, 2018.

[50] S. Kaewman, T. Srivarapongse, C. Theeraviriya, and G. Jirasirilerd, "Differential evolution algorithm for multilevel assignment problem: a case study in chicken transportation," *Mathematical and Computational Applications*, vol. 23, no. 4, p. 55, 2018.

Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 7848329, 11 pages
https://doi.org/10.1155/2020/7848329

Research Article

A Novel Parallel Assembly Sequence Planning Method for Complex Products Based on PSOBC

Yanfang Yang,[1,2] Miao Yang,[1] Liang Shu,[3] Shasha Li,[3] and Zhiping Liu[1,2]

[1]*School of Logistic Engineering, Wuhan University of Technology, Wuhan, China*
[2]*Engineering Research Center of Port Logistics Technology and Equipment, Ministry of Education, Wuhan, China*
[3]*The Low-Voltage Apparatus Technology Research Center of Zhejiang, Wenzhou University, Wenzhou, China*

Correspondence should be addressed to Liang Shu; shuliangalbert@163.com

Received 27 December 2019; Revised 18 March 2020; Accepted 19 March 2020; Published 26 May 2020

Guest Editor: Manjit Kaur

Parallel assembly sequence planning (PASP) greatly impacts on efficiency of assembly process. In traditional methods, large scale of matrix calculation still limits efficiency of PASP for complex products. A novel PASP method is proposed to address this issue. To avoid matrix calculation, the synchronized assembly Petri net (SAPN) is firstly established to describe the precedence relationships. Associated with the SAPN model, the PASP process can be implemented via particle swarm optimization based on bacterial chemotaxis (PSOBC). Characterized by an attraction-repulsion phase, PSOBC not only prevents premature convergence to a high degree, but also keeps a more rapid convergence rate than standard particle swarm optimization (PSO) algorithm. Finally, feasibility and effectiveness of the proposed method are verified via a case study. With different assembly parallelism degrees, optimization results show that assembly efficiency of the solution calculated by PSOBC method is 9.0%, 4.2%, and 3.1% better than the standard PSO process.

1. Introduction

ASP is one of the important parts of industrial manufacturing, which is able to provide fast and high efficiency guidelines for equipment assembly. High efficiency of ASP directly determines enterprise profit. However, ASP is usually difficult due to the large number of equipment components (assembly operations) and complicated priorities/constraints relations among components.

Usually ASP can be treated as an NP-hard problem [1], which is generally solved to maximize the multiple benefits during the assembly process [2]. There are several challenges remaining to construct efficient and comprehensive ASP, one of which is the modeling problem. To optimize ASP, the precedence relationships among every individual component need to be investigated. A number of methods have been developed to meet various requirements in the ASP optimization. AND/OR graph model was firstly developed by De Mello et al. [3], and it has been widely employed in assembly sequence generation researches. Precedence matrix was built and embodied into Petri net to generate the reachability graph by Yang et al. [4]. Similar assembly direction matrix, interference matrix, and sequence-relation matrix were introduced to generate optimal assembly sequence for eccentric milling machine in [5]. Such matrix method contributes to generation and acquisition of production information, and it is usually straightforward and easy to be implemented. However, due to the large number of equipment components and the complicated constrains, the size of precedence matrix is generally very large. Computational burden will limit the optimization process. To increase ASP efficiency, some other advanced methods were introduced and developed, e.g., weighted precedence graph [6], assembly tree [7], layout-based hierarchical graph [8], and part concatenation method [9–11]. In these methods, the weighted assembly graph reduces the complexity of ASP through considering qualitative and quantitative constraints. By using the assembly tree method, generation of feasible and stable sequences can be implemented through the recursive construction and stability criteria. Part concatenation method is robust and able to address ASP problem by considering all necessary assembly predicates. However, such models are still matrix based. Not only is enormous human calculation required to investigate complex products, but also human error would occur with a large probability. To address this problem, a SAPN model is developed to describe the operations, priorities, and constraints in assembly process. Each transition is associated with the predecessors by using assembly synchronizers, which illustrates the precedence relationships. The transition status is updated in each assembly operation and the assembility can be identified by the status checking.

ASP optimization could be implemented when the assembly model is developed. Generally, ASP can be categorized into two types, the SASP and the PASP. The assembly operations of SASP are constructed in series and are easy to be implemented. Since all the operations need to follow a linear sequence, the efficiency is limited. Assembly operations of PASP can be performed in parallel and the efficiency can be increased. However, it is still challenging to assemble complex equipment with PASP. To increase efficiency and correctness, the PASP needs to be optimally designed. Moreover, optimization of PASP needs to be subjected to strict predetermined constraints.

Heuristic algorithm is an effective way to solve optimization problem with constraints. Typical methods include genetic algorithm [12–14], PSO [15, 16], ant colony optimization algorithm [17, 18], and other evolutionary algorithms [19–21]. GA is a nonlinear estimation method that generates solutions to optimization problems using techniques inspired by natural evolution [22]. To tackle ASP of complex products, a chaotic PSO approach to generate the optimal or near-optimal assembly sequences of products was developed in [23]. Generally, heuristic searching needs to be implemented with numerical iterations. The coding rules and heuristic rules under predetermined constraints are important to construct the efficient searching. Like what has been discussed, most of the ASP models and the coding rules are matrix based. The searching/iteration performance is limited due to the matrix calculations. We have developed the SAPN model to describe the operations, priorities, and constraints in the assembly process. Coding rules of PSOBC such as position, velocity, and diversity function are represented with index vectors instead of matrix. The consequent efficiency can be increased.

As for the heuristic rules design, the prematurity phenomenon is the major problem that needs to be addressed. The searching result could be attracted to the local optima if the heuristic rules are designed improperly. Inspired by the phenomenon of chemotaxis in bacteria colonies, PSOBC was proposed to improve the performance of standard PSO [24]. Characterized by an attraction-repulsion phase, PSOBC not only prevents premature convergence to a high degree, but also keeps a more rapid convergence rate than standard PSO. This algorithm has been successfully used in areas like CO_2 emission problem [25] and hydropower stations operations [26]. The global and local search performance could be well balanced when the particle swarm diversity is properly controlled. Here we use a similar PSOBC framework to solve the PASP problem based on the proposed SAPN model. The diversity function is built up to avoid population prematurity. Feasibility and effectiveness of the proposed method are also verified via a case study. Optimization results have been compared with standard PSO method under different DPAs.

2. Establishment and Analysis of SAPN

2.1. Basic Petri Net

Definition 1 (Petri net). Let $N = (P, T, F)$ be the basic Petri net which should satisfy the following three conditions:

(1) $P \cup T \neq \emptyset \wedge P \cap T = \emptyset$,

(2) $F \subseteq P \times T \cup T \times P$,

(3) $\text{dom}\ (F) \cup \text{cod}\ (F) = P \cup T$.

Here "×" is the Cartesian product, $\text{dom}\,(F) = \{x \mid \exists y\colon (x, y) \in F\}$, and $\text{cod}\,(F) = \{y \mid \exists x\colon (x, y) \in F\}$. Condition (3) is defined to ensure that there is no isolated place, transition, or arc in the Petri net.

2.2. Symbols in SAPN

Definition 2 (synchronizer). Let $p = (T_1, T_2, (a_1, a_2))$ be a synchronizer, where $\forall t_{i_1} \in T_1$, $t_{i_2} \in T_2$: $p = t_{i_1}^{\bullet} = {}^{\bullet}t_{i_2}$, $w(t_{i_1}, p) = a_2$, $w(p, t_{i_2}) = a_1$, $T_1 = \{t_{11}, t_{12}, \ldots, t_{1m1}\}$, $T_2 = \{t_{21}, t_{22}, \ldots, t_{2m2}\}$, $1 \leq a_1 \leq m_1 = |T_1|$, and $1 \leq a_2 \leq m_2 = |T_2|$.

The scheme of the defined synchronizer is shown in Figure 1. The SAPN consists of a set of synchronizers corresponding to different assembly status. The synchronizers are used to illustrate the parallel and selective relations of each assembly operation, including the corresponding precedence relationships.

Definition 3 (SAPN). Let SAPN =(P, T, F, W, M, PT, TM) be the synchronized assembly Petri net consisting of assembly synchronizers. Places, transitions, arcs, and other tuples are integrated in the net. Detailed explanation of these tuples is given as follows:

$P = \{p_j\}$, $(j = 1, 2, \ldots, m)$ represents the assembly status, and m is the amount of the places.

$T = \{t_i\}$, $(i = 1, 2, \ldots, n)$ represents the assembly operations, and n is the amount of the transitions.

$F = \{f_k\}$, $(k = 1, 2, \ldots, l)$ represents the directed arcs between transitions and places, and l is the amount of the arcs.

$W = \{w_k\}$: $F \longrightarrow \{1, 2, 3, \ldots\}$ represents the weight functions associated with arcs. The weight functions determine the amount of the required tasks for a place to transform into another one. The weight functions should be assigned based on number of the arcs which share the same input/output place in the corresponding synchronizer. For example, there are m_1 arcs between T_1 and p sharing the same output place p in Figure 1, so the weight functions of these arcs should be both

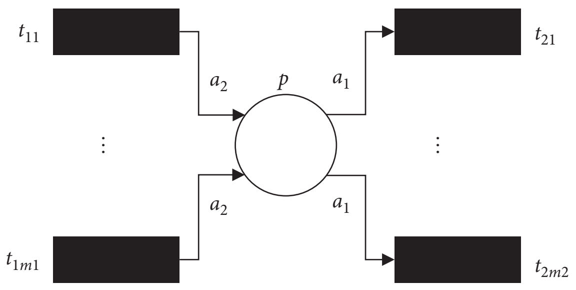


FIGURE 1: Scheme of synchronizer.

assigned as m_1. Also the weight functions of p's output arcs should be both assigned as m_2 according to similar recursion.

$M = \{m_i\}$: $T \longrightarrow \{0,\ 1,\ 2\}$ represents the markings associated with transitions. Here M is defined to represent the transition status. In the assembly process, the marking M is a piecewise constant vector and the values are dependent on the transition status based on the following rules:

$$m_i = \begin{cases} 0, & t_i \text{ is fired,} \\ 1, & t_i \text{ is fireable,} \\ 2, & t_i \text{ is unfireable, while it has't been fired.} \end{cases} \tag{1}$$

$PT = \{pt_{i_2}\}: T \longrightarrow \{t\}_{i_1}$, $\quad (i_1, i_2 = 1, 2, \ldots, n)$ represents the precedence relationships among equipment components. The element pt_{i_2} in PT is defined to collect the preceding transitions of t_{i_1}. In others words, if $\exists p = (T_1, T_2, (a_1, a_2))$: $t_{i_2} \in T_1 \wedge t_{i_1} \in T_2$, then the transitions in T_1 are the predecessors of the transitions in T_2.

$TM = \{tm_i\}$:T represents operation times of the transitions.

2.3. Construction of SAPN. This subsection introduces construction approach of SAPN through the following steps.

Step 1: Generate transitions for each necessary assembly operation.

Step 2: Assign t_{ms}, ms, and p_{ts} to each corresponding transition.

Step 3: Generate the starting place, the ending place, and the places between each transition and its predecessors.

Step 4: Integrate the duplicated input places of each transition.

Step 5: Generate the directed arcs from predecessors to the corresponding places and from the places to the corresponding successors.

Step 6: Assign the weight function to each arc.

SAPN model can be constructed through the detailed steps mentioned above. Step 1 and Step 2 generate the fundamental transitions. Generation of the corresponding places is processed in Step 3 and Step 4. Since SAPN is oriented to the complex products, the elements in SAPN are unavoidably large. Therefore, the integration process in Step 4 somewhat reduces the scale of SAPN model. Step 5 and Step 6 generate and assign the arcs, which link the elements in each synchronizer.

2.4. Analysis of Assembility. A transition which is fireable must satisfy the condition that its preceding transitions have all been fired in the synchronizer. Assembility of each component can be determined from the precedence relationships and the status, i.e., PT and M. The necessary and sufficient condition of t_{i_1} being fireable is given as below:

$$m_{i_1} = 1. \tag{2}$$

Equation (2) can also be rewritten as

$$\forall t_{i_2} \in pt_{i_1}: \sum m_{i_2} = 0 \wedge m_{i_1} \neq 0. \tag{3}$$

It is seen from (3) that t_{i_1} is fireable when its preceding transitions have all been fired; namely, $m_{i_2} = 0$. Analysis of assembility is necessary in the sequence optimization process to avoid violation of precedence relationships.

3. Principles of Sequence Optimization

In this section, PSOBC framework is developed to solve the PASP problem. The algorithm alternates between phases of attraction and repulsion. Once the diversity of population is too high, the individuals will be congregated by attraction force to explore better solution. If the diversity of population is too low, the individuals will be dispersed by repulsion force to ensure better convergence. Thus the sequence optimization can be efficiently performed.

3.1. Problem Statement. Here we use a simple example to illustrate the principles of PASP. The parallel assembly scheme is shown in Figure 2. It is assumed that there are 3 manipulators and 13 individual parts of equipment. The length of each rectangle in Figure 2 represents the assembly time of the corresponding part. Three manipulators are employed and five assembly steps are involved in the process. During the process, *time* represents the operation time of the whole assembly process. To improve efficiency, the goal of sequence planning is set to minimize the assembly time, *time*. DPA can be defined to describe the biggest number of components or parts to be assembled simultaneously, i.e., the amount of the manipulators. According to the definition, the DPA of the example in Figure 2 is 3.

To improve assembly performance, time study is considered to minimize assembly time. The objective function is given in the following formulation:

$$\min(time) = \sum_{s=1}^{S} time_s. \tag{4}$$

In (4), $time_s$ represents operation time of the subassembly s and $time$ represents the whole assembly time.

3.2. Coding Rules. In this subsection, coding rules for PASP problem are proposed according to its features. The rules include particle position, particle velocity, fitness function, and diversity function. Since matrix calculation is avoided, calculation process would be in high efficiency and effectiveness.

Definition 4 (particle position). Let the particle position $\mathbf{X}_d$ be a parallel sequence, which is expressed by a vector as below:

$$\mathbf{X}_d = [\, x_{d1} \;\; x_{d2} \;\; \ldots \;\; x_{di} \;\; \ldots \;\; x_{dn} \,]. \tag{5}$$

Since x_{di} represents subassembly index of transition t_i, it is supposed to be an integer.

Definition 5 (particle velocity). Let the particle velocity $\mathbf{V}_d$ be the updating rule for the particles, which is represented by another vector as below:

$$\mathbf{V}_d = [\, v_{d1} \;\; v_{d2} \;\; \ldots \;\; v_{di} \;\; \ldots \;\; v_{dn} \,]. \tag{6}$$

Since v_{di} represents the updating rule for transition t_i of particle $\mathbf{X}_d$, v_{di} is supposed to be a real number.

Definition 6 (fitness function). Let the fitness function be the formulation to evaluate particle positions as expressed in

$$FT(\mathbf{X}) = C_t - \sum_{s=1}^{S} \left[time_s + \delta_p F_p(s) + \delta_l F_l(s) \right]. \tag{7}$$

In (7), C_t is a constant defined to convert minimization to maximization. Assembly time is considered in the fitness function to optimize assembly efficiency, while the two penalty factors/functions are used to, respectively, investigate precedence relationships and parallel principles.

To calculate the fitness function, the first step is decoding **X** into parallel assembly sequence based on its definition, while the second step is calculating each $time_s$, F_p (s), and F_l (s). $time_s$ is the max operation time in subassemblys. F_p (s) equals the number of unfireable transitions in subassembly s, i.e., the marking values of the transitions equal 2 according to analysis of assembility in the synchronizer (3). After calculating each F_p (s), marking values of transitions in the subassembly should be set as 0, and marking values of the unfired transitions in SDPN should be set as 1 if their preceding transitions have all been fired, or the values remain equal to 2. F_l (s) equals 0 if operation number of subassembly s is less than DPA, or it equals DPA minus operation number.

Definition 7 (diversity of population). Let the diversity of population be the formulation shown in (8) to evaluate the average distance between all particles.

$$F_d(P_l) = \frac{1}{|L||P_l|} \sum_{d=1}^{|P_l|} \sqrt{\sum_{i=1}^{n} (x_{di} - \overline{x}_i)^2}. \tag{8}$$

In (8), $|P_l|$ is the population size, while $\overline{x}_i$ is the mean subassembly indices of t_i in the population. Diversity function is formulated to detect premature convergence, and then attraction/repulsion phase can be implemented to improve algorithm performance.

3.3. Algorithm Optimization. Let $\mathbf{V}_d^g$ be the velocity of particle d in generation g, and $\mathbf{X}_d^g$ be the position of particle d in generation g. The PSOBC algorithm should be implemented with the following equations:

$$\mathbf{V}_d^{g+1} = \omega \times \mathbf{V}_d^g + c_l r_1 \times \left(\mathbf{B}_{id} - \mathbf{X}_d^g \right) + c_g r_2 \left(\mathbf{B}_{gd} - \mathbf{X}_d^g \right), \tag{9}$$

$$\mathbf{X}_d^{g+1} = \mathbf{X}_d^g + \mathbf{V}_d^{g+1}, \tag{10}$$

$$\mathbf{V}_d^{g+1} = \omega \times \mathbf{V}_d^g - c_l r_1 \times \left(\mathbf{W}_{id} - \mathbf{X}_d^g \right) - c_g r_2 \left(\mathbf{W}_{gd} - \mathbf{X}_d^g \right). \tag{11}$$

In attraction phase, (9) and (10) should be exerted to optimize solutions with the principle of approaching local/global best positions. In repulsion phase, (10) and (11) should be exerted to improve convergence with the principle of escaping from local/global worst positions. The calculating rules of (9)~(11) are proposed as below.

$$\mathbf{X}_{d_1} \pm \mathbf{X}_{d_2} = \mathbf{V} = \left[x_{d_1 1} \pm x_{d_2 1}\, x_{d_1 2} \pm x_{d_2 2} \ldots x_{d_1 n} \pm x_{d_2 n} \right], \tag{12}$$

$$k \times \mathbf{V}_d = \mathbf{V} = [\, kv_d\, kv_d \ldots kv_d \,], \tag{13}$$

$$\mathbf{V}_{d_1} \pm \mathbf{V}_{d_2} = \mathbf{V} = \left[v_{d_1 1} \pm v_{d_2 1}\, v_{d_1 2} \pm v_{d_2 2} \ldots v_{d_1 n} \pm v_{d_2 n} \right], \tag{14}$$

$$\mathbf{X}_d + \mathbf{V}_d = \mathbf{X} = [\, x_{d1} + v_{d1}\, x_{d2} + v_{d2} \ldots x_{di} + v_{di} \ldots x_{dn} + v_{dn} \,]. \tag{15}$$

Since xs are integers and vs are real numbers, the elements in **X** should be both rounded to integers after calculation using (15).

4. Case Study

In this paper, a metallurgical reducer is taken as an example to verify the feasibility and the efficiency of the proposed method. The DPAs are selected as 2, 3, and 4, respectively. The explosive diagram of the reducer is shown in Figure 3, including 50 components in total. The main parts of the reducer are gears, boxes, caps, bears, shafts, lubricating rings, and eccentric sleeves.

4.1. Establishing SAPN. Based on the reducer structure, the places, transitions, arcs, and other tuples in SAPN = (P, T, F, W, M, PT, TM) are established in Section 2.3. Structure diagram of the SAPN is built as shown in Figure 4. In the net model, P is the set of places defined to represent component status. p_1 and p_{49} are the starting and ending places. T is the set of transitions, representing the assembly operations. Definitions of the transitions in the SAPN model are given in Table 1. F is the set of arcs, representing flow relations between transition and place. W is the set of weights associated with the arcs. The weight functions are assigned based on the definition mentioned in Section 2.2. For

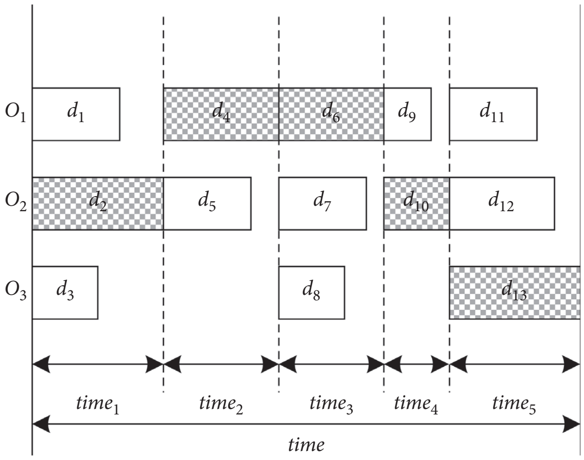


FIGURE 2: Scheme of parallel assembly.

example, weight functions of the arcs between p_2 and t_2, p_5 and t_5, and p_{28} and t_{28} are assigned as 1, while weight function of the arcs between p_{29} and t_{41} is assigned as 3. M is the set of markings associated with transitions. Precedence relationships are illustrated in PT to analyze the component assembility. TM is the set of assembly times associated with each assembly operation.

It is seen from Table 1 that large size of matrices would be generated if the matrix-based method was adopted for metallurgical reducer ASP design, which would be complex and difficult to implement. This problem can be addressed through the proposed PASP framework.

4.2. Population Initialization. Based on the heuristic mechanism of PSOBC, population initialization is required for the sequence optimization. The stages of population initialization can be structured as a flow chart represented in Figure 5. Detailed explanation of the initialization procedure is given in the following steps.

Step 1: Obtain the precedence relationships and the initial status of the components from **PT** and **M**. Build **IP** (an empty set of particles position) as the initial population.

Step 2: Build a new particle position **PDS** (an empty set of transitions).

Step 3: Build a new set of all fireable transitions **RTS** based on analysis of assembility.

Step 4: Build a new assembly step **DS** (an empty set of transitions). If n_r (the amount of transitions in **RTS**) exceeds DPA, randomly push 1~DPA assembly operations into **DS.** Otherwise, push all the operations form **RTS** into **DS.** Update status and marking of the transitions in **DS**, and push the transitions from **DS** into **PDS.**

Step 5: Remove the fired transitions in **RTS** based on **PT** and **M.**

Step 6: If n_r is positive, return to *step 4.* Otherwise, encode **PDS** into a particle position (an index vector) and push it into **IP**; return to *step 7.*

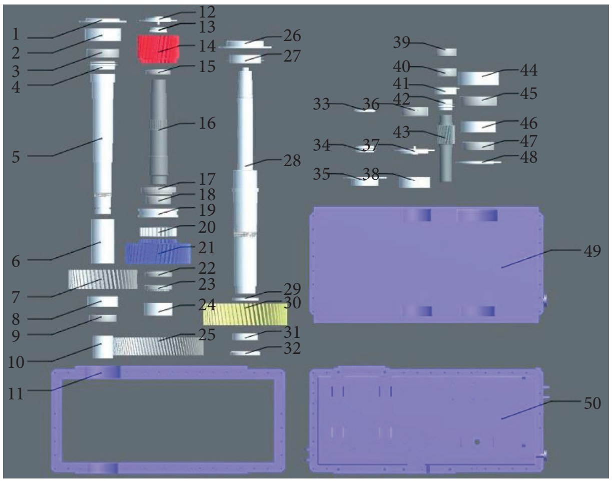


FIGURE 3: Explosive diagram of the metallurgical reducer.

Step 7: If n_p (the amount of particles) exceeds the population size n_t, output the initial population **IP** and end. Otherwise, return to *step 3.*

In the C# environment, the population size n_t is set as 50 and the DPAs are set as 2, 3, and 4, respectively. The three initial populations are exported after running the initialization codes. According to the biologically inspired mechanism of PSOBC, the initial population is the origin for the bacterial evolution.

The key procedure of PASP is the sequence optimization process implemented to find the optimal solution. Main stages of heuristic optimization can be structured as another flow chart represented in Figure 6. Detailed explanation of the procedure is illustrated in the following steps.

4.3. Generation of the Optimal Sequences. The key procedure of PDSP is the sequence optimization process implemented to find the optimal solution. Main stages of heuristic optimization can be structured as another flow chart represented in Figure 6. Detailed explanation of the procedure is illustrated in the following steps:

Step 1: Input the initial population from initialization process. Set phase as "attraction."

Step 2: Update the current best and worst particle position for individuals and the whole population.

Step 3: If the phase is "attraction," calculate the new velocity and position for each particle using (9) and (10). Otherwise, calculate using (10) and (11).

Step 4: Calculate diversity function of the population. If premature convergence occurs, set the phase as "repulsion" and return to *step 6.* Otherwise, return to *step 5.*

Step 5: Calculate diversity function of the population. If premature convergence is escaped, set the phase as "attraction" and return to *step 6.*

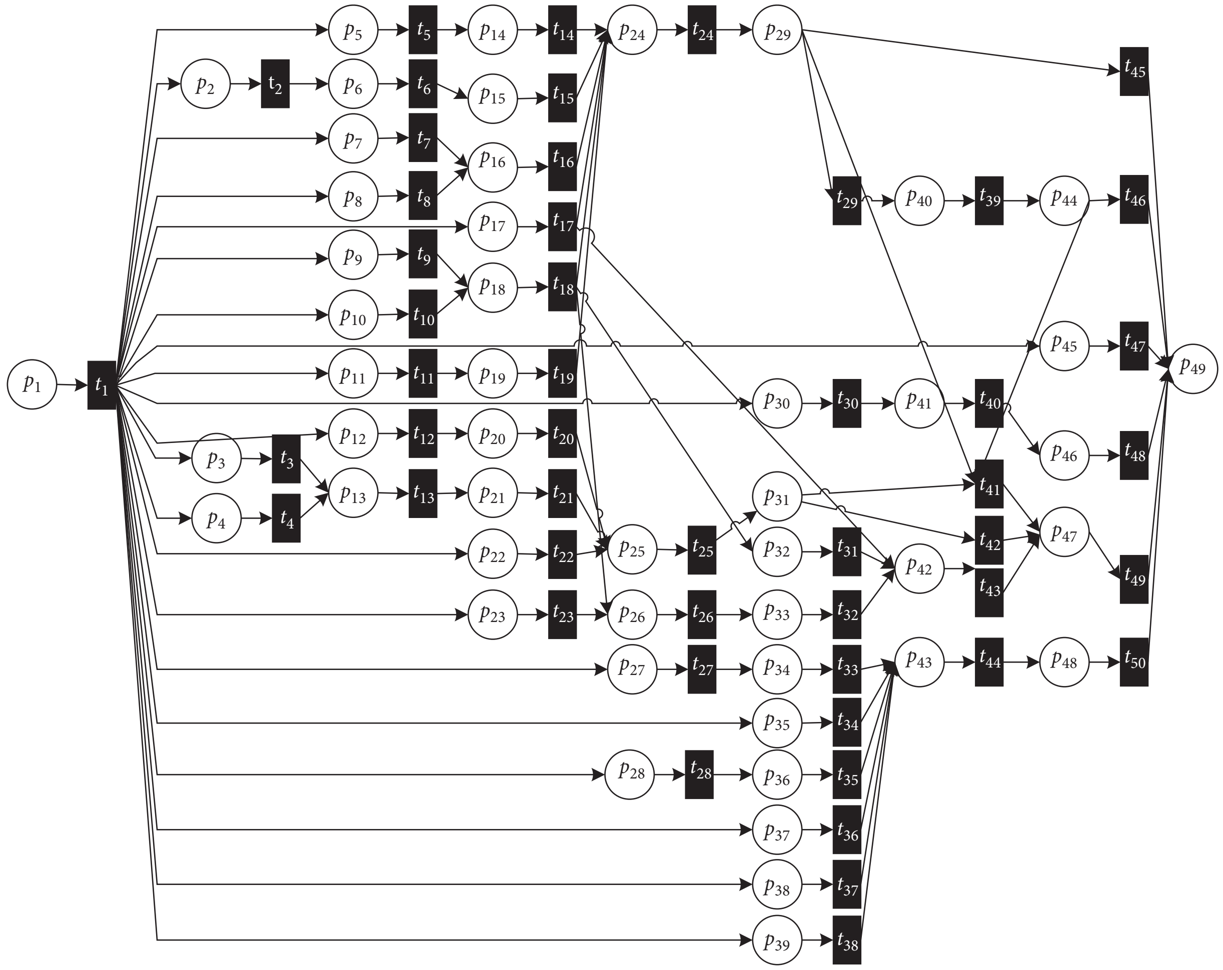


Figure 4: SAPN model of the reducer.

Table 1: Detailed definitions of transitions in the SAPN.

T	Targeted part	No.	PT	TM	M	T	Targeted part	No.	PT	TM	M
t_1	Null	Null	$\varnothing$	0	2	t_{26}	Lubricating ring 3	41	$\{t_{18}, t_{23}\}$	1.8	2
t_2	Bear 12	18	$\{t_1\}$	1.9	2	t_{27}	Bear 4	3	$\{t_1\}$	2.1	2
t_3	Bear 11	45	$\{t_1\}$	1.6	2	t_{28}	Bear 3	9	$\{t_1\}$	2.1	2
t_4	Eccentric sleeve 5	44	$\{t_1\}$	2.1	2	t_{29}	Bear 2	47	$\{t_{24}\}$	2.1	2
t_5	Spline	20	$\{t_1\}$	1.8	2	t_{30}	Lock 3	33	$\{t_1\}$	1.2	2
t_6	Eccentric sleeve 4	24	$\{t_2\}$	2.1	2	t_{31}	Bear 1	39	$\{t_{18}\}$	1.4	2
t_7	Bear 10	15	$\{t_1\}$	1.3	2	t_{32}	Lubricating ring 2	42	$\{t_{26}\}$	1.5	2
t_8	Bear 9	17	$\{t_1\}$	1.3	2	t_{33}	Sleeve 3	2	$\{t_{27}\}$	2.1	2
t_9	Gear 7	23	$\{t_1\}$	1.3	2	t_{34}	Sleeve 2	10	$\{t_1\}$	3.8	2
t_{10}	Gear 6	22	$\{t_1\}$	1.3	2	t_{35}	Eccentric sleeve 2	8	$\{t_{28}\}$	2.1	2
t_{11}	Bear 8	36	$\{t_1\}$	2.1	2	t_{36}	Lubricating ring 1	4	$\{t_1\}$	1.5	2
t_{12}	Lubricating ring 4	27	$\{t_1\}$	3.8	2	t_{37}	Gear 1	7	$\{t_1\}$	3.1	2
t_{13}	Sleeve 3	29	$\{t_3, t_4\}$	1.7	2	t_{38}	Sleeve 1	6	$\{t_1\}$	3.8	2
t_{14}	Inner splined sleeve	19	$\{t_5\}$	2.8	2	t_{39}	Eccentric sleeve 1	46	$\{t_{29}\}$	2.3	2
t_{15}	End cap 4	13	$\{t_6\}$	2.1	2	t_{40}	Lock 2	34	$\{t_{30}\}$	1.9	2
t_{16}	Gear 5	14	$\{t_7, t_8\}$	3.1	2	t_{41}	Middle box	11	$\{t_{31}, t_{24}, t_{39}\}$	2.1	2
t_{17}	Gear 4	25	$\{t_1\}$	3.6	2	t_{42}	Lock 1	32	$\{t_{25}\}$	1.6	2
t_{18}	Gear 3	21	$\{t_9, t_{10}\}$	3.5	2	t_{43}	Input shaft	43	$\{t_{31}, t_{32}\}$	4.2	2
t_{19}	Bear 7	38	$\{t_{11}\}$	2.1	2	t_{44}	Output shaft 1	5	$\{t_{33}\}$	7.2	2
t_{20}	Blank cap 2	26	$\{t_{12}\}$	0.9	2	t_{45}	End cap 3	37	$\{t_{33}\}$	0.9	1
t_{21}	Gear 2	30	$\{t_{13}\}$	3.8	2	t_{46}	End cap 3	37	$\{t_{39}, t_{32}, t_{41}\}$	0.9	1
t_{22}	Bear 6	31	$\{t_1\}$	1.9	2	t_{47}	End cap 2	12	$\{t_1\}$	0.9	1
t_{23}	Bear 5	40	$\{t_1\}$	2.1	2	t_{48}	End cap 1	35	$\{t_{40}\}$	1.8	1
t_{24}	Intermediate shaft	16	$\{t_{14}\sim t_{19}\}$	9.5	2	t_{49}	Upper box	49	$\{t_{42}, t_{43}\}$	4.1	1
t_{25}	Output shaft 2	28	$\{t_{20}\sim t_{22}\}$	12.1	2	t_{50}	Oil collecting hood	1	$\{t_{44}\}$	1.6	1

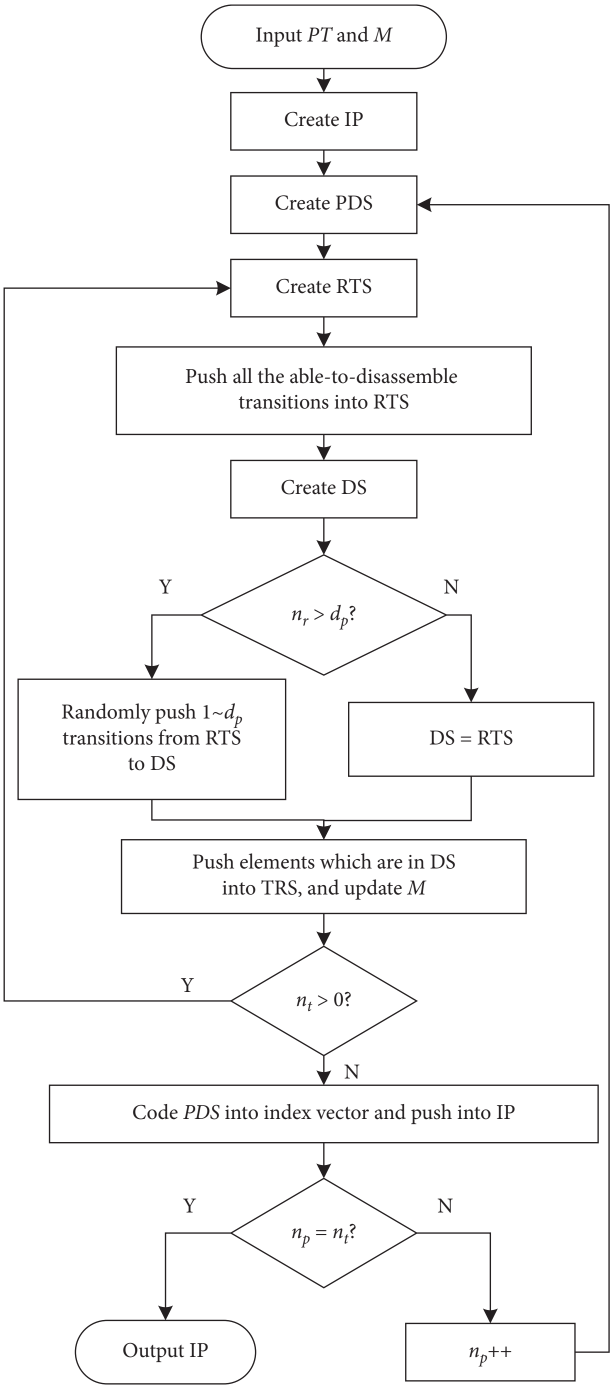

Input PT and M
Create IP
Create PDS
Create RTS
Push all the able-to-disassemble transitions into RTS
Create DS
Y
N
$n_r > d_p$?
Randomly push 1~d_p transitions from RTS to DS
DS = RTS
Push elements which are in DS into TRS, and update M
Y
$n_t > 0$?
N
Code PDS into index vector and push into IP
Y
N
$n_p = n_t$?
Output IP
n_p++

Figure 5: Flow chart of population initialization.

Step 6: If the termination condition is satisfied, decode the optimal solution and end. Otherwise, return to *step 2*.

In the optimization procedure, the parameters are set as $\delta_p = \delta_l = 5$, $itn_{\max} = 150$, $c_l = c_g = 2$, $d_h = 0.1$, and $d_l = 0.001$, and ω will linearly decrease from 1 to 0 during the searching process. Here, the parameters are set not only based on the features of PASP problem and PSOBC algorithm, but also referring to researches in [25, 26].

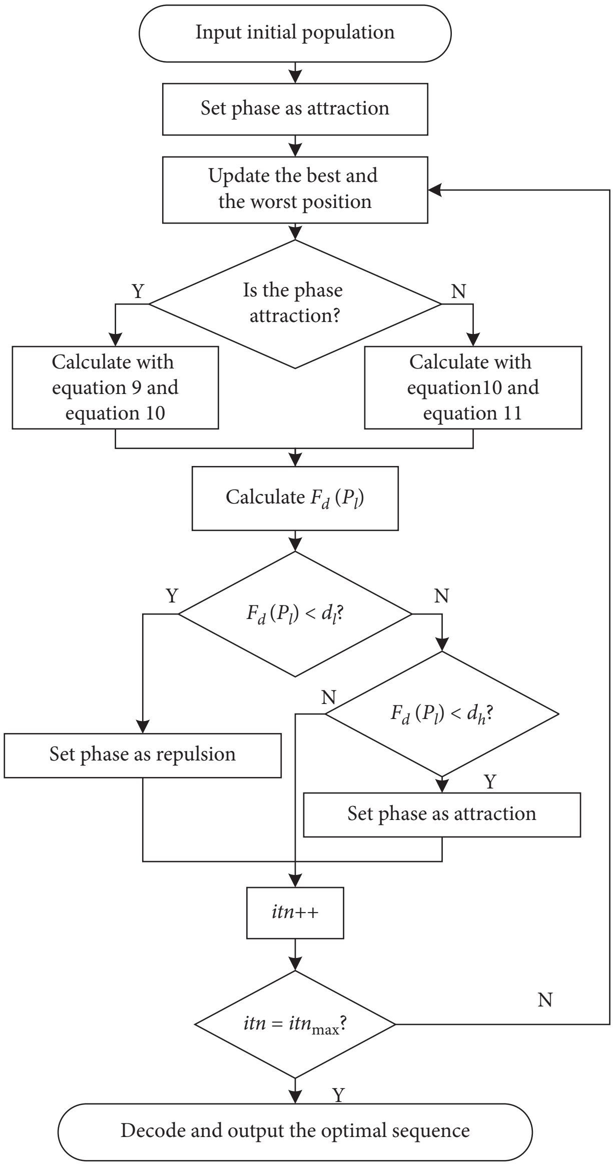
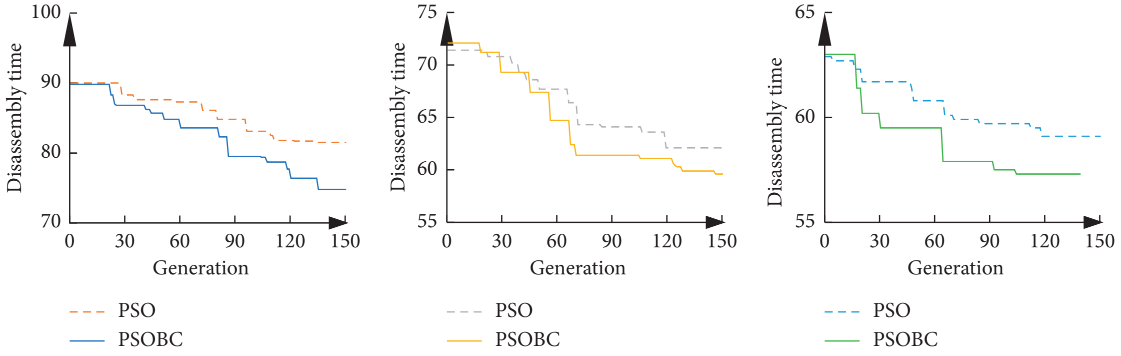


Figure 6: Flow chart of sequence optimization.

4.4. Discussions and Comparisons. Three different simulations are conducted to verify the proposed method when DPAs are settled as 2, 3, and 4, respectively. The proposed PSOBC simulations have been compared with standard PSO method in Figure 7. To keep consistency, the simulation parameters including learning factor, penalty factors, and inertia factor are chosen as the same in both PSOBC and PSO for different DPAs. In Figure 7(a), DPAs are both settled as 2. For PSO simulation, the assembly time is 81.5 seconds. The assembly time of PSOBC method is 74.8 seconds. Compared to the standard PSO, the efficiency is increased by 9.0%. Detailed explanation of 2-DPA assembly sequence is given in Table 2.

In Figure 7(b), DPAs are both settled as 3. It is seen that the assembly time of PSO is 62.1 seconds while the assembly time of PSOBC is 59.6 seconds. Compared to the standard PSO, the assembly efficiency is increased by 4.2%. The



(a) (b) (c)

Figure 7: Optimization results and comparison. (a) Optimization result with 2-DPD. (b) Optimization result with 3-DPD. (c) Optimization result with 4-DPD.

Table 2: Parallel assembly sequence of 2-DPA.

Subassembly index	Subassembly	Targeted parts
1	(17, 20)	Bear 9 and spline
2	(23, 15)	Gear 7 and bear 10
3	(22, 36)	Gear 6 and bear 8
4	(21, 14)	Gear 13 and gear 5
5	(40, 18)	Gear 7 and bear 12
6	(45, 24)	Bear 11 and eccentric sleeve 4
7	(38, 13)	Bear 7 and end cap 4
8	(19, 27)	Inner splined sleeve
9	(41, 44)	Lubricating ring 3 and eccentric sleeve 5
10	(26, 29)	Blank cap 2
11	(31)	Bear 6
12	(30, 25)	Gear 2 and gear 4
13	(28, 16)	Output shaft and intermediate shaft
14	(32, 3)	Lock 1 and bear 4
15	(42, 47)	Lubricating ring 2 and bear 2
16	(39)	Bear 1
17	(46, 6)	Eccentric sleeve 1 and sleeve 1
18	(33, 9)	Lock 3 and bear 3
19	(34, 8)	Lock 2 and eccentric sleeve 2
20	(7)	Gear 1
21	(43, 10)	Input shaft and sleeve 2
22	(2)	Sleeve 3
23	(11, 4)	Middle box and lubricating ring 1
24	(12, 37)	End cap 2 and end cap 3
25	(5, 49)	Output shaft 1 and upper box
26	(1, 35)	Oil collecting hood
27	(48)	Blank cap 1

optimal sequence carried out by PSOBC method is given in Table 3 when DPA is set as 3.

In Figure 7(c), DPAs are both settled as 4. Through standard PSO method, assembly time of the solution is 59.1 seconds. The assembly time by using PSOBC is 57.3 seconds. The assembly efficiency is increased by 3.1%. The optimized sequence carried out by PSOBC is given in Table 4.

Based on the optimization results and comparison, the following conclusions can be drawn:

(1) Due to implementation of SAPN model, the PASP problem can be effectively and efficiently solved for complex products, while huge matrix calculation is avoided and possibility of human error is reduced.

(2) Compared with the standard PSO algorithm, premature convergence can be effectively prevented through repulsion operation. The solution of PASP problem can be further optimized.

Table 3: Parallel assembly sequence of 3-DPA.

Subassembly index	Subassembly	Targeted parts
1	(44, 45, 18)	Eccentric sleeve 5, bear 11, and bear 12
2	(29, 27, 23)	Sleeve 3, lubricating ring 4, and gear 7
3	(26, 21, 22)	Blank cap 2, gear 3, and gear 6
4	(9, 3, 31)	Bear 3, bear 4, and bear 6
5	(6, 24, 20)	Sleeve 1, eccentric sleeve 4, and spline
6	(15, 17, 19)	Bear 10, bear 9, and inner splined sleeve
7	(7, 2, 36)	Gear 1, sleeve 3, and bear 8
8	(30, 38, 25)	Gear 2, bear 7, and gear 4
9	(4, 8)	Lubricating ring 1 and eccentric sleeve 2
10	(14, 10, 13)	Gear 5, sleeve 2, and end cap 4
11	(39, 40)	Bear 1 and bear 5
12	(5, 28, 16)	Output shaft 1
13	(32, 42, 46)	Lock 1, lubricating ring 2, and eccentric sleeve 1
14	(43, 11, 34)	Input shaft, middle box, and lock 2
15	(49, 1, 37)	Upper box, oil collecting hood, and end cap 3
16	(35, 12, 48)	End cap 1, end cap 2, and blank cap 1

Table 4: Parallel assembly sequence of 4-DPA.

Subassembly index	Subassembly	Targeted parts
1	(27, 17)	Lubricating ring 4 and bear 9
2	(26, 15, 20, 18)	Blank cap 2, bear 10, spline, and bear 12
3	(19, 36, 24)	Inner splined sleeve, bear 8, and eccentric sleeve 4
4	(38, 45, 44, 31)	Bear 7, bear 11, eccentric sleeve 5, and bear 6
5	(23, 29, 4, 22)	Gear 7, sleeve 3, lubricating ring 1, and gear 6
6	(30, 21, 14)	Gear 2, gear 3, and gear 5
7	(25, 13)	Gear 4 and end cap 4
8	(28, 16)	Output shaft 2 and intermediate shaft
9	(7, 3, 47, 40)	Gear 1, bear 4, bear 2, and bear 5
10	(46, 2, 39, 41)	Eccentric sleeve 1, sleeve 3, bear 1, and lubricating ring 3
11	(9, 42, 32, 11)	Bear 3, lubricating ring 2, lock 1, and middle box
12	(8, 33, 43)	Eccentric sleeve 2, lock 3, and input shaft
13	(34, 6, 10)	Lock 2, sleeve 1, and sleeve 2
14	(49, 37, 5)	Upper box, end cap 3, and output shaft 1
15	(35, 1, 48, 12)	End cap 1, oil collecting hood, blank cap 1, and end cap 2

5. Conclusions

A novel PASP optimization method is developed for complex product in this paper. A SAPN model is proposed to describe the precedence relationships in the assembly process, while huge matrix calculation is avoided and possibility of human error is reduced. To optimize PASP problem, a PSOBC method is developed along with the SAPN model. Verifications show that assembly time of the optima calculated using the proposed method is 9.0%, 4.2%, and 3.1% better than the standard PSO algorithm when the DPA is 2, 3, and 4, respectively. Assembly time of the three optimal sequences is 74.8 seconds, 59.6 seconds, and 57.3 seconds, separately. Parallel assembly process can be optimally planned by using the proposed method.

Nomenclature

Sets

P: Finite set of places
T: Finite set of transitions
F: Finite set of arcs
W: Finite set of weights
M: Finite set of markings
pt: Finite set of preceding transitions
PT: Finite set of pts
TM: Finite set of operation times

Indices

i, i_1, i_2: Indices of transitions
j: Index of places
k: Index of arcs
g: Index of iterations
s: Index of subassemblies
d, d_1, d_2: Indices of particles

Symbols

p: Place
t: Transition
f: Arc
w: Weight
m: Marking
tm: Operation time of each transition

O:	Operator
P_l:	Population
$\mathbf{B}_{id}$, $\mathbf{B}_{gd}$:	Current local/global best positions
$\mathbf{W}_{id}$, $\mathbf{W}_{gd}$:	Current local/global worst positions

Parameters

δ_p:	Penalty factor for precedence relationships violation
δ_l:	Penalty factor for parallel principles violation
d_h:	Upper bound of diversity function
d_l:	Lower bound of diversity function
itn:	Current number of iterations
$itn_{\max}$:	Max number of iterations
L:	Diagonal length of searching area
ω:	Inertia factor
c_l:	Local learning factor
c_g:	Global learning factor
r:	Random number with uniform distribution on (0, 1)
DPA:	Degree of parallel assembly

Variables

$time$:	Operation time of assembly/subassembly
$\mathbf{X}$:	Vector of particle position
$\mathbf{V}$:	Vector of particle velocity
$FT\ (\mathbf{X})$:	Fitness function
$F_p\ (s)$:	Penalty function for violating precedence relationships
$F_l(s)$:	Penalty function for violating parallel principles
$F_d\ (P_l)$:	Diversity function

Abbreviations

SAPN:	Synchronized assembly Petri net
ASP:	Assembly sequence planning
PASP:	Parallel assembly sequence planning
SASP:	Sequential assembly sequence planning
PSO:	Particle swarm optimization
PSOBC:	Particle swarm optimization based on bacterial chemotaxis.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the Zhejiang Key Research and Development Program under Grant 2017C01008, the Analytical and Testing Technology Project of Zhejiang Province under Grant 2018C37068, the Key Technology Research Project of Wenzhou under Grants 2018ZG020 and ZG2017002, and the Open Research Fund of the Low-Voltage Apparatus Technology Research Center of Zhejiang under Grant 201711-01.

References

[1] B. Deepak, M. R. G. Bala Murali, and B. Biswal, "Assembly sequence planning using soft computing methods: a review," *Proceedings of the Institution of Mechanical Engineers, Part E: Journal of Process Mechanical Engineering*, vol. 233, no. 3, pp. 653–683, 2019.

[2] M. R. Bahubalendruni and B. B. Biswal, "A review on assembly sequence generation and its automation," *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, vol. 230, no. 5, pp. 824–838, 2016.

[3] L. S. De Mello and A. C. Sanderson, "AND/OR graph representation of assembly plans," *IEEE Transactions on Robotics and Automation*, vol. 6, no. 2, pp. 188–199, 1990.

[4] Y. Yang, P. Yang, J. Li et al., "Research on virtual haptic disassembly platform considering disassembly process," *Neurocomputing*, vol. 348, pp. 74–81, 2019.

[5] Y.-j. Wu, Y. Cao, and Q.-f. Wang, "Assembly sequence planning method based on particle swarm algorithm," *Cluster Computing*, vol. 22, no. S1, pp. 835–846, 2019.

[6] Y. Wang and D. Tian, "A weighted assembly precedence graph for assembly sequence planning," *The International Journal of Advanced Manufacturing Technology*, vol. 83, no. 1–4, pp. 99–115, 2016.

[7] A. Bedeoui, R. B. Hadj, M. Hammadi et al., "Assembly sequence plan generation of heavy machines based on the stability criterion," *The International Journal of Advanced Manufacturing Technology*, 2019.

[8] W. Pan, Y. Wang, and X.-D. Chen, "Domain knowledge based non-linear assembly sequence planning for furniture products," *Journal of Manufacturing Systems*, vol. 49, pp. 226–244, 2018.

[9] M. Bahubalendruni, A. Gulivindala, S. Varupala et al., "Optimal Assembly Sequence generation through computational approach," *Sādhanā*, vol. 44, no. 8, p. 174, 2019.

[10] M. V. A. R. Bahubalendruni, A. Gulivindala, M. Kumar, B. B. Biswal, and L. N. Annepu, "A hybrid conjugated method for assembly sequence generation and explode view generation," *Assembly Automation*, vol. 39, no. 1, pp. 211–225, 2019.

[11] M. R. Bahubalendruni and B. B. Biswal, "An intelligent approach towards optimal assembly sequence generation," *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, vol. 232, no. 4, pp. 531–541, 2018.

[12] L. Zhang, H. Lv, D. Tan et al., "Adaptive quantum genetic algorithm for task sequence planning of complex assembly systems," *Electronics Letters*, vol. 54, no. 14, pp. 870–872, 2018.

[13] H. S. Wang, C. H. Tu, and K. H. Chen, "Supplier selection and production planning by using guided genetic algorithm and dynamic nondominated sorting genetic algorithm II approaches," *Mathematical Problems in Engineering*, vol. 2015, pp. 1–15, 2015.

[14] M. Zhang, L. Wang, Z. Cui et al., "Fast nondominated sorting genetic algorithm II with lévy distribution for network topology optimization," *Mathematical Problems in Engineering*, vol. 2020, Article ID 3094941, 12 pages, 2020.

[15] X. Hu, Z. Xu, L. Yang et al., "A novel assembly Line Scheduling Algorithm based on CE-PSO," *Mathematical Problems in Engineering*, vol. 2015, Article ID 685824, 9 pages, 2015.

[16] R. Ab, F. Mohd, A. Tiwari et al., "Integrated optimization of mixed-model assembly sequence planning and line balancing using Multi-Objective Discrete Particle Swarm

Optimization," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing: AIEDAM*, vol. 33, no. 3, pp. 332–345, 2019.

[17] Y. Zhao, W. Li, X. Wang et al., "Path planning of slab library crane based on improved ant colony algorithm," *Mathematical Problems in Engineering*, vol. 2019, Article ID 7621464, 16 pages, 2019.

[18] J. Huo, Z. Wang, F. Chan et al., "Assembly line balancing based on beam ant colony optimisation," *Mathematical Problems in Engineering*, vol. 2018, Article ID 2481435, 17 pages, 2018.

[19] J.-F. Tsai, J. G. Carlsson, D. Ge, Y.-C. Hu, and J. Shi, "Improved quantum-inspired evolutionary algorithm for engineering design optimization," *Mathematical Problems in Engineering*, vol. 2012, pp. 1–7, 2012.

[20] W. Hongbin, D. Jian, and W. Yueling, "High-order feedback iterative learning control algorithm with forgetting factor," *Mathematical Problems in Engineering*, vol. 2015, Article ID 826409, 7 pages, 2015.

[21] M. F. F. Ab Rashid, W. Hutabarat, and A. Tiwari, "Multi-objective discrete particle swarm optimisation algorithm for integrated assembly sequence planning and assembly line balancing," *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, vol. 232, no. 8, pp. 1444–1459, 2018.

[22] L. Shu, M. Dapino, G. Wu, and D. Chen, "Frequency-dependent sliding-mode control of Galfenol-driven unimorph actuator based on finite-element model," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 2, pp. 1071–1082, 2016.

[23] Y. Wang and J. H. Liu, "Chaotic particle swarm optimization for assembly sequence planning," *Robotics and Computer-Integrated Manufacturing*, vol. 26, no. 2, pp. 212–222, 2010.

[24] B. Niu, Y. L. Zhu, X. X. He et al., "An improved particle swarm optimization based on bacterial chemotaxis," *Proceedings of the World Congress on Intelligent Control and Automation (WCICA)*, vol. 1, pp. 3193–3197, 2006.

[25] Y. Cheng and Y. Ren, "Heuristic search solvers for differential game model of CO_2 emission in electricity market," *Journal of Computational Information Systems*, vol. 8, no. 19, pp. 8151–8158, 2012.

[26] X. Y. Zhang, C. Li, and Z. Li, "Optimal reactive power dispatch based on mixed bacterial chemotaxis algorithm," *Applied Mechanics and Materials*, vol. 494-495, pp. 1849–1852, 2014.

Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 6765274, 19 pages
https://doi.org/10.1155/2020/6765274

Research Article

Image Processing-Based Pitting Corrosion Detection Using Metaheuristic Optimized Multilevel Image Thresholding and Machine-Learning Approaches

Nhat-Duc Hoang [1,2]

[1] *Institute of Research and Development, Duy Tan University, Da Nang 550000, Vietnam*
[2] *Faculty of Civil Engineering, Duy Tan University, Da Nang 550000, Vietnam*

Correspondence should be addressed to Nhat-Duc Hoang; hoangnhatduc@dtu.edu.vn

Received 30 December 2019; Revised 24 February 2020; Accepted 13 April 2020; Published 5 May 2020

Guest Editor: Dr. Dilbag Singh

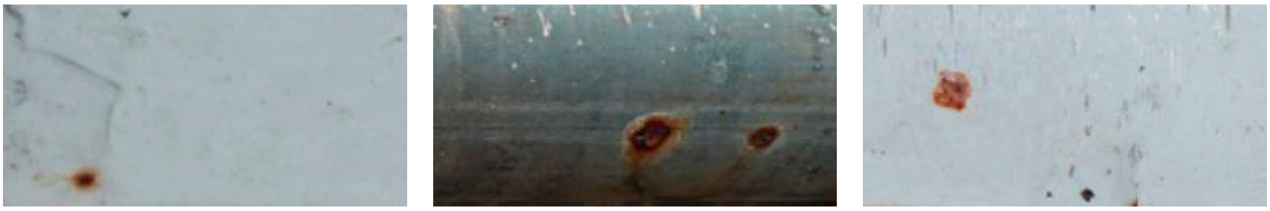
Pitting corrosion can lead to critical failures of infrastructure elements. Therefore, accurate detection of corroded areas is crucial during the phase of structural health monitoring. This study aims at developing a computer vision and data-driven method for automatic detection of pitting corrosion. The proposed method is an integration of the history-based adaptive differential evolution with linear population size reduction (LSHADE), image processing techniques, and the support vector machine (SVM). The implementation of the LSHADE metaheuristic in this research is multifold. This optimization algorithm is employed in the task of multilevel image thresholding to extract regions of interest from the metal surface. Image texture analysis methods of statistical measurements of color channels, gray-level co-occurrence matrix, and local binary pattern are used to compute numerical features subsequently employed by the SVM-based pattern recognition phase. In addition, the LSHADE metaheuristic is also used to optimize the hyperparameters of the machine-learning approach. Experimental results supported by statistical test points out that the newly developed approach can attain a good predictive result with classification accurate rate = 91.80%, precision = 0.91, recall = 0.94, negative predictive value = 0.93, and F1 score = 0.92. Thus, the newly developed method can be a promising tool to be used in a periodic structural health survey.

1. Introduction

Currently, ageing infrastructure elements with constraints on maintenance budgets are the main concern of infrastructure management agencies around the world. These facts urge the implementation of smart and cost-effective methods in the field of structure health monitoring [1–7]. Typical objectives of structure health monitoring include the correct recognition of the presence, the location, and the type of the structural defects. These pieces of information can be then used by various assessment models to support decisions on rehabilitation options to maximize the service life of infrastructure elements [8, 9].

For metal infrastructure elements, corrosion negatively affects their durability and operability. It is reported that corrosion is a dominant form of defects with 42% of frequency of failure mechanisms in engineering structures [10]. Therefore, recognition as well as diagnostics of corroded areas is an important task in periodic structural heath surveys [11, 12]. The surveys' outcome significantly helps owners or maintenance agencies to judge the effectiveness of the currently employed protection methods and to prioritize rectifying measures [13, 14].

Particularly, pitting corrosion, recognized by isolated corroded damage units within the metal surface, is a severe type of structural defect [3, 13]. This defect appears on the surface of various civil engineering structures including bridges, high-rise buildings, pipelines, and storage tanks (see Figure 1). Pitting corrosion is generally more harmful than uniform corrosion since it is hard to detect and predict, as well as design against its damages [15]. This localized corrosion damages may have diverse shapes (often hemispherical or



(a) (b) (c)

Figure 1: Pitting corrosion found in (a) bridge structure, (b) pipeline, and (c) cladding structure.

cup-shaped). Moreover, pits can be covered by a membrane of corrosion products.

If left undetected, pitting corrosion can lead to catastrophic failure of civil engineering structures. It is because this kind of damage only causes small loss of material with insignificant effect on its surface, while it can bring about extremely dangerous damages to deep areas of structures below the tiny pits. One example of the devastating effect of pitting corrosion is the explosion caused by gasoline fumes leaking from a steel gasoline pipe in Guadalajara (Mexico) on April 22, 1992; this accident caused the deaths of 12 people [10, 16]. Pitting corrosion is also the cause of the deadly collapse of the U.S. Highway 35 bridge in 1967 within which 46 people died [17]. On 04/22/1992, a gas explosion in Guadalajara (Spain) is believed to be brought about by pitting corrosion appeared on gas pipes [10]; this event caused the deaths of 252 people and injuries of about 1,500 people [10].

In Vietnam, as well as in many other nations, surveying infrastructure elements is usually performed manually by human inspectors. Although the manual method can help to attain accurate detection results, its notable downsides are low productivity and effects of subjective criteria [14]. These are due to the time-consuming processes of measurement and data report as well as inconsistent experience/assessment of surveyors. Moreover, the sheer number of existing structure elements creates a significant challenge to human inspectors to perform surveying works frequently and detect structural damage timely. Therefore, automatic approach for pitting corrosion detection is an urgent need of infrastructure management agencies.

Accordingly, this study aims at constructing an intelligent method for automatic recognition of metallic surface area subject to pitting corrosion. The newly constructed model is an integration of metaheuristic, image processing techniques, and machine learning. The History-Based Adaptive Differential Evolution with Linear Population Size Reduction (LSHADE) metaheuristic approach is employed for multileveling image thresholding as well as optimizing the performance of the Support Vector Machine- (SVM-) based data classifier. The LSHADE optimized multileveling image thresholding supported by the connected component labeling algorithm is used to extract the region of interest (ROI) from image samples. Based on the extracted ROI, texture information including statistical measurements of color channels, gray-level co-occurrence matrix, and local binary pattern is used to characterize properties of the metal surface. The SVM relies on the texture information to separate input samples into two categories of nonpitting corrosion (the negative class) and pitting corrosion (the positive class). A dataset consisting of 213 image samples has been collected to train and verify the proposed integrated model.

Furthermore, the model training phase of the SVM necessitates an appropriate setting of its hyperparameters including the penalty coefficient and the kernel function parameter. The task of determining such hyperparameters is generally regarded as the model selection problem [18]. This is a crucial task in machine learning because hyperparameters strongly influence the generalization capability of prediction models [19–21]. A poor model selection may either result in an overfitted or underfitted model. The model selection can be a challenging task because hyperparameters are often searched in continuous domains [22–25].

Accordingly, there are infinite possible solutions to the model selection problem and an exhaustive search to determine the optimal set of hyperparameters is infeasible. Therefore, metaheuristic can be utilized to tackle the model selection task [26]. Metaheuristic is regarded as a high-level heuristic designed to seek for a sufficiently good solution to a global optimization problem. This advanced method generally does not require assumptions regarding the optimization task being solved and can be applied to a wide range of problems [27–31]. Metaheuristics have been demonstrated to be capable optimization methods which can help to identify solutions with quality superior to conventional methods (e.g., iterative algorithms and gradient descent-based algorithms) [32–34]. With this motivation, this study employs the LSHADE algorithm [35], a state-of-the-art metaheuristic to optimize the SVM model performance. The LSHADE is selected in this study due to its outstanding performance reported in recent works [36–38].

In summary, the main contributions of the current study can be stated as follows. (1) A novel integrated framework based on image processing techniques, metaheuristic optimization, and machine-learning prediction for pitting corrosion is proposed. (2) An autonomous model operation is achieved by means of the LSHADE metaheuristic which minimizes human's efforts for model construction and parameter tuning. (3) An integrated approach of the LSHADE and SVM algorithms is proposed for achieving better corrosion detection accuracy compared to benchmark machine-learning models.

The rest of the paper is organized as follows: Section 2 reviews the research method; Section 4 is dedicated to describing the newly developed model which employs the aforementioned methods of the LSHADE, multilevel image thresholding, image texture computation, and SVM; Section 5 reports the experimental results, followed by Section 6 which summarizes this study with several concluding remarks.

2. Related Works

Among various methods of automatic structural health monitoring, image processing-based visual inspection is commonly employed due to the available of low-cost digital cameras and a fast advance of computer vision techniques. Choi and Kim [39] relies on attributes (color, texture, and shape) extracted from digital images. In addition, data dimensionality reduction and linear classifiers are employed to identify corrode areas from metal surface [39]. However, due to the complexity of the problem of interest, the accuracy of the aforementioned detection model can be enhanced with nonlinear and more sophisticated data classification approaches. Valor et al. [12] constructed Markov chain models for stochastic modeling of pitting corrosion appearing on metallic structures. Medeiros et al. [40] demonstrated the effectiveness of the gray level co-occurrence matrix and color descriptors in classification of corroded and noncorroded surfaces. High dynamic range imaging is used in [3] for the recognition of pitting corrosion via visual inspection [3].

Chen et al. [41] employed Fourier-transform-based and color image processing methods for recognition of defects on the steel bridge surface. Ji et al. [42] put forward a computer vision-based method for rating of corrosion defects on coated materials using the watershed segmentation method. Shen et al. [43] established an automatic method for detecting steel bridge coating rust defect based on image texture analysis and discrete Fourier transform. Jahanshahi and Masri [44] investigated the influence of color space, color channels, and subimage block size on corrosion detection capability. This model also employed color wavelet-based texture analysis algorithms for recognition of defective areas [44].

A digital image processing method based on the K-means clustering algorithm, the double-center-double-radius algorithm, and the least-square support vector machine has been used by Liao and Lee [45] to detect rust defects on steel bridge coatings. Liao and Cheng [46] relied on the least-square support vector machine, spectral power distribution, spectral reflectance, and matrix restoration to detect discoloration status of a steel bridge coating. A model used for detection and assessment of corrosion on pipelines through image filtering and morphological operations has been constructed by Bondada et al. [13]. Enikeev et al. [14] utilized the linear support vector machine method for recognizing pitting corrosion on aluminum surfaces. Zhao et al. [47] employed deep-learning and artificial bee colony algorithm to identify material corrosion. Zhang et al. [48] put forward a segmentation process for detecting localized corrosion on the rust-removed metallic surface using deep-learning technique. Ivasenko and Chervatyuk [49] propose an image processing-based method for segmentation of rusted areas on painted construction surfaces. Based on a recent review work carried out by Ahuja and Shukla [50], there is an increasing trend of applying image processing techniques for automatic corrosion detection; this work also points out a great potentiality of machine learning for corrosion recognition performed on large area. Following this trend of research, the current study proposes a novel approach for automatic detection of pitting corrosion. The proposed method is an integration of image processing techniques, metaheuristic optimization, and machine-learning-based pattern recognition. The individual components needed to establish the newly developed model for pitting corrosion detection are presented in the subsequent section of the article.

3. Research Method

3.1. Multilevel Image Thresholding. In the field of image processing, multilevel thresholding of a gray image is a widely utilized method for coping with a variety of computer vision tasks including feature extraction, object detection, and image analysis [51–55]. Particularly for multilevel thresholding, the image histogram is first computed. Based on this histogram, the gray levels of an image are categorized into multiple clusters based on a set of thresholds. To determine such thresholds appropriately, the Otsu criterion, formulated by Otsu [56], that relies on the maximization of between-class variance approach can be used due to its ease of implementation simplicity and good image segmentation performance [57–59]. Since the computation of the threshold values based on the Otsu criterion is a computationally expensive image processing operation, metaheuristic approaches are often employed [60, 61].

Given a digital image in L gray levels 0, 1, . . ., $L-1$, its histogram $H = H = \{f_0, f_1, \ldots, f_{L-1}\}$ can be constructed. Herein, f_i denotes the occurrence frequency of gray level i. Let $N = \sum_{i=0}^{L-1} f_i$ represent the total number of pixels in the image; the i^{th} gray level occurrence probability is computed as follows:

$$p_i = \frac{f_i}{N}. \tag{1}$$

The objective is to segment the image of interest into $K+1$ clusters (C_0, C_1, ..., C_k, . . ., C_K) using K thresholds chosen from the set $T = T = \{t_0, t_1, \ldots, t_k, \ldots, t_K\}$, where $0 \le t_k \le L$. C_k denotes a set of gray levels. For each cluster C_k, the cumulative probability w_k and mean gray level η_k can be obtained via

$$\begin{aligned} w_k &= \sum_{i \in C_k} p_i, \\ \eta_k &= \sum_{i \in C_k} \frac{i \times p_i}{w_k}, \quad \text{where}\, k \in \{0, 1, \ldots, K\}. \end{aligned} \tag{2}$$

The mean intensity η_T of the whole image and the between-class variance σ_B^2 are computed as follows [59]:

$$\begin{aligned}\eta_T &= \sum_{k=0}^{K} w_k \times \eta_k = \sum_{i=0}^{L-1} i \times p_i,\\ \sigma_B^2 &= \sum_{k=0}^{K} w_k \times \left(\eta_k - \eta_T\right)^2 = w_0 \times \left(\eta_0 - \eta_T\right)^2 + w_1\\ &\quad \times \left(\eta_1 - \eta_T\right)^2 + \cdots + w_K \times \left(\eta_K - \eta_T\right)^2.\end{aligned} \tag{3}$$

It is noted that the threshold levels for a given number of clusters are determined based on maximizing the separation between cluster means. Thus, the optimal thresholding values can be attained by maximizing the between-class variances mathematically stated as follows:

$$\left(t_1^{Op}, t_2^{Op}, \ldots, t_K^{Op}\right) = \underset{\left(t_1, t_1, \ldots, t_K\right) \in K}{\arg\max} \left\{\sigma_B^2\left(t_1, t_1, \ldots, t_K\right)\right\}. \tag{4}$$

3.2. Connected Component Labeling (CCL). Connected component labeling (CCL) is an operation on a binary image. This method analyzes the binary-1 pixels and divides the binary image into distinctive component regions [62–64]. The CCL is often followed by further property measurement operations on each region [57, 65]. In essence, the CCL algorithm carries out the unit change from individual pixel to region; all pixels having value binary 1 and are connected to each other are grouped into one cluster [64]. In this study, the iterative algorithm proposed by Haralick [66] is employed for performing CCL. This method is selected due to its simplicity and the fact that it does not require auxiliary storage to yield the labeled image from the binary image. The iterative algorithm includes an initialization step and a sequence of top-down label propagation followed by bottom-up label propagation repeated until no label changes is observed.

3.3. Image Texture Analysis. In the field of computer vision, an image texture is commonly regarded as a set of metrics used to quantify the perceived texture or regional characteristic of an image [64]. This set of metrics provides information regarding the spatial arrangement of color intensity in an image sample. Image textures can express intuitive quantities of the surface image including the degrees of roughness and smoothness. Due to such reasons, image texture analysis can be highly useful for pitting corrosion recognition.

3.3.1. Measurement of Statistical Properties of Color Channels. Since corrosion often results in areas on the metal surface with distorted color, using the statistical properties of image color channels (red, green, and blue) can be helpful for the task of interest. Let I be a variable that denotes the gray levels of an image sample. The first-order histogram $P(I)$ is obtained as follows [67]:

$$P_c(I) = \frac{N_{I,c}}{W \times H}, \tag{5}$$

where c denotes a color channel, $N_{I,c}$ is the number of pixels having intensity value I, and H and W are the height and width of the image sample, respectively.

Accordingly, the mean (μ_c), standard deviation (σ_c), skewness (S_c), kurtosis (K_c), entropy (E_c), and range (R_c) of color channel are given as follows [68]:

$$\begin{aligned}\mu_c &= \sum_{i=0}^{NL-1} I_{i,c} \times P_c(I),\\ \sigma_c &= \sqrt{\sum_{i=0}^{NL-1} \left(I_{i,c} - \mu_c\right)^2 \times P_c(I)},\\ S_c &= \frac{\sum_{i=0}^{NL-1} \left(I_{i,c} - \mu_c\right)^3 \times P_c(I)}{\sigma_c^3},\\ \eta_c &= \frac{\sum_{i=0}^{NL-1} \left(I_{i,c} - \mu_c\right)^4 \times P_c(I)}{\sigma_c^4},\\ K_c &= -\sum_{i=0}^{NL-1} P_c(I) \times \log_2\left(P_c(I)\right),\\ R_c &= \operatorname{Max}\left(I_c\right) - \operatorname{Min}\left(I_c\right),\end{aligned} \tag{6}$$

where since the image samples are 8 bit, $NL = 256$ denotes the number of discrete intensity values.

3.3.2. Gray-Level Co-Occurrence Matrix (GLCM). Properties extracted from the Gray-Level Co-Occurrence Matrix (GLCM) [69, 70] are highly effective for texture discrimination. This method aims at analyzing the repeated occurrence of certain gray-level patterns existing in an image texture. Therefore, it can be used to assess the coarseness of image regions [67, 71]. Let r and θ denote a distance and a rotation relationship between two pixels. The GLCM, denoted as P_δ, represents a probability of the two gray levels of i and j having the relationship specifying by r and θ.

As suggested by Haralick et al. [69], four GLCMs with $r = 1$ and $\theta = 0^\circ$, 45°, 90°, and 135° can be used to characterize an image texture. Accordingly, the measurements of angular second moment (AM), contrast (CO), correlation (CR), and entropy (ET) can be computed and employed for texture discrimination [69, 72]:

$$\begin{aligned}\mathrm{AM} &= \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} P_\delta^N(i, j)^2,\\ \mathrm{CO} &= \sum_{k=0}^{N_g-1} k^2 \underbrace{\sum_{i=1}^{N_g} \sum_{j=1}^{N_g} P_\delta^N(i, j)}_{|i-j|=k},\\ \mathrm{CR} &= \frac{\sum_{i=1}^{N_g} \sum_{j=1}^{N_g} i \times j \times P_\delta^N(i, j) - \mu_X \mu_Y}{\sigma_X \sigma_Y},\\ \mathrm{ET} &= -\sum_{i=1}^{N_g} \sum_{j=1}^{N_g} P_\delta^N(i, j) \log\left(P_\delta^N(i, j)\right),\end{aligned} \tag{7}$$

where N_g is the number of gray level values; μ_X, μ_Y, σ_X, and σ_Y denote the means and standard deviations of the marginal distribution associated with $P_\delta^N(i, j)$ [69].

3.3.3. Local Binary Pattern (LBP). Local Binary Patterns (LBP), proposed by [73, 74], is a nonparametric approach used to summarize local structures of an image sample. This approach essentially compares each pixel with its neighboring ones. The notable advantages of the LBP are its computational efficiency and tolerance of monotonic illumination variations [75]. This image processing technique has been applied successfully for analyzing texture and has also been extended to be applied in various computer vision tasks face image analysis [76–78], image and video retrieval [79], visual inspection [80], remote sensing [81], and human action recognition [82].

The LBP algorithm labels the pixels of an image sample with LBP codes, which expresses the local structure around each pixel of interest. The size of the neighboring pixels is usually 3 × 3. Accordingly, the center pixel is compared with its eight neighbors. The neighboring pixel is coded as 1 if its gray intensity is greater than that of the center pixel. Otherwise, neighboring pixel is coded as 0. Given a center pixel at x_c and y_c, its LBP can be obtained as follows [75]:

$$\mathrm{LBP}(x_c, y_c) = \sum_{p=0}^{p-1} s(i_p - i_c) \times 2^p, \tag{8}$$

where i_c and i_p denote gray intensities of the center pixel and its neighboring pixels. $s(x)$ is 1 if $x \geq 0$ and 0 if $x < 0$.

3.4. Support Vector Machine (SVM) for Pattern Recognition. The Support Vector Machine (SVM), formulated by Vapnik [83], is a powerful method for solving nonlinear and complex pattern recognition tasks. This machine-learning approach relies on the concept of structural risk minimization to construct the decision hyperplane that classifies data into distinctive groups. Therefore, the SVM is less prone to overfitting than other machine-learning methods such as neural networks which are relied on the theory of empirical risk minimization [18, 84]. Notably, to deal with noisy data as well as nonlinear separability problems, the SVM employs the framework of maximum margin construction and kernel tricks [22, 85–87]. In addition, the learning phase of a SVM model used for data classification is formulated as to solving a convex optimization problem (i.e., a quadratic programming problem); this fact ensures a global convergence of the model training phase [18].

Given a training dataset $\{x_k, y_k\}_{k=1}^N$ with a vector $x_k \in R^n$ representing image texture information and a scalar $y_k \in \{-1, +1\}$, denoting the class labels (either nonpitting corrosion or pitting corrosion), a SVM model constructs a decision boundary expressed in the form of a hyperplane to separate the data samples into two categories. To do so, it is required to solving a nonlinear programming problem described as follows [88–90]:

$$\begin{aligned} &\text{Maximize} \quad J_p(w, e) = \frac{1}{2} w^T w + c \frac{1}{2} \sum_{k=1}^N e_k^2 \\ &\text{s.t.} \quad y_k (w^T \phi(x_k) + b) \geq 1 - e_k, \quad k = 1, \ldots, N, e_k \geq 0, \end{aligned} \tag{9}$$

where $w \in R^n$ denotes a normal vector to the classification hyperplane and $b \in R$ is the model bias; $e_k > 0$ represents slack variables; c denotes a penalty constant; and $\phi(x)$ is the employed nonlinear data mapping function.

Notably, an explicit expression of the data mapping function $\phi(x)$ is not required. The model training and prediction phases only necessitate the product of $\phi(x)$ in the input space which is called a Kernel function:

$$K(x_k, x_l) = \phi(x_k)^T \phi(x_l). \tag{10}$$

The Radial Basis Function Kernel (RBFK) is often used for nonlinear data classification:

$$K(x_k, x_l) = \exp\left(-\frac{\|x_k - x_l\|^2}{2\sigma^2}\right), \tag{11}$$

where σ denotes a model hyperparameter which needs to be specified by the user.

Accordingly, the final SVM model used for pitting corrosion is given by

$$y(x_l) = \mathrm{sign}\left(\sum_{k=1}^{SV} \alpha_k y_k K(x_k, x_l) + b\right), \tag{12}$$

where α_k represents the solution of the dual form of the abovementioned nonlinear programming problem and *SV* denotes the number of support vectors found by the model training phase.

3.5. The History-Based Adaptive Differential Evolution with Linear Population Size Reduction (LSHADE). The History-Based Adaptive Differential Evolution with Linear Population Size Reduction (LSHADE), proposed in [35, 91], is a popular variant of the standard Differential Evolution (DE) [92]. The LSHADE inherits the integrated mutation-crossover operation of the DE used for exploring and exploiting the search space. Tanabe and Fukunaga [91] also enhances the searching capability of the standard optimization algorithm by a novel adaptive strategy used for fine-tuning the mutation scale (*F*) and the crossover probability (*CR*) coefficients which are the two crucial hyperparameters of the DE. This adaptive strategy is based on a record of successful population members. In addition, to improve the mutation operator, a method called DE/current-to-pbest/1 is implemented. Finally, for meliorating enhance convergence rate of the LSHADE, a population size shrinking schedule is implemented. Due to such advantages, superior performance of the LSHADE algorithm has been reported in various comparative studies [36, 93–95].

The overall structure of the LSHADE metaheuristic method is illustrated in Figure 2. Given the searched domain

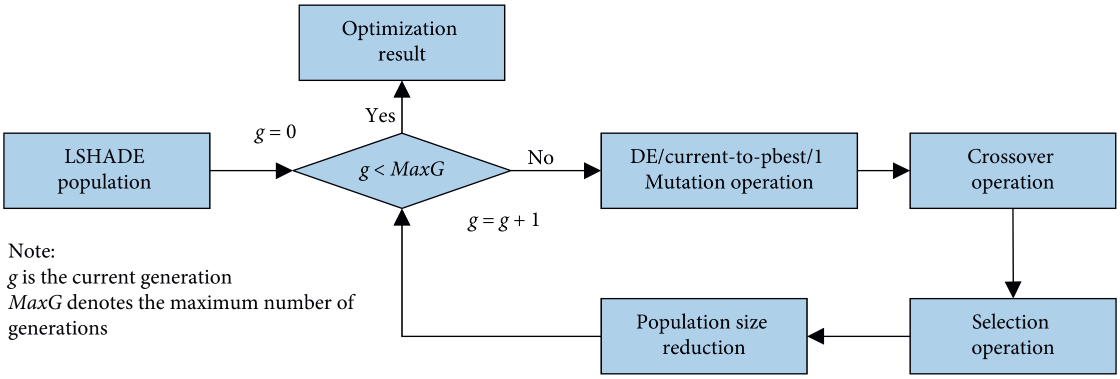

Optimization result
Yes
g = 0
LSHADE population
g < MaxG
No
DE/current-to-pbest/1 Mutation operation
Crossover operation
g = g + 1
Note:
g is the current generation
MaxG denotes the maximum number of generations
Population size reduction
Selection operation

Figure 2: The operational flow of the LSHADE algorithm.

(lower and upper boundaries), the number of decision variable, and the population size (*PS*), a population of *PS* members is randomly generated within the searched domain. The DE/current-to-pbest/1 and the crossover operations are used to generate a new candidate solution via the creations of mutated solution ($v_{i,g+1}$) and trial solution ($u_{\mathrm{i,g+1}}$) [35, 91]:

$$\begin{gathered} v_{i,g+1} = x_{i,g} + F_i\left(x_{r1,g} - x_{r2,g}\right) + F_i\left(x_{\mathrm{pbest},g} - x_{i,g}\right), \\ u_{j,i,g+1} = \begin{cases} v_{j,i,g+1}, & \text{if } \mathrm{rand}_j \le Cr \text{ or } j = rnb(i), \\ x_{j,i,g}, & \text{if } \mathrm{rand}_j > Cr \text{ and } j \ne rnb(i), \end{cases} \end{gathered} \tag{13}$$

where $x_{\mathrm{pbset,g}}$ denotes the best found solution at g^{th} generation and rand represents a uniform random number ranging between 0 and 1.

Based on the fitness value of the trial solution and its corresponding parent, a selection operation is performed to preserve better solution and cast out the worse. Moreover, the LSHADE relies on two archives of MF and MCR which record the mean values of the mutation scale and the crossover probability. To update these mean values, the two sets of SF and SCR storing all *CR* and *F* values of successful child solutions are used. Furthermore, at the end of each generation, the population size gets shrunk by casting out inferior population members to enhance the algorithm convergence speed [35, 96].

4. The Proposed Metaheuristic Optimized Image Processing and Machine-Learning Method for Automatic Pitting Corrosion Detection

This section of the article aims at describing the structure of the proposed metaheuristic optimized image processing and machine-learning method for automatic recognition of pitting corrosion. The overall structure of the proposed model is presented in Figure 3 which can be divided into several steps.

4.1. Image Sample Preparation. In the first step, to construct the SVM machine-learning model used for pitting corrosion recognition, the set of image samples capturing the texture of metal structures must be prepared. This image set includes samples which contain pitting corrosion and samples without such defect. Based on the collected image samples, regions of interest are extracted within which areas on metal surfaces having distorted color are identified. During field trips in Danang city (Vietnam), 120 image samples containing pitting corrosion (the positive class) have been attained and labeled by human inspectors. It is noted that one image samples may have multiple areas of pitting corrosion. Accordingly, the number of extracted regions of interest that belongs to the positive class is 124. To guarantee a balanced dataset, the number of the negative (without pitting corrosion) data samples should also be 124. Therefore, a set of 93 image samples with no pitting corrosion areas has been included in this study. These 93 image samples have been used to extract 124 regions of interest belonging to the negative class. To train and test the proposed machine-learning model used for pitting corrosion detection, the collected dataset has been divided into two subsets of training (90%) and testing (10%) datasets. The training set is utilized for model construction and testing set is reserved for verifying the model predictive capability.

It is proper to note that ground truth labels of data samples have been determined by human inspectors. Herein, the label = −1 means the negative class and the label = 1 denotes a positive class. The collected images in this study have been taken by the Cannon EOS M10 (CMOS 18.0 MP) and Nokia 7.2 (48 MP main sensor). To enhance the speed of the texture computation phase and to ensure the consistency of an image region, the image size has been set to be 64 × 64 pixels. The image samples are illustrated in Figure 4. Additionally, to better cope with the diversity of the metal surface, the negative class of nonpitting corrosion deliberately includes samples of intact surfaces and stains.

4.2. Extraction of Region of Interest (ROI). Since an area subject to pitting corrosion can have a diverse form of shape, it is necessary to automatically identify the ROI that covers this area. Image texture of this ROI can be subsequently computed and used for pitting corrosion detection. This study proposes a novel integration of metaheuristic optimized multilevel image thresholding, morphological

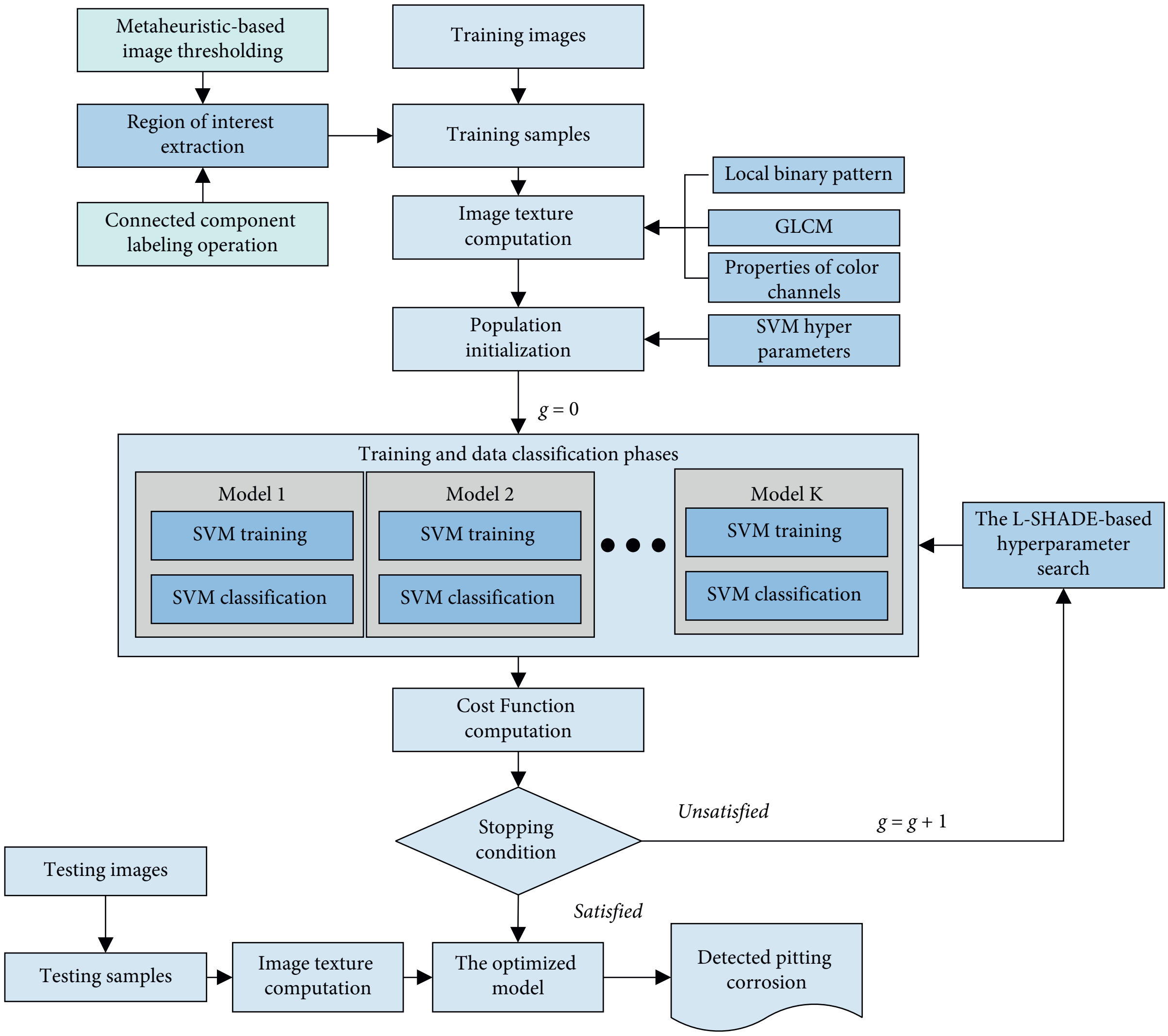
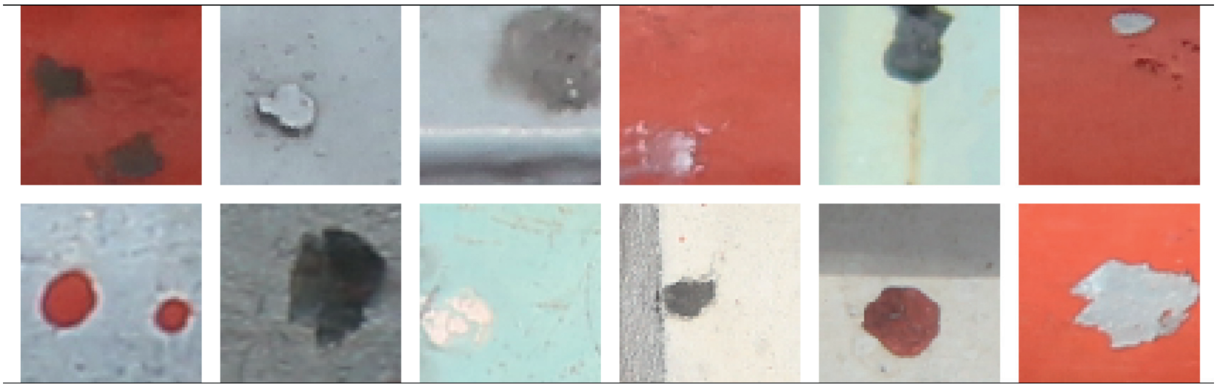
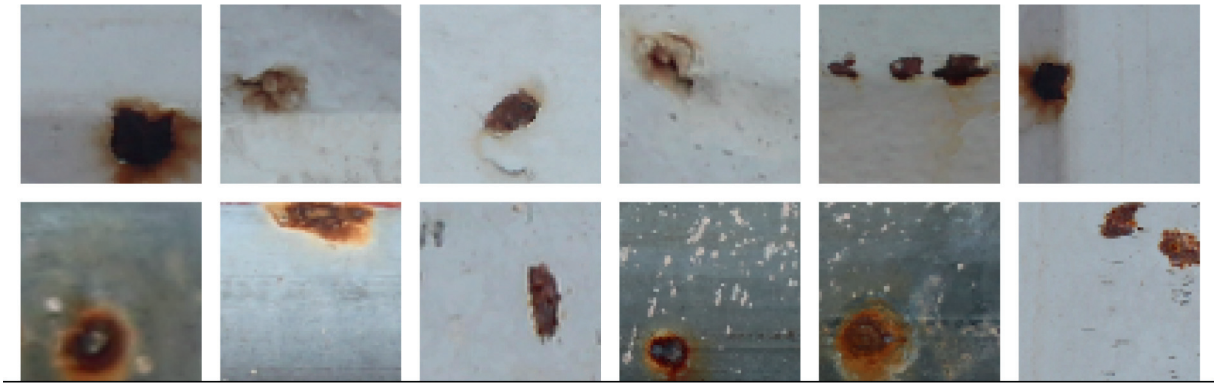

Metaheuristic-based image thresholding
Region of interest extraction
Connected component labeling operation
Training images
Training samples
Image texture computation
Local binary pattern
GLCM
Properties of color channels
Population initialization
SVM hyper parameters
$g = 0$
Training and data classification phases
Model 1
SVM training
SVM classification
Model 2
SVM training
SVM classification
Model K
SVM training
SVM classification
The L-SHADE-based hyperparameter search
Cost Function computation
Stopping condition
Unsatisfied
$g = g + 1$
Satisfied
Testing images
Testing samples
Image texture computation
The optimized model
Detected pitting corrosion

Figure 3: The LSHADE optimized image processing and machine learning for pitting corrosion detection.



(a)

Figure 4: Continued.



(b)

Figure 4: The collected image samples: (a) nonpitting corrosion and (b) pitting corrosion.

operation, and other image processing techniques to determine the ROI from images of the metal surface.

The whole process of ROI extraction is presented in Algorithm 1. The first step is to smooth the image sample by removing noise; a median filter with window size of 7 × 7 is implemented. The color image is then converted to its corresponding gray-scale one. Based on this gray-scale image, the multilevel image thresholding with the aforementioned Otsu objective function is carried out. Based on several trial-and-error experiments with the collected images, the appropriate number of pixel groups has been found to be three. The LSHADE metaheuristic is then employed to determine the most desired thresholds used to separate image pixels into distinctive groups. The metaheuristic based multilevel image thresholding phase is illustrated in Figure 5. The population size of the LSHADE and the maximum number of evolutionary generations are experimentally set to be 20 and is 100, respectively.

Based on the thresholded image, morphological operations (filling and removing small objects) are performed. In addition, the operation of background removal is carried out. Herein, an object is defined as background if its width or height is equal to that of the whole image sample. Based on the binary image representing each pixel group, the CCL is used to identify separated pitting corrosion areas. Subsequently, image convolution and image cropping operations are used to extract ROI. This whole process of ROI extraction is displayed in Figures 6 and 7.

4.3. Image Texture Computation. As mentioned earlier, image texture is used in this study to characterize the feature of the metal surface. The statistical measurements of color channels, GLCM, and LBP descriptors are computed based on each extracted ROI. For each image sample, one color channel yields six statistical measurements of mean, standard deviation, skewness, kurtosis, entropy, and range. Therefore, the number of statistical measurements of color channels is $6 \times 3 = 18$. In addition, the four co-occurrence matrices corresponding to the directions of 0^o, 45^o, 90^o, and 135^o are computed and each of which yields the four indices of the angular second moment, contrast, correlation, and entropy are acquired from one co-occurrence matrix. Hence, the number of GLCM-based texture descriptors is $4 \times 4 = 16$. Finally, 59 features which are the histogram of the LBP after removing nonuniform patterns [73] are included in the feature set. Accordingly, the total number of texture descriptors is $18 + 16 + 59 = 93$.

4.4. The Model Hyperparameter Optimization and the SVM Classification (SVC) for Pitting Corrosion Detection. Based on the above stated ROI extraction and image texture computation, a dataset consisting of 248 data samples and 93 input features can be used for the subsequent model hyperparameter optimization and the SVM classification (SVC) for pitting corrosion detection. It is noted that the created dataset has two class outputs: −1 for nonpitting corrosion (negative class) and +1 for pitting corrosion (positive class). Furthermore, for standardizing the data ranges, this dataset has been preprocessed by the Z-score data normalization described as the following equation:

$$X_{ZN} = \frac{X_o - m_X}{s_X}, \tag{14}$$

where X_o and X_{ZN} denote an original and a normalized feature, respectively, and m_X and s_X represent the mean and the standard deviation of the original feature, respectively.

After the dataset has been standardized, the principal component analysis is then applied to for dimensionality reduction. The three values of total variance explained of 90%, 95%, and 99% are used to select appropriate number of input features used for the SVC-based pitting corrosion recognition. In addition, the LSHADE metaheuristic approach is utilized to search for the most desired values of the SVM model hyperparameters including the penalty coefficient (c) and the kernel function parameter (σ). The LSHADE randomly generates an initial population of the SVM model hyperparameters. In each evolutionary generation, this metaheuristic method gradually explores and exploits the search space for better solution candidates capable of yielding high quality pitting corrosion detection models. Notably, the feasible domains of the penalty coefficient (c) and the kernel function parameter (σ) are [1, 1000] and [0.1, 1000], respectively. In addition, the population size

```
Specify an input image sample ($I$) and the number of thresholds ($K$)
Apply median filter to $I$ to obtain $I_{MF}$
Convert $I_{MF}$ into a gray-scale image $I_G$
Use LSHADE to identify the set of optimal thresholds $(t_1^{Op}, t_2^{Op}, \ldots, t_K^{Op})$
Perform image multilevel thresholding on $I_G$
For each image segment $k$
    Perform image binarization to obtain $I_{Bin,k}$
    Perform morphological operations on $I_{Bin,k}$ (filling and removing noise)
    Remove background
    Perform CCL operation on $I_{Bin,k}$ to obtain a list of objects
    For each object $j$ within $I_{Bin,k}$
        Construct its binary image $I_{BIN}^{j}$
        Perform image convolution:
        $I_{CV} = I * I_{BIN}^{j}$
        Identify the enclosing rectangle $I_{REC}$
        Perform image cropping operation to obtain ROI
    End For
End For
Return ROIs
```

Algorithm 1: The process of extracting ROIs.

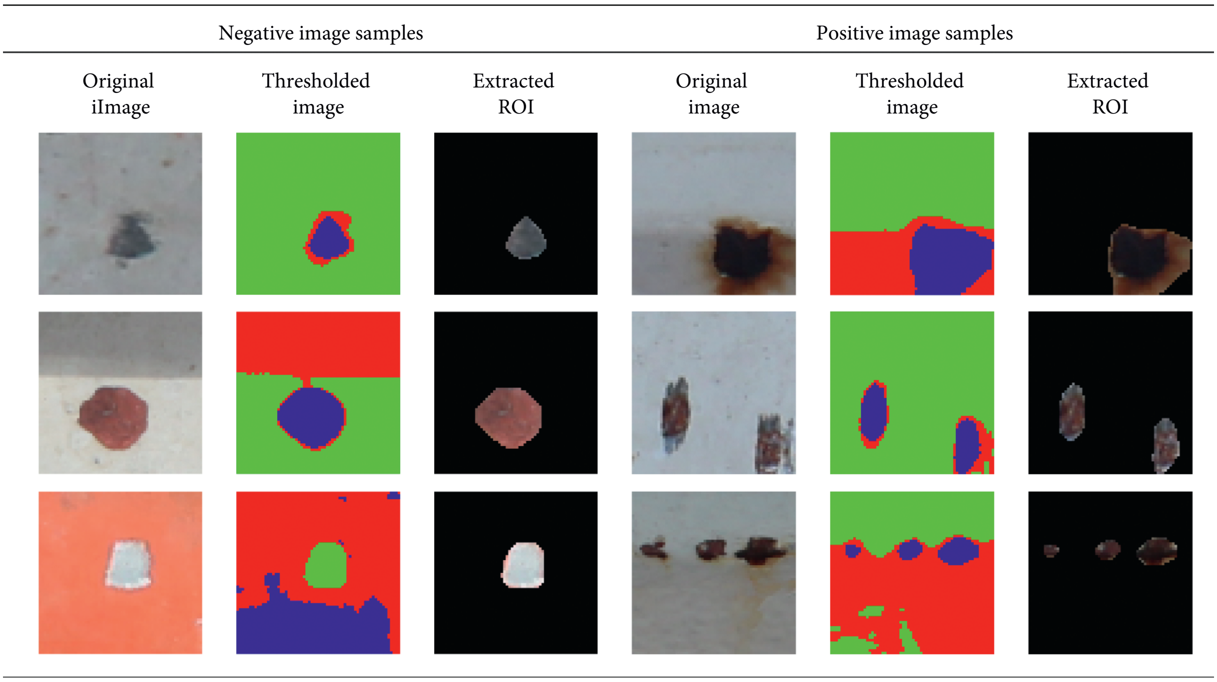


Figure 5: Illustration of the ROI extractions.

and the number of the LSHADE searching generations are selected to be 20 and 100, respectively.

In order to optimize the SVM performance, a K-fold crossvalidation (with $K = 5$) is employed in this study. The average predictive performance obtained from this cross-validation is employed to quantify the model predictive capability. Accordingly, the following cost function (CF) is used by the proposed integration of LSHADE and SVM used for pitting corrosion detection:

$$CF = \frac{\sum_{k=1}^{K} (FNR_k + FPR_k)}{K}, \qquad (15)$$

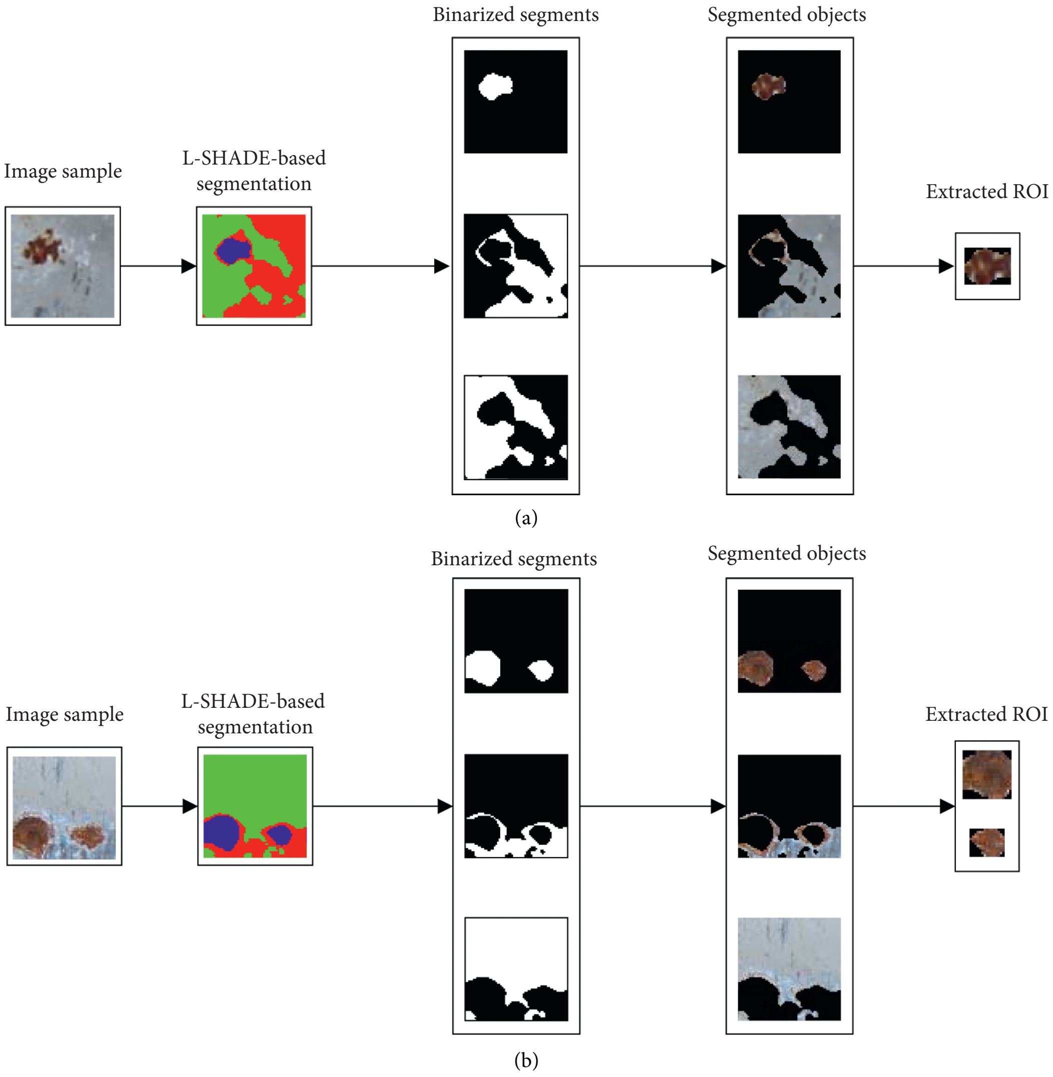


Figure 6: Extraction of region of interest (ROI) from image containing pitting corrosion: (a) one defective object and (b) multiple defective objects.

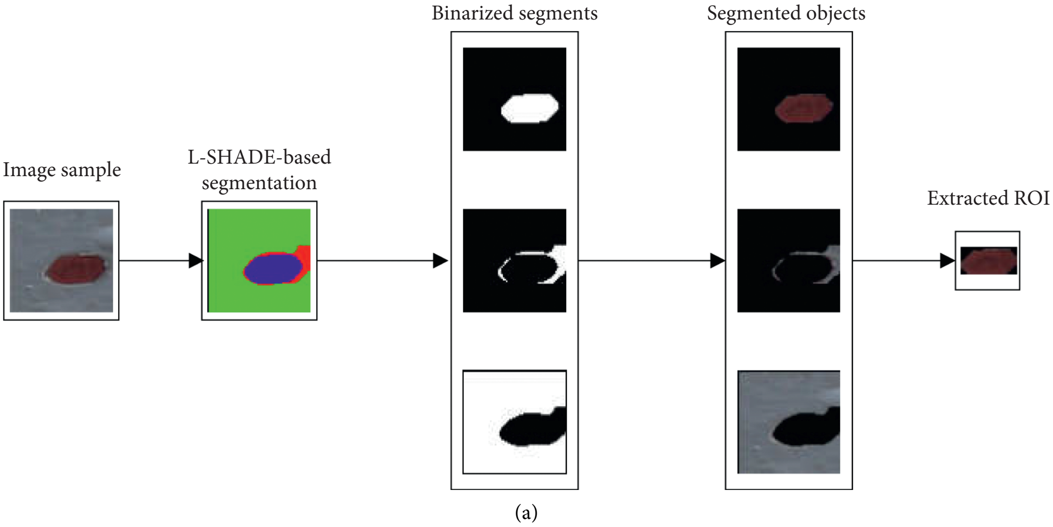
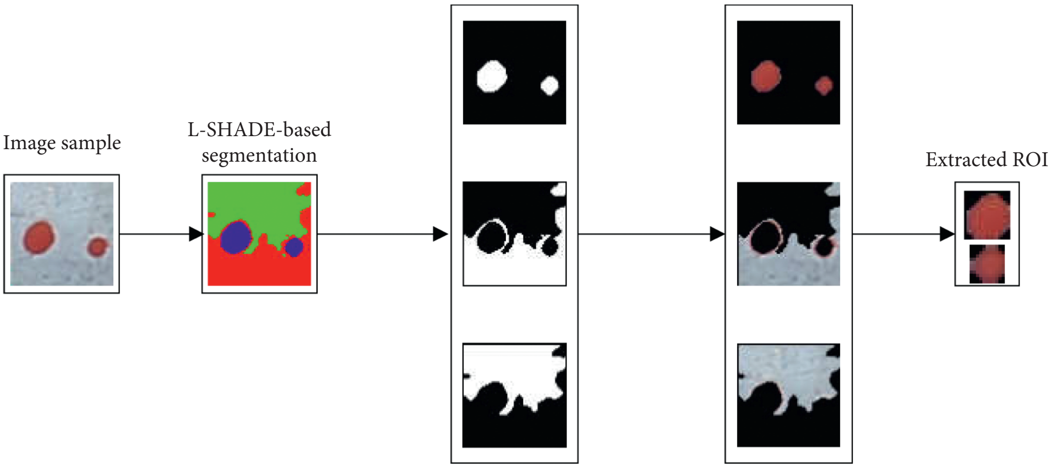


Figure 7: Continued.



(b)

Figure 7: Extraction of region of interest (ROI) from image containing no pitting corrosion: (a) one object and (b) multiple objects.

where FNR_k and FPR_k represent the false negative rate (FNR) and the false positive rate (FPR) obtained from kth run of the aforementioned K-fold crossvalidation, respectively.

The FNR and FPR are calculated as follows:

$$\begin{aligned} \text{FNR} &= \frac{\text{FN}}{\text{FN} + \text{TP}}, \\ \text{FPR} &= \frac{\text{FP}}{\text{FP} + \text{TN}}, \end{aligned} \tag{16}$$

where FN, FP, TP, and TN denote false negative, false positive, true positive, and true negative data samples, respectively.

5. Experimental Results and Discussions

The proposed metaheuristic approach used for pitting corrosion detection, named as LSHADE-SVC-PCD, has been developed in Visual C#.NET environment (Framework 4.6.2). It is noted that the LSHADE optimization method as well as the employed image processing techniques have been constructed by the authors; the SVM classification model is implemented via built-in functions supported by the Accord. NET Framework [97]. Experiments with the newly developed LSHADE-SVC-PCD model are performed on the ASUS FX705GE - EW165T (Core i7 8750H, 8GB Ram, and 256GB solid-state drive).

To train and test the integrated LSHADE-SVC-PCD model used for pitting corrosion detection, the collected dataset has been divided into two subsets of training and testing datasets. The training dataset, accounting for 90% of the original dataset, is used for model construction and the rest of the dataset is reserved for testing the model generalization. In addition, to alleviate the effect of randomness caused by data sampling and to evaluate the predictive capability of the newly developed method reliably, the training and testing data sampling process has been repeated 20 times. In each time, 10% of the dataset is randomly chosen to create the testing dataset; the other 90% of the dataset is employed for model training. The datasets used for model training and testing are illustrated in Table 1.

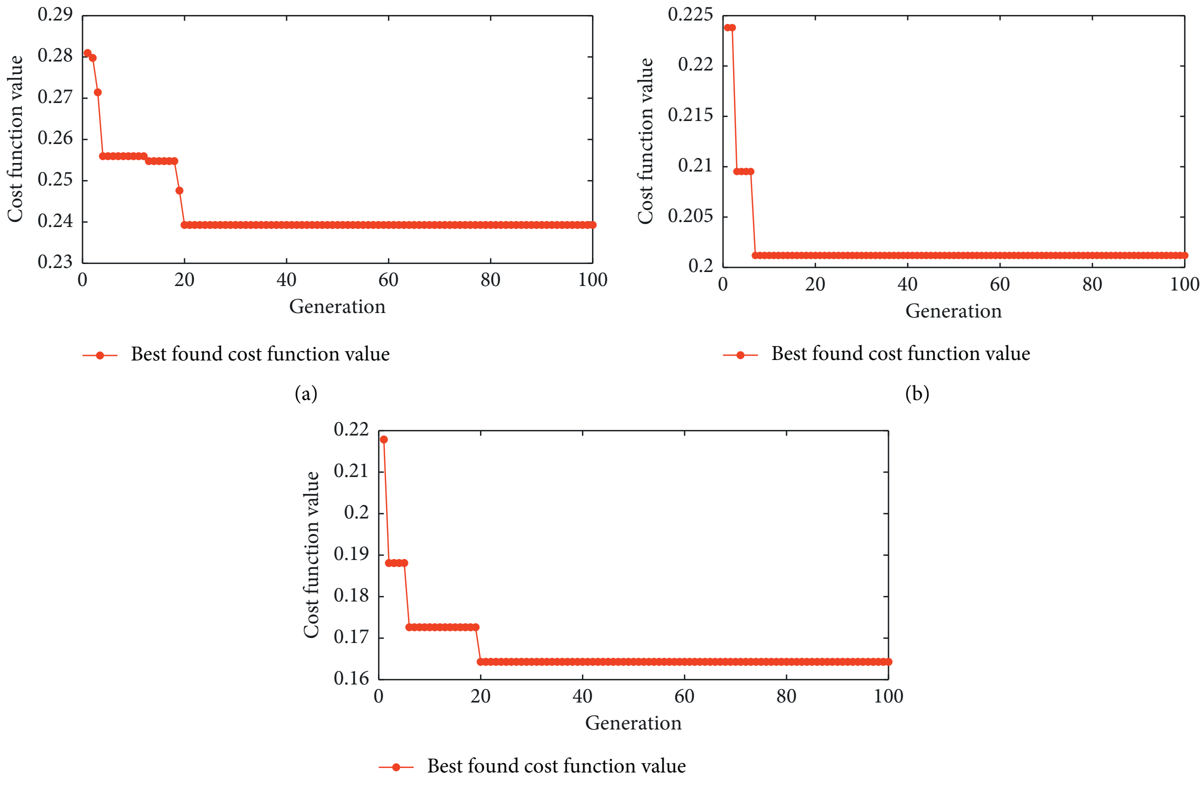
Moreover, to assess the predictive capability of the proposed LSHADE-SVC-PCD, classification accuracy rate (CAR), precision, recall, negative predictive value (NPV), and F1 score are computed from the outcomes of the TP, TN, FP, and FN. These indices are computed in the following equations [98]:

$$\begin{aligned} \text{CAR} &= \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \times 100\%, \\ \text{Precision} &= \frac{\text{TP}}{\text{TP} + \text{FP}}, \\ \text{Recall} &= \frac{\text{TP}}{\text{TP} + \text{FN}}, \\ \text{NPV} &= \frac{\text{TN}}{\text{TN} + \text{FN}}, \\ \text{F1 Score} &= \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}}. \end{aligned} \tag{17}$$

The LSHADE with an initial population size = 20 and a maximum number of evolutionary generations = 100 has been employed and utilized to determine the most appropriate set of the SVM model's hyperparameters including the penalty parameter (c) and the kernel function parameter (σ). As stated earlier, the original dataset having 93 features has been inspected by the commonly used principal component analysis (PCA) to seek for possibility of dimensionality reduction. Based on the PCA outcomes with three scenarios of the values of total variance explained (90%, 95%, and 99%), the dimensionality of the original dataset can be reduced to 4, 8, and 19. The evolutionary progresses of the LSHADE metaheuristic-optimized SVM corresponding to these three scenarios of dimensionality reduction are graphically described in Figure 8. The LSHADE optimization results are reported in Table 2. The prediction performances of the

Table 1: The training and testing datasets.

Datasets	Samples	Features							Class Label
		F1	F2	F3	. . .	F91	F92	F93	
Training	1	0.2188	0.0547	0.0117		1999.1644	0.0002	7.1721	−1
	2	0.2857	0.0769	0.0220		1287.5000	0.0004	6.6308	−1
	. . .	. . .	. . .	. . .	. . .	. . .	. . .	. . .	. . .
	222	0.2500	0.1167	0.0167		929.3030	0.0005	6.0505	1
	222	0.2143	0.0857	0.0143		1839.1722	0.0002	7.2027	1
Testing	1	0.1517	0.0227	0.0049		536.7429	0.0005	5.9750	−1
	2	0.1909	0.0636	0.0182		1091.6222	0.0005	6.1862	−1
	. . .	. . .	. . .	. . .	. . .	. . .	. . .	. . .	. . .
	24	0.0909	0.0364	0.0136		608.5503	0.0008	7.2868	1
	26	0.0859	0.0286	0.0052		638.9420	0.0007	7.4187	1



(a)

(b)

(c)

Figure 8: The LSHADE optimization process for datasets: (a) Case 1 (dimensionality = 4), (b) Case 2 (dimensionality = 8), and (c) Case 3 (dimensionality = 19).

Table 2: Optimization results of the LSHADE-SVC-PCD.

Case	Total variance explained (%)	Data dimensionality	The penalty coefficient	The kernel function parameter
1	90	4	498.32	33.15
2	95	8	443.43	9.89
3	99	19	422.80	56.66

LSHADE-SVC-PCD corresponding to different running cases are summarized in Table 3. As can be seen from this table, the dataset from Case 3 has helped to gain the most desired testing performance with CAR = 91.80%, precision = 0.91, recall = 0.94, NPV = 0.93, and F1 score = 0.92. Therefore, this dataset has been selected to be used in the subsequent part of model result comparison.

In addition, to demonstrate the superiority of the proposed LSHADE-SVC-PCD, the Backpropagation Neural Network (BPNN) [99, 100] and the Random Forest models

Table 3: Prediction performance of the LSHADE-SVC-PCD.

Case	Phases	Indices	CAR (%)	TP	TN	FP	FN	Precision	Recall	NPV	F1 score
1	Training	Mean	86.61	97.55	95.60	16.15	13.70	0.86	0.88	0.87	0.87
		Std.	0.99	2.96	1.96	1.39	1.84	0.01	0.02	0.01	0.01
	Testing	Mean	85.20	10.90	10.40	1.85	1.85	0.85	0.86	0.85	0.85
		Std.	6.50	1.97	1.82	1.18	1.35	0.10	0.10	0.10	0.07
2	Training	Mean	92.83	104.35	102.65	9.05	6.95	0.92	0.94	0.94	0.93
		Std.	0.60	2.83	2.30	0.94	1.10	0.01	0.01	0.01	0.01
	Testing	Mean	87.20	11.35	10.45	1.85	1.35	0.86	0.89	0.89	0.87
		Std.	6.03	2.64	1.79	1.35	0.99	0.11	0.08	0.08	0.07
3	Training	Mean	93.05	105.55	101.95	9.90	5.60	0.91	0.95	0.95	0.93
		Std.	0.69	2.46	2.27	1.14	1.56	0.01	0.01	0.01	0.01
	Testing	Mean	91.80	12.00	10.95	1.20	0.85	0.91	0.94	0.93	0.92
		Std.	4.64	1.70	2.04	0.93	1.01	0.07	0.07	0.08	0.04

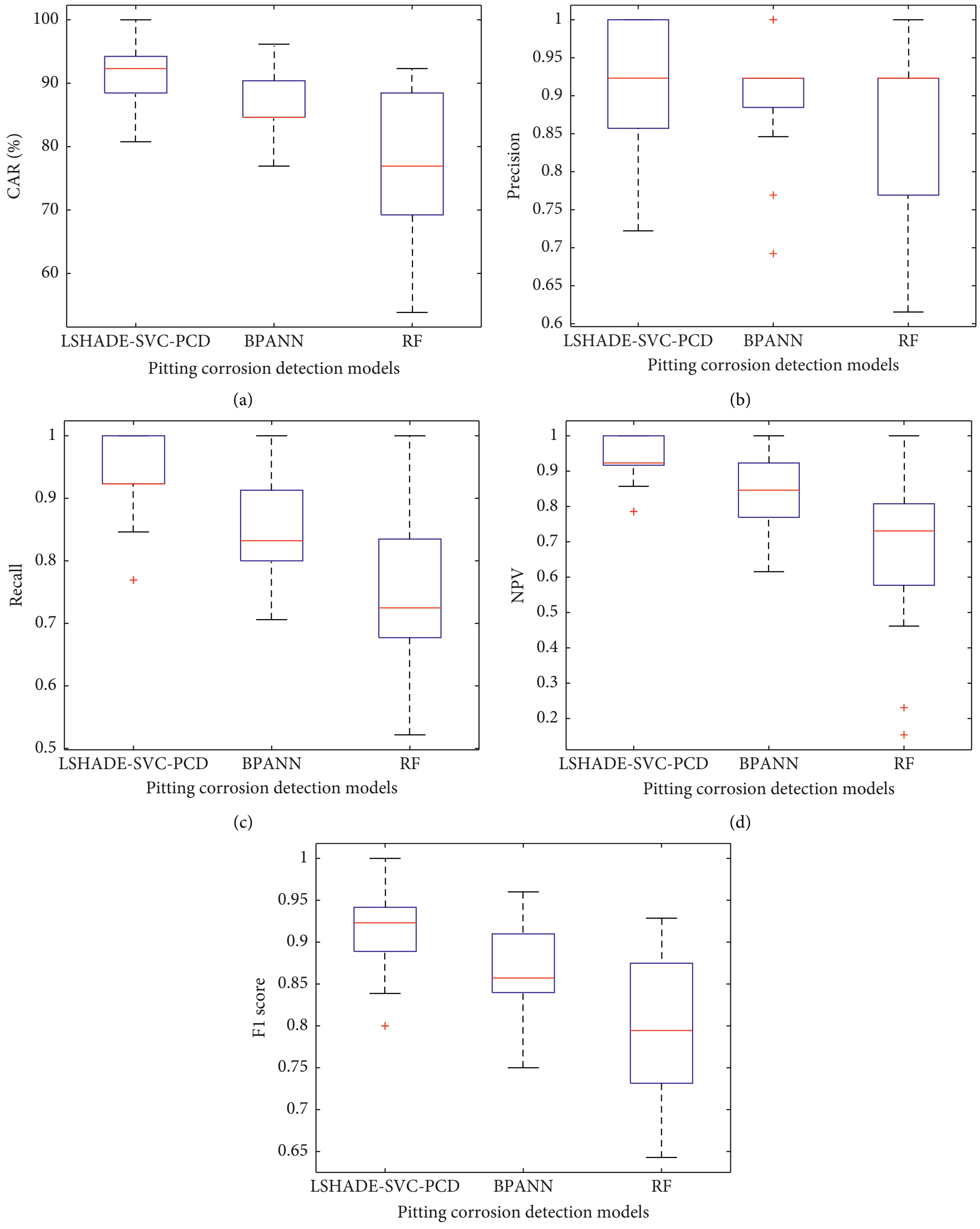
Table 4: Prediction result comparison.

Phase	Indices	LSHADE-SVC-PCD		BPANN		RF	
		Mean	Std	Mean	Std	Mean	Std
Training	CAR (%)	93.11	0.73	94.05	2.03	92.61	7.80
	TP	105.75	1.13	106.75	2.70	108.05	4.35
	TN	100.95	1.24	102.05	3.19	97.55	17.44
	FP	10.05	1.24	4.25	2.70	2.95	4.35
	FN	5.25	1.13	8.95	3.19	13.45	17.44
	Precision	0.91	0.01	0.96	0.02	0.97	0.04
	Recall	0.95	0.01	0.92	0.03	0.90	0.10
	NPV	0.95	0.01	0.92	0.03	0.88	0.16
	F1 score	0.93	0.01	0.94	0.02	0.93	0.06
Testing	CAR (%)	91.54	5.91	86.15	5.63	77.31	10.94
	TP	12.00	0.89	11.75	0.94	11.25	1.44
	TN	11.80	1.21	10.65	1.42	8.85	2.74
	FP	1.20	1.21	1.25	0.94	1.75	1.44
	FN	1.00	0.89	2.35	1.42	4.15	2.74
	Precision	0.91	0.07	0.90	0.07	0.87	0.11
	Recall	0.92	0.07	0.84	0.08	0.75	0.12
	NPV	0.93	0.06	0.82	0.11	0.68	0.21
	F1 score	0.92	0.06	0.87	0.05	0.80	0.09

are used as benchmark methods. The BPNN- and RF-based classifiers have been developed in Visual C#.NET by the authors and trained with the mini-batch mode [100, 101]; the batch size is selected to be 32 and the number of neuron in the hidden layer is selected to be $(2/3)D_X + C_N$ according to the recommendation of Heaton [102]; herein, D_X and C_N are the numbers of features and class outputs, respectively. The BPNN model is then constructed with the sigmoidal activation function with the maximum number of epochs = 1000 epochs and the learning rate = 0.01. Moreover, the RF model is established with the number of individual classification tree = 50.

The prediction performances of the proposed LSHADE-SVC-PCD model and the two benchmark models are summarized in Table 4 and box plots are provided in Figure 9. As can be observed, the performance of the LSHADE-SVC-PCD (CAR = 91.80%, precision = 0.91, recall = 0.94, NPV = 0.93, and F1 score = 0.92) is better than those of the BPANN (CAR = 86.15%, precision = 0.90, recall = 0.84, NPV = 0.82, and F1 score = 0.87) and RF (CAR = 77.31%, precision = 0.87, recall = 0.75, NPV = 0.68, and F1 score = 0.80). Furthermore, the two-sample t-test [103] with a focus on the CAR index is also employed in this study to better confirm the statistical significance of the model predictive performances. This statistical test is often employed to inspect the null hypothesis that prediction performances may be drawn from normal distributions with equal means. In this experiment, the significant level (p-value) of the test is set to be 0.05 and results of the hypothesis testing are provided in Table 5. As can be seen from the testing results, the p-values < 0.05 reliably reject the null hypothesis.

Moreover, the coefficients of variation of the proposed model as well as the two benchmark models are provided in Figure 10. The coefficient of variation (COV) [104], also regarded as the relative standard deviation, is an index for measuring dispersion of a probability distribution. The COV is computed as the ratio of the standard deviation to the mean and can express the reliability of a prediction model's performance [105]. Generally, a small COV value associates with a small variation on prediction outcome and is an indicator of a reliable machine-learning model. As can be



(a)

(b)

(c)

(d)

(e)

Figure 9: Box plots of model performances: (a) CAR, (b) precision, (c) recall, (d) NPV, and (e) F1 Score.

Table 5: The t-test outcomes of pairwise model comparisons.

Model comparison	Test outcome	p-value
LSHADE-SVM-PCD vs. BPANN	Significant	0.00656
LSHADE-SVM-PCD vs. RF	Significant	0.00001

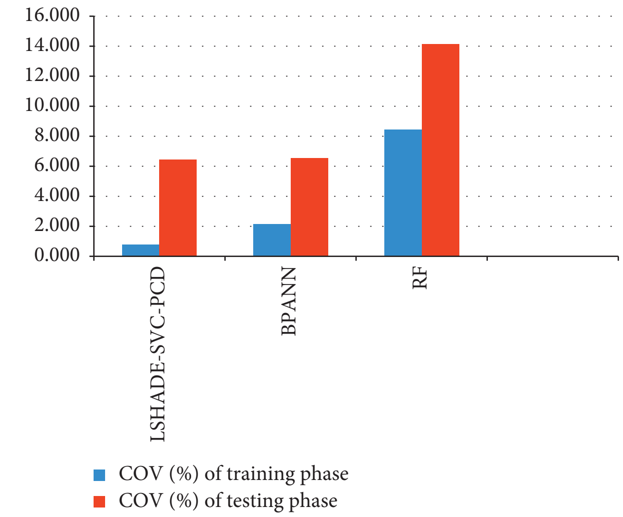


Figure 10: Analysis on the coefficient of variation (COV) of the model performances.

seen from Figure 10, the COV values computed for the CAR index (for the training and testing phases) points out that the LSHADE-SVC-PCD (with COV = 0.78% for the training phase and COV = 6.46% for the testing phase) is superior to the BPANN (with COV = 2.16% for the training phase and COV = 6.53% for the testing phase) and the RF (with COV = 8.43% for the training phase and COV = 14.15% for the testing phase). Thus, it is able to confirm that the proposed LSHADE-SVC-PCD is best suited for the task of pitting corrosion detection.

Based on the experimental results, the proposed framework which integrates the LSHADE metaheuristic, multilevel image thresholding, image processing, and SVM-based pattern recognition can deliver satisfactory performance on the task of pitting corrosion detection. Nevertheless, since the role of the LSHADE metaheuristic in this study is two-fold, optimizing the multilevel image thresholding and fine-tuning the SVM-based pattern recognition model, a future direction of the current work may consider to apply multiobjective metaheuristic optimization methods [23, 26, 106]. In addition, since the background of metal surfaces may contain noisy objects, sophisticated image quality enhancement methods including image dehazing [107–110], image filtering [111–114], gradient profile prior [115–118], and deep-learning-based image fusion [119–121] can be useful for meliorating the accuracy rate of pitting corrosion detection.

6. Conclusions

Pitting corrosion is a severe damage that can bring about critical failure of infrastructure elements. This study has put forward an intelligent method for automatic detection of this damage via the employment of the LSHADE metaheuristic, SVM machine-learning, and image-processing techniques. The application of metaheuristic in this study is multifold. The LSHADE metaheuristic is first utilized in the task of multilevel image thresholding to extract ROI which is subsequently used by the texture descriptors (the statistical measurements of color channels, the GLCM, and the LBP). The LSHADE optimizer is then applied to search for the most appropriate SVM model's hyperparameters including the penalty coefficient and the kernel function parameter. The SVM is then employed to generalize a classification boundary that separates input data into two distinctive categories of pitting corrosion and nonpitting corrosion. Experimental results with a repeated data sampling with 20 runs confirms that the newly developed LSHADE-SVC-PCD is highly suitable for the computer vision task of interest with CAR = 91.80%, precision = 0.91, recall = 0.94, NPV = 0.93, and F1 score = 0.92. Hence, the newly developed model can be a useful tool to assist infrastructure management agencies in the task of periodic structural health survey.

Although the LSHADE-SVC-PCD is capable of delivering good predictive outcome, one shortcoming of the study is that the number of the collected image samples is still limited. Therefore, more effort on data collected should be focused to enhance the size of the image dataset; this may help to improve the generalization of the pitting corrosion detection method. Another limitation of the current study is that the LSHADE-SVC has not been integrated with feature selection methods. Future directions of the current study may include the utilization of more sophisticated component-labeling algorithms, texture description, and feature selection methods to enhance the feature extraction phases. In addition, when the size of the collected dataset is enlarged, the employment of advanced deep-learning model can be worth-attempting to obtain higher detection accuracy. Although the LSHADE metaheuristic has resulted in good optimization outcomes, this method has a drawback of requiring a considerable amount of computational expense. Therefore, other sophisticated hyperparameters tuning (e.g., Bayesian optimization [122]) can be used as alternative approaches for optimizing the SVC-based pitting corrosion detection model.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was financially supported by Duy Tan University.

References

[1] Z. Chen, L. Liu, L. Li, and H. Li, “A two-stage model for project optimization in transportation infrastructure management system,” *Mathematical Problems in Engineering*, vol. 2014, Article ID 914515, 8 pages, 2014.

[2] M. Gao, X. Wang, S. Zhu, and P. Guan, “Detection and segmentation of cement concrete pavement pothole based on image processing technology,” *Mathematical Problems in Engineering*, vol. 2020, Article ID 1360832, 13 pages, 2020.

[3] B. Ghosh, V. Pakrashi, and F. Schoefs, "Detection of pitting corrosion in steel using image processing," in *Proceedings of the 2nd International Conference on Applications Heritage and Constructions in Coastal and Marine Environment, MEDACHS 2010*, La Rochelle, France, April 2010.

[4] T. P. Huynh and H. Haick, "Autonomous flexible sensors for health monitoring," *Advanced Materials*, vol. 30, no. 50, Article ID 1802337, 2018.

[5] M. R. Kaloop, J. W. Hu, E. Elbeltagi, and A. El Refai, "Structural health monitoring and assessment: sensors and analysis," *Journal of Sensors*, vol. 2018, Article ID 9834958, 2 pages, 2018.

[6] G. Lian-sheng, D. Han-cheng, and C. Jia-qi, "Research on predicting the rutting of asphalt pavement based on a simplified burgers creep model," *Mathematical Problems in Engineering*, vol. 2017, Article ID 3459704, 14 pages, 2017.

[7] T. Wang, M. Z. A. Bhuiyan, G. Wang, M. A. Rahman, J. Wu, and J. Cao, "Big data reduction for a smart city's critical infrastructural health monitoring," *IEEE Communications Magazine*, vol. 56, no. 3, pp. 128–133, 2018.

[8] W. Li, R. Deng, Y. Zhang, Z. Sun, X. Hao, and J. Huyan, "Three-dimensional asphalt pavement crack detection based on fruit fly optimisation density peak clustering," *Mathematical Problems in Engineering*, vol. 2019, Article ID 4302805, 15 pages, 2019.

[9] A. Rouhan and F. Schoefs, "Probabilistic modeling of inspection results for offshore structures," *Structural Safety*, vol. 25, no. 4, pp. 379–399, 2003.

[10] Z. Petrovic, "Catastrophes caused by corrosion military technical courier," 2016, https://scindeks-clanci.ceon.rs/data/pdf/0042-8469/2016/0042-84691604048P.pdf.

[11] F. Bonnin-Pascual and A. Ortiz, "Corrosion detection for automated visual inspection, developments in corrosion protection," in *Developments in Corrosion Protection*, pp. 620–632, IntechOpen, London, UK, 2014.

[12] A. Valor, F. Caleyo, L. Alfonso, J. C. Velázquez, and J. M. Hallen, "Markov chain models for the stochastic modeling of pitting corrosion," *Mathematical Problems in Engineering*, vol. 2013, Article ID 108386, 13 pages, 2013.

[13] V. Bondada, D. K. Pratihar, and C. S. Kumar, "Detection and quantitative assessment of corrosion on pipelines through image analysis," *Procedia Computer Science*, vol. 133, pp. 804–811, 2018.

[14] M. Enikeev, L. Enikeeva, and M. Maleeva, "Machine learning in the problem of recognition of pitting corrosion on aluminum surfaces," *Data Science* in *Proceedings of the 4th International Conference on Information Technology and Nanotechnology, Samara*, vol. 151, pp. 186–192, Moscow, Russia, April 2018.

[15] NACE, *Pitting Corrosion National Association of Corrosion Engineers*, NACE, Houston, TX, USA, 2019, https://www.nace.org/resources/general-resources/corrosion-basics/group-1/pitting-corrosion.

[16] A. Kreimer, *Managing Disaster Risk in Mexico: Market Incentives for Mitigation Investment*, World Bank Publications, Washington, DC, USA, 1999.

[17] M. Harkin, "Pitting corrosion, and what to do about it facility executive," 2017, https://www.facilityexecutive.com/2017/06/pitting-corrosion-what-to-do-about-it/.

[18] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*, Springer, Berlin, Germany, 2011.

[19] J.-S. Chou and T. T. H. Truong, "Sliding-window metaheuristic optimization-based forecast system for foreign exchange analysis," *Soft Computing*, vol. 23, no. 10, 2019.

[20] M. Kaur, H. K. Gianey, D. Singh, and M. Sabharwal, "Multi-objective differential evolution based random forest for e-health applications," *Modern Physics Letters B*, vol. 33, no. 5, Article ID 1950022, 2019.

[21] H. Moayedi, M. Gör, M. Khari, L. K. Foong, M. Bahiraei, and D. T. Bui, "Hybridizing four wise neural-metaheuristic paradigms in predicting soil shear strength," *Measurement*, vol. 156, p. 107576, 2020.

[22] S. Deng, X. Wang, Y. Zhu, F. Lv, and J. Wang, "Hybrid grey wolf optimization algorithm based support vector machine for groutability prediction of fractured rock," *Mass Journal of Computing in Civil Engineering*, vol. 33, Article ID 04018065, 2019.

[23] A. Gupta, D. Singh, and M. Kaur, "An efficient image encryption using non-dominated sorting genetic algorithm-III based 4-D chaotic maps," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 3, pp. 1309–1324, 2020.

[24] M. Kaur and V. Kumar, "Beta chaotic map based image encryption using genetic algorithm," *International Journal of Bifurcation and Chaos*, vol. 28, no. 11, Article ID 1850132, 2018.

[25] D. Prayogo and Y. T. T. Susanto, "Optimizing the prediction accuracy of friction capacity of driven piles in cohesive soil using a novel self-tuning least squares support vector machine," *Advances in Civil Engineering*, vol. 2018, Article ID 6490169, 9 pages, 2018.

[26] M. Kaur, D. Singh, K. Sun, and U. Rawat, "Color image encryption using non-dominated sorting genetic algorithm with local chaotic search based 5D chaotic map," *Future Generation Computer Systems*, vol. 107, pp. 333–350, 2020.

[27] M. Kaur and V. Kumar, "Parallel non-dominated sorting genetic algorithm-II-based image encryption technique," *The Imaging Science Journal*, vol. 66, no. 8, pp. 453–462, 2018.

[28] M. Kaur and V. Kumar, "A comprehensive review on image encryption techniques," *Archives of Computational Methods in Engineering*, vol. 27, no. 1, pp. 15–43, 2020.

[29] H. Moayedi, D. Tien Bui, A. Dounis, L. Kok Foong, and B. Kalantar, "Novel nature-inspired hybrids of neural computing for estimating soil shear strength," *Applied Sciences*, vol. 9, no. 21, p. 4643, 2019.

[30] A. P. Piotrowski, "Differential Evolution algorithms applied to Neural Network training suffer from stagnation," *Applied Soft Computing*, vol. 21, pp. 382–406, 2014.

[31] A. P. Piotrowski and J. J. Napiorkowski, "Optimizing neural networks for river flow forecasting - evolutionary Computation methods versus the Levenberg-Marquardt approach," *Journal of Hydrology*, vol. 407, no. 1–4, pp. 12–27, 2011.

[32] M. Abdel-Basset, L. Abdel-Fatah, and A. K. Sangaiah, "Chapter 10—metaheuristic algorithms: a comprehensive review," in *Computational Intelligence for Multimedia Big Data on the Cloud with Engineering Applications*, A. K. Sangaiah, M. Sheng, and Z. Zhang, Eds., Academic Press, Cambridge, MA, USA, pp. 185–231, 2018.

[33] F. Han, J. Jiang, Q.-H. Ling, and B.-Y. Su, "A survey on metaheuristic optimization for random single-hidden layer feedforward neural network," *Neurocomputing*, vol. 335, pp. 261–273, 2019.

[34] V. K. Ojha, A. Abraham, and V. Snášel, "Metaheuristic design of feedforward neural networks: a review of two decades of research," *Engineering Applications of Artificial Intelligence*, vol. 60, pp. 97–116, 2017.

[35] R. Tanabe and A. Fukunaga, "Improving the search performance of SHADE using linear population size reduction," in *Proceedings of the 2014 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1658–1665, Beijing, China, July 2014.

[36] P. Biswas, P. Suganthan, and G. Amaratunga, "Optimal power flow solutions using algorithm success history based adaptive differential evolution with linear population reduction," in *Proceedings of the 2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 249–254, Miyazaki, Japan, October 2018.

[37] A. Viktorin, R. Senkerik, M. Pluhacek, T. Kadavy, and A. Zamuda, "Distance based parameter adaptation for Success-History based Differential Evolution," *Swarm and Evolutionary Computation*, vol. 50, Article ID 100462, 2019.

[38] A. Zamuda, "Adaptive constraint handling and success history differential evolution for CEC 2017 constrained real-parameter optimization," in *Proceedings of the 2017 IEEE Congress on Evolutionary Computation (CEC)*, pp. 2443–2450, San Sebastian, Spain, June 2017.

[39] K. Y. Choi and S. S. Kim, "Morphological analysis and classification of types of surface corrosion damage by digital image processing," *Corrosion Science*, vol. 47, no. 1, pp. 1–15, 2005.

[40] F. N. S. Medeiros, G. L. B. Ramalho, M. P. Bento, and L. C. L. Medeiros, "On the evaluation of texture and color features for nondestructive corrosion detection," *EURASIP Journal on Advances in Signal Processing*, vol. 2010, no. 1, Article ID 817473, 2010.

[41] P. H. Chen, H. K. Shen, C. Y. Lei, and L. M. Chang, "Fourier-transform-based method for automated steel bridge coating defect recognition," *Procedia Engineering*, vol. 14, pp. 470–476, 2011.

[42] G. Ji, Y. Zhu, and Y. Zhang, "The corroded defect rating system of coating material based on computer vision," *Transactions on Edutainment VIII*, Springer, Berlin, Germany, pp. 210–220, 2012.

[43] H.-K. Shen, P.-H. Chen, and L.-M. Chang, "Automated steel bridge coating rust defect recognition method based on color and texture feature," *Automation in Construction*, vol. 31, pp. 338–356, 2013.

[44] M. R. Jahanshahi and S. F. Masri, "Effect of color space, color channels, and sub-image block size on the performance of wavelet-based texture analysis algorithms: an application to corrosion detection on steel structures," *Computing in Civil Engineering*, vol. 2013, pp. 685–692, 2013.

[45] K.-W. Liao and Y.-T. Lee, "Detection of rust defects on steel bridge coatings via digital image recognition," *Automation in Construction*, vol. 71, pp. 294–306, 2016.

[46] K.-W. Liao and D.-R. Cheng, "Restoration of the distorted color to detect the discoloration status of a steel bridge coating using digital image measurements," *Advanced Engineering Informatics*, vol. 33, pp. 96–111, 2017.

[47] Y. Zhao, G. Wang, H. Zhang, and L. Li, "Material corrosion classification based on deep learning," *Chemical Engineering Transactions*, vol. 71, pp. 775–780, 2018.

[48] C. Zhang, C. Zhu, C. Yang, and C. Zhu, "Revisiting image ordinal estimation: how to deal with ordinal relationship in deep learning?" *Journal of Electronic Imaging*, vol. 28, no. 1, pp. 1–14, 2019.

[49] I. Ivasenko and V. Chervatyuk, "Detection of rust defects of protective coatings based on HSV color model," in *Proceedings of the 2019 IEEE 2nd Ukraine Conference on Electrical and Computer Engineering (UKRCON)*, pp. 1143–1146, Lviv, Ukraine, July 2019.

[50] S. K. Ahuja and M. K. Shukla, "A survey of computer vision based corrosion detection approaches," in *Proceedings of the Information and Communication Technology for Intelligent Systems (ICTIS 2017)*, vol. 2, pp. 55–63, Cham, Switzerland, 2018.

[51] G. Ding, F. Dong, and H. Zou, "Fruit fly optimization algorithm based on a hybrid adaptive-cooperative learning and its application in multilevel image thresholding," *Applied Soft Computing*, vol. 84, Article ID 105704, 2019.

[52] N.-D. Hoang, Q.-L. Nguyen, and D. T. Bui, "Image processing-based classification of asphalt pavement cracks using support vector machine optimized by artificial bee colony," *Journal of Computing in Civil Engineering*, vol. 32, Article ID 04018037, 2018.

[53] J. Qin, C. Wang, and G. Qin, "A multilevel image thresholding method based on subspace elimination optimization," *Mathematical Problems in Engineering*, vol. 2019, Article ID 6706590, 11 pages, 2019.

[54] O. Tarkhaneh and H. Shen, "An adaptive differential evolution algorithm to optimal multi-level thresholding for MRI brain image segmentation," *Expert Systems with Applications*, vol. 138, Article ID 112820, 2019.

[55] M. Wang, G. Pan, and Y. Liu, "A novel imperialist competitive algorithm for multithreshold image segmentation," *Mathematical Problems in Engineering*, vol. 2019, Article ID 5982410, 18 pages, 2019.

[56] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[57] N.-D. Hoang, "Detection of surface crack in building structures using image processing technique with an improved Otsu method for image thresholding," *Advances in Civil Engineering*, vol. 2018, Article ID 3924120, 10 pages, 2018.

[58] S. C. Satapathy, N. Sri Madhava Raja, V. Rajinikanth, A. S. Ashour, and N. Dey, "Multi-level image thresholding using Otsu and chaotic bat algorithm," *Neural Computing and Applications*, vol. 29, no. 12, pp. 1285–1307, 2018.

[59] X. Yue and H. Zhang, "A multi-level image thresholding approach using Otsu based on the improved invasive weed optimization algorithm," *Signal, Image and Video Processing*, vol. 14, no. 3, pp. 575–582, 2019.

[60] M. H. Merzban and M. Elbayoumi, "Efficient solution of Otsu multilevel image thresholding: a comparative study," *Expert Systems with Applications*, vol. 116, pp. 299–309, 2019.

[61] N. Muangkote, K. Sunat, and S. Chiewchanwattana, "R r-cr -IJADE: an efficient differential evolution algorithm for multilevel image thresholding," *Expert Systems with Applications*, vol. 90, pp. 272–289, 2017.

[62] F. Diaz-del-Rio, P. Sanchez-Cuevas, H. Molina-Abril, and P. Real, "Parallel connected-Component-Labeling based on homotopy trees," *Pattern Recognition Letters*, vol. 131, pp. 71–78, 2020.

[63] L. He, X. Ren, Q. Gao, X. Zhao, B. Yao, and Y. Chao, "The connected-component labeling problem: a review of state-of-the-art algorithms," *Pattern Recognition*, vol. 70, pp. 25–43, 2017.

[64] L. G. Shapiro and G. C. Stockman, *Computer Vision*, Prentice-Hall, Upper Saddle River, NJ, USA, 2001.

[65] K. Appiah, A. Hunter, P. Dickinson, and H. Meng, "Accelerated hardware video object segmentation: from foreground detection to connected components labelling,"

Computer Vision and Image Understanding, vol. 114, no. 11, pp. 1282–1291, 2010.

[66] R. M. Haralick, "Some neighborhood operators," in *Real-Time Parallel Computing: Imaging Analysis*, M. Onoe, K. Preston, and A. Rosenfeld, Eds., Springer US, Boston, MA, USA, pp. 11–35, 1981.

[67] N.-D. Hoang, "Automatic detection of asphalt pavement raveling using image texture based feature extraction and stochastic gradient descent logistic regression," *Automation in Construction*, vol. 105, Article ID 102843, 2019.

[68] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*, Academic Press, Cambridge, MA, USA, 2009.

[69] R. M. Haralick, K. Shanmugam, and I. H. Dinstein, "Textural features for image classification," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 6, pp. 610–621, 1973.

[70] R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1992.

[71] G. M. Hadjidemetriou, P. A. Vela, and S. E. Christodoulou, "Automated pavement patch detection and quantification using support vector machines," *Journal of Computing in Civil Engineering*, vol. 32, Article ID 04017073, 2018.

[72] F. Tomita and S. Tsuji, *Computer Analysis of Visual Textures*, Springer Science & Business Media, New York, NY, USA, 1990.

[73] T. Ojala and M. Pietikäinen, "Unsupervised texture segmentation using feature distributions," *Pattern Recognition*, vol. 32, no. 3, pp. 477–486, 1999.

[74] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," *Pattern Recognition*, vol. 29, no. 1, pp. 51–59, 1996.

[75] D. Huang, C. Shan, M. Ardabilian, Y. Wang, and L. Chen, "Local binary patterns and its application to facial image analysis: a survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 41, no. 6, pp. 765–781, 2011.

[76] A. Hadid, G. Zhao, T. Ahonen, and M. Pietikäinen, "Face analysis using local binary patterns," in *Handbook of Texture Analysis*, pp. 347–373, Imperial College Press, London , UK, 2008.

[77] L. Liu, P. Fieguth, G. Zhao, and M. Pietikäinen, "Extended local binary pattern fusion for face recognition," in *Proceedings of the 2014 IEEE International Conference on Image Processing (ICIP)*, pp. 718–722, Paris, France, October 2014.

[78] F. Peng, L. Qin, and M. Long, "Face presentation attack detection based on chromatic Co-occurrence of local binary pattern and ensemble learning," *Journal of Visual Communication and Image Representation*, vol. 66, Article ID 102746, 2020.

[79] D. Grangier and S. Bengio, "A discriminative kernel-based approach to rank images from text queries," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 8, pp. 1371–1384, 2008.

[80] T. Mäenpää, J. Viertola, and M. Pietikäinen, "Optimising colour and texture features for real-time visual inspection," *Pattern Analysis & Applications*, vol. 6, no. 3, pp. 169–175, 2003.

[81] A. Lucieer, A. Stein, and P. Fisher, "Multivariate texture-based segmentation of remotely sensed imagery for extraction of objects and their uncertainty," *International Journal of Remote Sensing*, vol. 26, no. 14, pp. 2917–2936, 2005.

[82] S. Arivazhagan, R. N. Shebiah, R. Harini, and S. Swetha, "Human action recognition from RGB-D data using complete local binary pattern," *Cognitive Systems Research*, vol. 58, pp. 94–104, 2019.

[83] V. N. Vapnik, *Statistical Learning Theory*, John Wiley & Sons, Inc, Hoboken, NJ, USA, 1998.

[84] L. H. Hamel, *Knowledge Discovery with Support Vector Machines*, John Wiley & Sons, Inc., Hoboken, NJ, USA, 2009.

[85] B. T. Pham, A. Jaafari, I. Prakash, and D. T. Bui, "A novel hybrid intelligent model of support vector machines and the MultiBoost ensemble for landslide susceptibility modeling," *Bulletin of Engineering Geology and the Environment*, vol. 78, no. 4, pp. 2865–2886, 2018.

[86] B. T. Pham, D. Tien Bui, and I. Prakash, "Bagging based Support Vector Machines for spatial prediction of landslides," *Environmental Earth Sciences*, vol. 77, no. 4, p. 146, 2018.

[87] H. Shin and J. Paek, "Automatic task classification via support vector machine and crowdsourcing," *Mobile Information Systems*, vol. 2018, Article ID 6920679, 9 pages, 2018.

[88] N.-D. Hoang and V.-D. Tran, "Image processing-based detection of pipe corrosion using texture analysis and metaheuristic-optimized machine learning approach," *Computational Intelligence and Neuroscience*, vol. 2019, Article ID 8097213, 13 pages, 2019.

[89] H. Wei, M. Wang, B. Song, X. Wang, and D. Chen, "Study on the magnitude of reservoir-triggered earthquake based on support vector machines," *Complexity*, vol. 2018, Article ID 2830690, 10 pages, 2018.

[90] Y. Zhou, W. Su, L. Ding, H. Luo, and P. E. D. Love, "Predicting safety risks in deep foundation pits in subway infrastructure projects: support vector machine approach," *Journal of Computing in Civil Engineering*, vol. 31, Article ID 04017052, 2017.

[91] R. Tanabe and A. Fukunaga, "Success-history based parameter adaptation for Differential Evolution," in *Proceedings of the 2013 IEEE Congress on Evolutionary Computation*, pp. 71–78, Cancun, Mexico, June 2013.

[92] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[93] N. H. Awad, M. Z. Ali, P. N. Suganthan, and R. G. Reynolds, "An ensemble sinusoidal parameter adaptation incorporated with L-SHADE for solving CEC2014 benchmark problems," in *Proceedings of the 2016 IEEE Congress on Evolutionary Computation (CEC)*, pp. 2958–2965, Vancouver, Canada, July 2016.

[94] A. P. Piotrowski, "Review of differential evolution population size," *Swarm and Evolutionary Computation*, vol. 32, pp. 1–24, 2017.

[95] A. P. Piotrowski, "L-SHADE optimization algorithms with population-wide inertia," *Information Sciences*, vol. 468, pp. 117–141, 2018.

[96] T.-D. Nguyen, T.-H. Tran, H. Nguyen, and H. Nhat-Duc, "A success history-based adaptive differential evolution optimized support vector regression for estimating plastic viscosity of fresh concrete," *Engineering with Computers*, 2019.

[97] Accord, "Accord. NET framework," 2019, http://www.accord-framework.net/.

[98] D. Tien Bui, N.-D. Hoang, H. Nguyen, and X.-L. Tran, "Spatial prediction of shallow landslide using Bat algorithm optimized machine learning approach: a case study in Lang

Son Province, Vietnam," *Advanced Engineering Informatics*, vol. 42, Article ID 100978, 2019.

[99] M. T. Hagan, H. B. Demuth, M. H. Beale, and O. D. Jesús, *Neural Network Design*, Martin Hagan, Atlanta, GA, USA, 2nd edition, 2014.

[100] G. Montavon, G. Orr, and K.-R. Müller, *Neural Networks: Tricks of the Trade*, Springer-Verlag, Berlin, Germany, 2012.

[101] S. Skansi, *Introduction to Deep Learning—From Logical Calculus to Artificial Intelligence*, Springer International Publishing, New York, NY, USA, 2018.

[102] J. Heaton, "Artificial intelligence for humans," *Deep Learning and Neural Networks*, vol. 3, 2015.

[103] G. W. Snedecor and W. G. Cochran, *Statistical Methods*, Iowa State University Press, Iowa, IA, USA, 8th edition, 1989.

[104] B. Everitt, *The Cambridge Dictionary of Statistics*, Cambridge University Press, Cambridge, MA, USA, 1998.

[105] T. Whelen and P. Siqueira, "Coefficient of variation for use in crop area classification across multiple climates," *International Journal of Applied Earth Observation and Geoinformation*, vol. 67, pp. 114–122, 2018.

[106] C. C. Coello, G. B. Lamont, and V. DAv, *Evolutionary Algorithms for Solving Multi-Objective Problems*, Springer, Berlin, Germany, 2007.

[107] D. Berman, T. Treibitz, and S. Avidan, "Non-local image dehazing," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1674–1682, Las Vegas, NV, USA, June 2016.

[108] M. Kaur, D. Singh, V. Kumar, and K. Sun, "Color image dehazing Using gradient channel prior and guided L0," *Filter Information Sciences*, vol. 521, pp. 326–342, 2020.

[109] D. Singh and V. Kumar, "A comprehensive review of computational dehazing techniques," *Archives of Computational Methods in Engineering*, vol. 26, pp. 1395–1413, 2019.

[110] D. Singh, V. Kumar, and M. Kaur, "Image dehazing using window-based integrated means filter," *Multimedia Tools and Applications*, 2019.

[111] H. Chen, H. Zhao, D. Han, and K. Liu, "Accurate and robust crack detection using steerable evidence filtering in electroluminescence images of solar cells," *Optics and Lasers in Engineering*, vol. 118, pp. 22–33, 2019.

[112] N.-D. Hoang and Q.-L. Nguyen, "Fast local laplacian-based steerable and sobel filters integrated with adaptive boosting classification tree for automatic recognition of asphalt pavement cracks," *Advances in Civil Engineering*, vol. 2018, Article ID 5989246, 17 pages, 2018.

[113] D. Singh and V. Kumar, "Dehazing of outdoor images using notch based integral guided filter," *Multimedia Tools and Applications*, vol. 77, no. 20, pp. 27363–27386, 2018.

[114] D. Singh and V. Kumar, "Single image defogging by gain gradient image filter," *Science China Information Sciences*, vol. 62, no. 7, p. 79101, 2019.

[115] J. Pan, D. Sun, H. Pfister, and M. Yang, "Blind image deblurring using dark channel prior," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1628–1636, Las Vegas, NV, USA, June 2016.

[116] D. Singh and V. Kumar, "Image dehazing using Moore neighborhood-based gradient profile prior," *Signal Processing: Image Communication*, vol. 70, pp. 131–144, 2019b.

[117] D. Singh, V. Kumar, and M. Kaur, "Single image dehazing using gradient channel prior," *Applied Intelligence*, vol. 49, no. 12, pp. 4276–4293, 2019.

[118] J. Sun, Z. Xu, and H.-Y. Shum, "Image super-resolution using gradient profile prior," in *Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, Anchorage, AK, USA, June 2008.

[119] M. Kaur and D. Singh, "Fusion of medical images using deep belief networks," *Cluster Computing*, 2019.

[120] Y. Liu, X. Chen, Z. Wang, Z. J. Wang, R. K. Ward, and X. Wang, "Deep learning for pixel-level image fusion," *Recent advances and future prospects Information Fusion*, vol. 42, pp. 158–173, 2018.

[121] J. Piao, Y. Chen, and H. Shin, "A new deep learning based multi-spectral image fusion method," *Entropy*, vol. 21, no. 6, p. 570, 2019.

[122] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer Science+Business Media, New York, NY, USA, 2009.

Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 4515480, 14 pages
https://doi.org/10.1155/2020/4515480

Research Article

A VNS-EDA Algorithm-Based Feature Selection for Credit Risk Classification

Wei Chen,[1] **Zhongfei Li**,[1,2] **and Jinchao Guo**[2]

[1]*School of Business, Sun Yat-Sen University, Guangzhou 510275, China*
[2]*School of Management, Xinhua College of Sun Yat-Sen University, Guangzhou 510520, China*

Correspondence should be addressed to Zhongfei Li; lnslzf@mail.sysu.edu.cn

Received 28 December 2019; Revised 2 March 2020; Accepted 12 March 2020; Published 27 April 2020

Guest Editor: Dr. Dilbag Singh

Many quantitative credit scoring models have been developed for credit risk assessment. Irrelevant and redundant features may deteriorate the performance of credit risk classification. Feature selection with metaheuristic techniques can be applied to excavate the most significant features. However, metaheuristic techniques suffer from various issues such as being trapped in local optimum and premature convergence. Therefore, in this article, a hybrid variable neighborhood search and estimation of distribution technique with the elitist population strategy is proposed to identify the optimal feature subset. Variable neighborhood search with the elitist population strategy is used to direct its local searching in order to optimize the ergodicity, avoid premature convergence, and jump out of the local optimum in the searching process. The probabilistic model attempts to capture the probability distribution of the promising solutions which are biased towards the global optimum. The proposed technique has been tested on both publicly available credit datasets and a real-world credit dataset in China. Experimental analysis demonstrates that it outperforms existing techniques in large-scale credit datasets with high dimensionality, making it well suited for feature selection in credit risk classification.

1. Introduction

Credit risk assessment is one of the most important issues for serving small- and medium-sized enterprises (SMEs) in the commercial banking industry. Credit risk scoring model based on data mining and machine learning technique has already aroused great concern in financial credit field [1–4]. The scoring model can avoid some problems such as inaccuracy of risk classification and low precision of quantitative credit scoring, caused by the information asymmetry, to some extent. It is clear that, as an effective credit risk assessment tool, credit risk scoring model has played a more and more important role in financial institutions' credit decision-making.

Currently, data availability in credit loan is significantly enhanced by information technology. Multisource data, such as personal basic information, economy behavior, and social activity, are required for credit risk scoring purposes, in order to take preventive measures during the credit monitoring process and prioritize recovery efforts. However, these large-scale data are commonly high dimensional that leads to the curse of dimensionality. Redundant and irrelevant features in credit risk scoring model can increase the complexity of computation and produce inaccurate results in risk analysis. As we known, dimensionality reduction is used for feature extraction, abandonment, and decorrelation in machine learning. Feature selection and feature extraction are two common methods for dimensionality reduction [5–7]. Both the methods not only help in simplifying features but also can enhance stability and generalization of the classifier to improve learning ability, efficiency, and convenience. The major difference between two methods is that features obtained by feature extraction are not the original feature, while features obtained by feature selection are the part of original feature [8, 9]. From the degree of the combination with machine learning, feature selection can be divided into three categories such as filter method, wrapper method, and embedding method [9–12]. The first one is irrespective to the subsequent machine learning process. The second one directly uses feature subset as criteria to evaluate the performance of learning model. The last technique integrates feature selection process into training a machine learning model. In practical classification problem, accuracy obtained by the wrapper and embedding method is usually higher than the filter method [13]. Besides, the optimal selection based on subspace searching can remove the redundant feature, and it makes the performance of this method is always better than the optimal selection based on sorting methods [14].

In credit risk assessment, feature selection is usually used in many different credit risk classification models and quantitative credit scoring model [15–17]. Financial institutions need to pay much attention to the most significant risk features which are from a large number of original features. Because they not only can better categorize credit risk, be further used to set interest on loan and construct prewarning management system, but also can be utilized directly for the establishment of credit risk system or evaluating index system to set up risk control mechanism. Consequently, identifying the optimal features is critical to increase the predictive accuracy and efficiency when using the credit risk scoring model. Metaheuristic techniques can be utilized in searching space reduction for feature selection [18]. However, metaheuristic techniques suffer from various issues such as being trapped in local optimum and premature convergence. Therefore, this article has proposed an efficient technique inspired by feature correlations and metaheuristic techniques to optimize feature selection quality. To evaluate the proposed method, the comparative experiments with state-of-the-art methods are carried out on three real-world credit datasets by considering classification quality measures: accuracy, F-measure, sensitivity, and specificity.

The rest of this article is organized as follows: Section 2 summarizes relevant literatures, Section 3 details the proposed technique, Section 4 describes credit datasets, Section 5 discusses experimental results, and Section 6 concludes the article and future directions.

2. Related Work

This section gives a brief review of previous works related to correlation-based feature selection and metaheuristic searching techniques and presents the aforementioned techniques in credit risk assessment.

2.1. Feature Selection Based on Correlations. Basically, the relevant features have two characteristics: the importance for the following applications such as being input variables in machine learning and low redundancy among the feature variables. A feature is considered irrelevant if it contains no information about classification. Redundancy is generally defined in terms of feature correlations. It is widely accepted that two features are redundant to each other if their values are correlated [19]. It has been shown that redundancy among the features can degrade performance. Furthermore, linear correlation analysis may not be able to detect nonlinear dependencies between features, especially for large-scale data mining problem. Therefore, feature correlations can be determined using symmetrical uncertainty (SU) [20] based on entropy which is a measure of uncertainty. It is empirically computed as follows:

$$SU(X,Y) = 2\left[\frac{H(X) - H(X \mid Y)}{H(X) + H(Y)}\right], \tag{1}$$

where $\mathrm{IG}(X \mid Y) = H(X) - H(X \mid Y)$ denotes information gain (IG) that measures the reduction in uncertainty about the value of X given the value of Y. Thus, the bigger the IG is, the more significant the correlation between X and Y is. $H(X) = -\sum_i P(x_i)\log_2(P(x_i))$ denotes $H(X)$ that represents entropy and measures the uncertainty about the values of X. Similarly, $H(X \mid Y) = -\sum_j P(y_j)\sum_i P(x_i \mid y_j)\log_2(P(x_i \mid y_j))$ denotes $H(X \mid Y)$ that is the conditional entropy and measures the uncertainty about the value of X given the value of Y.

Moreover, in order to overcome selection bias in favor of features with more values, each value should be normalized to [0, 1] for guaranteeing all values having the same scale. A value of 1 indicates that the value of either feature completely predicts the value of the other; a value of 0 indicates that X and Y are independent. Thus, SU can be used as a correlation measure between features. In feature correlation analysis, X and Y, respectively, represent the feature F_r and the classification Y. Therefore, the feature subset evaluation function of correlation-based feature selection (CFS) [21] is computed as follows:

$$\overset{\mathrm{SU_{CF}}}{M_S = k\sqrt{k + k(k-1)\ ^{\mathrm{SU_{FF}},}}} \tag{2}$$

where $\mathrm{SU_{CF}} = (1/k) \cdot \sum_{F_{ri} \in S} \mathrm{SU}(F_{ri}, Y)$ denotes the average correlations between feature F_r and the classification Y. $\mathrm{SU_{FF}} = (2/(k(k-1))) \cdot \sum_{F_{ri} \in S} \sum_{\substack{F_{rj} \in S \\ F_{ri} \neq Fj}} \mathrm{SU}(F_{ri}, F_{rj})$ denotes the average correlations between feature F_{ri} and the feature F_{rj}. CFS is based on feature subspace searching and derived from the definition of the evaluation function. The optimal feature subset with correlation measurement can be successfully obtained, and the predominant features are highly relevant to classification but are of small or no significant correlation to eachother.

2.2. Feature Selection Combining with Metaheuristic Techniques. Although there exist numerous feature selection techniques, the challenging issues that handle a large dimensionality number of samples has gained increasing attention [22]. In credit risk assessment, the goal of feature selection is to minimize the error with respect to the given inputs for credit classification and scoring. Piramuthu [23] discussed a few means of improving credit risk performance through data preprocessing, specifically through feature selection and construction. Because of computing complexity, metaheuristic techniques are often used to raise efficiency. Hybrid feature selection with metaheuristic techniques is usually considered as a combination of filter and wrapper technique to improve learning performance and enhance comprehensibility.

Feature selection can be treated as a combinatorial optimization problem. The binary value of the variable x_i indicates that feature i from N features is present ($x_i = 1$) or absent ($x_i = 0$) in a reduced feature subset. The problem can be represented as follows:

$$\max_{x=(x_1,\cdots x_n)\in\{0;1\}^n} F(x), \tag{3}$$

where $F(x)$ denotes the objective function of the practical problems. In credit risk classification, $F(x)$ specifically represents classification accuracy of default risk. Selecting n appropriate features from a set of N features has been proved to be a NP-hard problem [24]. Metaheuristic techniques are used as an objective optimization method that can realize continuously adaptive optimizing searching, which is capable to balance diversity, accuracy, and efficiency. It is well suitable for solving feature selection optimization problems [25, 26]. Especially, metaheuristic techniques based on swarm intelligence and evolutionary algorithm can better complete continuously adaptive optimizing searching, which has been widely applied in many research fields [27–32]. Therefore, feature selection with metaheuristic techniques, such as genetic algorithm (GA), simulated annealing (SA), and particle swarm optimization (PSO), are commonly used to identify the optimal feature subset and enhance classification accuracy in credit risk assessment [33–38].

It can be seen more and more works incorporate metaheuristic techniques to select the relevant features, rather than enumerating all features. However, there are two main limitations to this kind of approach: trapping in local optimum and being sensitive in initial solution setting. Aiming at the two limitations, an efficient hybrid variable neighborhood search (VNS) and estimation of distribution algorithm (EDA) with the elitist population strategy is proposed, which is expected to reap the benefits of accuracy and simplicity from the traditional feature selection and keep the computational expense down from metaheuristic searching. EDA is a stochastic optimization searching algorithm based on building and sampling explicit probabilistic models of promising candidate solutions [39] and directly uses the objective function as searching message. The key aspect of EDA is that the probability distribution of the solution space is defined explicitly, whereas in most evolutionary algorithms, the distribution is defined implicitly. This explicit utilization in optimization offers some significant advantages over other metaheuristics in evolutionary computation, such as compared with the novel GA, EDA can improve a population of solutions with random sampling from the probability distribution, rather than with random and adaptive searching probabilistic method based on biosphere natural selection and genetic mechanism. GA specifically uses the designed mutation and crossover operator to generate a new adaptive population, while EDA estimates the probability distribution of the solution space and updates the probabilistic model with superior population to generate a new and better candidate population replacing the old population entirely, and the sampling procedure is repeated. Eventually, the optimal solution can be obtained until the termination of iterative constraint is met. Besides, EDA provides explicit evolutionary procedure to present how the problem is solved with a great deal of candidate solutions. Thus, EDA is successfully applied to a large number of combinatorial optimization problems, such as schedule optimization [40], multiobjective knapsack [41], and feature extraction for bioinformatics [42]. In addition, VNS is one of the incremental optimization algorithms developed from local searching techniques that expand searching space of potential solutions. It generally explores specific neighborhood based on the current incumbent solution and searches potential promising neighborhood solutions only if an improvement has been generated [43]. In general, the larger the local searching range is, the greater the chance that a high-quality optimal solution can be obtained. Specific neighborhood structure and moving operators are designed to perform a systematic searching for improving searching capability, which can reduce computational time when expanding the diversity of solutions.

3. VNS-EDA with the Elitist Population Strategy

The proposed technique achieves its objectives by using two main well-established techniques: VNS and EDA. The critical aspects are discussed in this section, including feature coding, solution of the adaptability function of objectives, and incremental searching with the elitist population strategy based on the hybridization of metaheuristic searching.

Machine learning is actually treated as the optimization process of the loss between the predicted and real data. It mainly relies on the optimization algorithms to approach the optimal solution for reaching minimum loss. However, fitting with the large-scale data is required to solve nonconvex problem, and it is always possible to fall into local optimum. Additionally, the evolutionary optimization techniques always begin with a random initial population and then evolve from one generation to another when the evolution stops. Many simulation experiments indicate that it often generates a satisfied solution for middle or small-scale applications within permissive time, not for the large-scale learning problem. Therefore, the proposed technique is based on the following two basic rules: take advantage of high-quality solutions from prior information and achieve a better tradeoff between accuracy and convergence including two aspects: one is neighborhood expansion for local searching, and another one is a probabilistic model for global searching in solution space. The proposed technique combines the good property of local heuristic searching with the elitist population and the good distribution of global heuristic searching in the evolution process.

Specifically, when problem-specific information is available, one party of initial population must be inserted a set of high-quality solutions. Inserting high-quality solutions into the initial solutions that represents the explicit significant features with low redundancy are selected to become a basic component of the population. The remainder of the population is filled by using neighborhood operators that represent the potential significant features. The neighborhood structure is defined as the solution space to keep population diversity, which is very important to avoid premature convergence. The union of the elitist population and the new generation is defined as the predominant population, and it is indispensable to keep dominance and ensure the effectiveness of convergence. Additionally, each searching step will converge to the global optimum using a probabilistic model in the form of a probability vector of the promising solution. Therefore, the dominance and flexibility can be passed on to offspring, and the approximate optimum can be attained.

3.1. Feature Coding and Fitness Function. Feature selection is formulated as the unconstrained optimization problem. Thus, all features are transferred into bit string by binary coding. "One of K" rules is used for representing the feature mask. The bit with value 1 means one feature is selected, and the number 0 indicates this feature is not selected; that is, a list of 0 with N elements representing all features are not selected. If one feature is selected, the value corresponding to the bit variable is set to 1. Thus, the rule defines that when the ith feature of N features is selected, the ith element value is set to 1 in the list, and other unselected feature variables are set 0. Each individual in population represents a selected feature subset.

The global optimum can be reasonably found by giving suitable and comprehensive fitness evaluation function. In order to describe direct acting factors for feature selection in credit risk classification, a fitness evaluation is presented based on the classification accuracy and the number of features as depicted in equation (4). Both are two important factors that directly affect feature selection quality [44]. One predefined weight w_a is for the classification accuracy, and the other w_f is for the proportion of the selected features in all original features. The former suggests the precision and reliability of the model, and the latter reflects the complexity of the model. Generally, the higher the accuracy with the fewer features is, the higher the fitness evaluation value is:

$$\text{fitness} = w_a \times \text{accuarcy} - w_f \times \frac{m}{M}, \tag{4}$$

where accuarcy denotes the classification accuracy predicted by a fixed classifier, m is the number of the selected features, and M is the number of all original features.

3.2. Searching Space Reduction and Refining. As we known, feature selection with metaheuristic techniques typically do not require any information about the problem being solved except for the representation of solution and fitness function. Nonetheless, the drawback is that the global optimum cannot be always guaranteed because of randomicity and premature convergence. The enhancements in the proposed technique are accomplished in two stages for targeting the optimal feature subset: (1) the generation of initial solution incorporating feature correlations and (2) the generation of feature subset solution that is being more likely to the global optimum. The aim of the first stage is preliminary for restricting the solution searching space in a large-scale feature selection. Given that a good problem-specific knowledge essentially and significantly affects the performance, additional prior information can be an available measure for a good and fast selection. The second stage makes the reduced feature subset to be refined by updating probability vector of the predominant population.

In the first stage, the generation of the initial solution is performed to prevent the proposed technique from consuming time in exploring irrelevant features. The redundancy among the features can cause the degradation of classification performance. Starting point has a very important role in the searching space reduction. One technique used to bias the initial population towards good solutions is called seeding work [45, 46] by inserting high-quality solutions into the initial population. Seeding work attempts to achieve a better tradeoff between searching efficiency and convergence by incorporating prior problem-specific information. Without considering any prior information, a large solution space often results in selecting a large number of feature subsets, even can lead to the omission of relevant features. Instead, it can simplify the scale of optimization and reduce the searching scope. For example, the searching space consists of 2^d solutions in CFS, a common greedy algorithm sequential forward selection (SFS) can be used to perform an intelligent searching, and the number of solutions can be dropped to $d(d-1)/2$.

In the second stage, the generation of the feature subset refining represents that the candidates without sacrificing the solution quality can be obtained by iterative searching, based on an efficient fitness evaluation through probability distribution statistics in the restricted scope. It details the union of the elitist and the potential population that may be incorporated into the probabilistic model to speed up the optimization, which can mitigate the effect of the initial choice. The local searching is often applied to a number of randomly generated initial solutions. The diversity given by the neighborhood structure are embedded into the evolutionary process so as to avoid premature convergence and not to neglect potential relevant features. Most of local searching are based on the r-flip neighborhood. In this article, we use 1 and 2-flip neighborhoods considering the computational time. After converting each feature subset to the binary coding population, 1-flip and 2-flip neighborhoods are used for a slightly varied solutions, which represents a permutation with a relatively lower computing time. New populations are attained by removing random parts and generating the replaced parts with sampling from 1-flip and 2-flip. During the processing of neighborhood searching from iteration to iteration, 1-flip neighborhood operator firstly is used to meet the requirement of slightly varied solutions. As the normalized solution space without improving their quality anymore, 2-flip neighborhood operator is performed to explore a slightly larger space. If it is possible to further improve the solution quality, 1-flip neighborhood is again used for iteration or otherwise continues to search in a larger solution space based on 2-flip neighborhood.

Although a slightly neighborhood can help to actualize local searching, the randomness may lead to the weak convergence. Thus, the high-quality solutions can be achieved by the elitist population strategy. The elite reproduction can retain good individuals and guarantee the number stability of the population in each iteration. Increasing the diversity with the elitist population strategy can mitigate risk to trap in local optimum and avoid premature convergence. In the refining process, only the union of the elitist population and the offspring which come from neighborhood structure will appear at the starting of the evolutionary process and act as the reduced feature subset. With this way, these restrictions to the searching space can reduce the irrelevant feature efficiently, and the candidates can be limited in the feasible space. Additionally, the candidates can be represented as probability vectors. The probabilistic model attempts to make the higher probability vectors sampled. The higher probability vector in the probabilistic model can be obtained from individuals with the best values of fitness evaluation, and each generation is updated by superior population, which is more strongly biased towards the global optimum. The influence of the population on the convergence rate to an optimum can be explained by the population evolution with competitive learning. Predominant population always make the probability of generating the global optimum that has increased when the iteration is continued.

More specifically, in each iteration, the candidates are ordered according to their fitness value, and the different ones which have the best fitness are selected to establish the probabilistic model. It is important to note that, sometimes, the number of selected individuals is less than or equal to the population. In either case, the population is supplemented with the best fitness individuals. The probabilistic model is constructed by the predominant variables from the best fitness individuals. The probability distribution of any individual should depend on the joint probability distribution, which defines probabilities of the corresponding variable in each generation. Due to low redundancy among the reduced features, each feature can be generated independently based on the entries in the probability vector. Suppose that the probability vector $p = (p_1, p_2, \ldots, p_n)$ which is sampled to generate new candidate is based on the probabilistic model. A value of 1 is generated in position i in the selected solution with probability p_i. In each generation, the probability vector p_i is updated according to equation (5), and each p_i is set to the proportion of 1s in the new population. Then, most best solutions are selected to give higher probabilities to the promising solutions which represent the samples of the global optimum. Equation (5) is depicted as follows:

$$p_l(x) = p\left(x|D_l^s\right) = \prod_{i=1}^{n} p_l(x_i) = \prod_{i=1}^{n} \frac{\sum_{j=1}^{N} \delta_j\left(X_i = x_i|D_l^s\right)}{N}, \tag{5}$$

where $p_i = \delta_j(X_i = x_i|D_l^s) = \begin{cases} 1 & X_i = x_i \\ 0 & \text{else} \end{cases}$ denotes the probability of generating a value of 1 in position i of solution strings. This repeated refinement of the probabilistic model can keep increasing the probability to generate predominant solutions. After a reasonable number of iteration, the global optimum with high probability would be generated and then reformulated into the feature subsets.

3.3. The General Framework of VNS-EDA Technique. In order to precisely describe the framework of VNS-EDA technique, the main steps are demonstrated, as shown in Figure 1. The detailed explanation is as follows. At first, the original features are extracted by feature redundancy elimination with correlation measurement. Then, the neighborhood operators with the elite reproduction retain the predominant population and guarantee the number stability of the population to get a good population diversity and convergence. The best candidates among the obtained locally optimal solutions are output. Furthermore, the probabilistic model is used to direct its global searching in solution space to jump out of the local optimum. Finally, new high-quality solutions are extracted and fed to a classifier, which is trained offline using labeled training data. The testing data are used to evaluate performance metrics. Thus, the proposed technique helps in reducing the searching region and has a higher effectiveness in reaching the global optimum. Algorithm 1 presents the detail of the proposed technique.

4. Credit Datasets

Two types of datasets are used for performance analysis: (a) the private ZhongAn Credit dataset which is further described in the section and (b) the publicly available German Credit dataset and LendingClub dataset, which have been frequently used as benchmarks in the literatures and in data science competitions.

4.1. ZhongAn Dataset. This private credit dataset is collected from a business loan company called ZhongAn Credit in Shenzhen, China. The company mainly serves SMEs to offer a medium-sized credit loan between CNY50000 and CNY500000. The dataset covered the period from January to December in 2017. The total number of samples is 16900, including 15419 nondefault customers who successfully fulfilled their credit obligations as positive samples and 1481 default customers who were late in performing their obligations as negative samples. It can be seen that the ratio of positive to negative samples is about 10 : 1. It is known that the classifier often breaks down when the size of the training examples per class is not balanced. Thus, the negative number 1481 is multiplied by the empirical member 5, as a reference value for the number of positive samples. Eventually, the 7405 nondefault customers are selected from 15419 nondefault customers as the training samples by the random stratified sampling method. Based on data availability and quality, the list of some typical features with their explanations can be found in Table 1. The expression and meanings of other variables are omitted due to page limit.

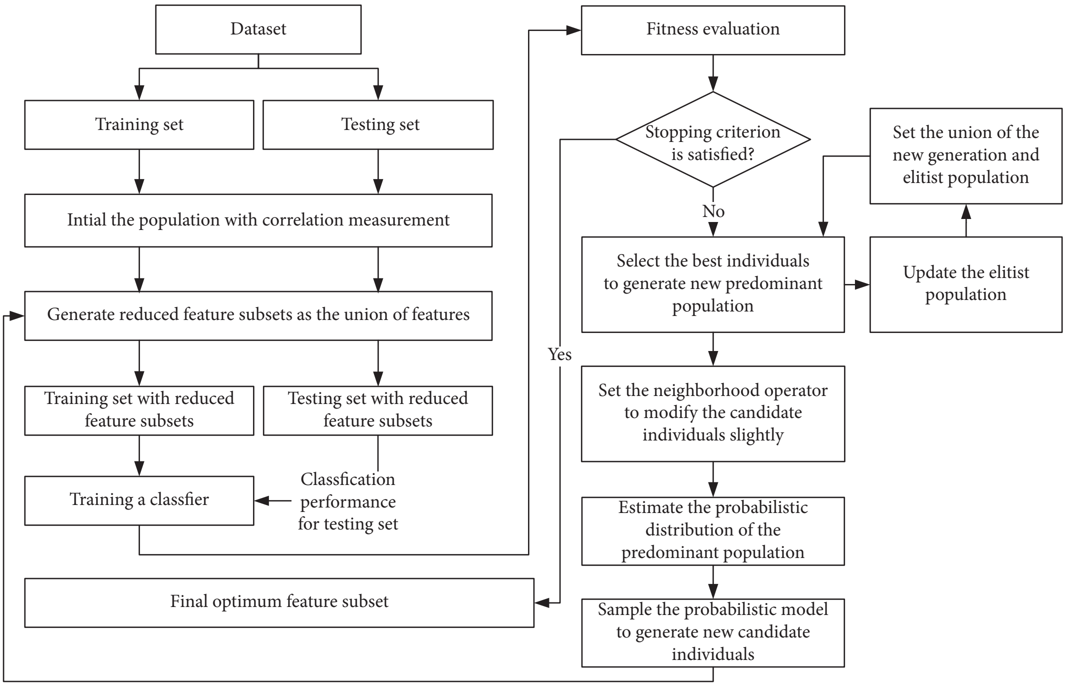

Dataset
Training set
Testing set
Intial the population with correlation measurement
Generate reduced feature subsets as the union of features
Training set with reduced feature subsets
Testing set with reduced feature subsets
Training a classfier
Classfication performance for testing set
Final optimum feature subset
Fitness evaluation
Stopping criterion is satisfied?
No
Yes
Select the best individuals to generate new predominant population
Update the elitist population
Set the union of the new generation and elitist population
Set the neighborhood operator to modify the candidate individuals slightly
Estimate the probabilistic distribution of the predominant population
Sample the probabilistic model to generate new candidate individuals

Figure 1: The general framework of the proposed VNS-EDA feature selection.

```
Input: the original features $D$, population size $M$, predominant population size $N$, weighting parameters $w_a$ and $w_f$,
neighborhood operators one-flip and two-flip, and iterations $n$
Output: the best feature subsets $D'$
begin
generate random $M$ individuals of initial population $P(0)$ according to the values of the symmetrical uncertainty (correlation
measurement between features)
set the neighborhood operators one-flip and two-flip, designate an empty set as the elitist population $S = [\,]$
while (termination criteria not met) do
    compute the fitness of each individuals according to equation (4) in $P(0)$
    $k \longleftarrow 0$
    select the top $N$ promising individuals from $P(0) \cup S$ as a predominant population $P(g)$ and update the elitist population $S(g)$
    with $P(g)$
    use the neighborhood operators to generate new individuals in $P(g)$
    build the probabilistic model $M(g) = P_{l+1}(X)$ from $P(g)$ according to equation (5)
    sample $M(g)$ to generate new candidate individuals as a new generation $P'(g)$
    $k \longleftarrow k + 1$
end while
the process ends when the termination criteria is satisfied
```

Algorithm 1: VNS-EDA with the elitist population strategy.

4.2. Two Public Datasets. German credit dataset is available from the UCI Repository of Machine Learning Databases [47], and it consists of 20 input features with 7 numerical and 13 categorical variables which describe the credit history, loan amounts, loan purposes, personal information, and so on. This original dataset is composed of 700 instances of creditworthy applicants and 300 instances of credit applicants who are default, and it is small scale compared with the other two datasets. The full list of features with their explanations and descriptive statistics can be found in the literature [33].

LendingClub is a US peer-to-peer (P2P) lending company which offers loan trading between \$1000 and \$40000 for high-credit worthy borrowers based on their information including borrower and loan attributes, and the standard loan period is usually three years. This original dataset can be obtained from the official website https://www.lendingclub.com/info/download-data.action. It covered the period from January to June in 2018, and loan status consisted of seven status with "Current", "Issued", "Fully Paid", "In Grace Period", "Late (31–120 days)", "Late (16–30 days)", "Charged Off", and "Default". Due to the object of credit risk classification, "Fully Paid" is considered to be the creditworthy applicants; "In Grace Period", "Late (31–120 days)", "Late (16–30 days)", "Charged Off", and "Default" are treated as

Table 1: Credit risk feature variable.

	Feature variable	Data type	
Repayment capacity	Liquidity ratio	Number	Short-term liquidity
	Quick ratio	Number	
	. . .	. . .	
	Cash ratio	Number	
	Inventory turnover	Number	Operational capability
	Accounts receivable turnover	Number	
	Total asset turnover	Number	
	Cash conversion cycle	Number	
	. . .	. . .	
	Fixed assets turnover	Number	
	Asset profit ratio	Number	Profit-making capability
	Gross profit ratio	Number	
	. . .	. . .	
	Operating profit ratio	Number	
Repayment willing	Age	Number	
	Education	Category	Junior high school, high school, bachelor degree, or above
	Housing	Category	Mortgage loan, mortgage without loan, rental, others
	Gender	Category	Male, female
	. . .	. . .	. . .
	Marital	Category	Married, single, divorced
	Domicile	Category	Residents, nonresidents, others

the bad credit applicants. The total number of samples is 10613. It is composed of 7692 good instances and 2921 bad instances. Because the ratio of good to bad instances is about 3 : 1, the size of the training examples per class can be considered to be balanced. Some of the important features with descriptive statistics are shown in Table 2.

5. Experiments

5.1. Implementation Details. The experiments are conducted to evaluate the proposed and existing techniques for credit risk classification. Data clearing, converting, and sampling are completed before the comparative experiments. Generally, the categorical features are transformed into the numerical ones and converted into binary string, the missing data can be filled with the median amount, and each original feature is linearly scaled to the range [0, 1]. Besides, the class label in all datasets is defined as 0 for nondefault (positive) samples and 1 for default (negative) samples. The size of the examples per class can be considered to reach a balance. Python language is used for exploratory data analysis and engineering algorithms. We build the packing algorithm component and the learning model using Scikit learn with Keras and Deap [48].

The comparisons are drawn between the proposed and three state-of-the-art techniques: population-based incremental learning (PBIL), GA, and PSO. Each generated solution represented by binary data is extracted and converted into different feature subsets. The number of features in all reduced subsets is apparently different. A simple SVM classifier is learned from large amounts of labeled data with the original feature and the reduced feature subsets. Moreover, 10-fold cross-validation has been used to verify the comparative performance metrics, which is widely used to overcome the underfitting and overfitting issue in many literatures [49]. The whole sample on one dataset are divided into three parts: training data (70%), validation data (15%), and testing data (15%).

Moreover, the common parameters are set to the same values. The parameters of the proposed technique are as the following: population size $M = 20$, predominant population size $N = 10$, and weighting parameters $w_a = 0.5$ and $w_f = 0.5$; neighborhood operators are one-flip and two-flip. The unique parameters of the other algorithms are set according to the general empirical values [50]. GA is as following: population size $M = 20$, crossover rate cxpb = 0.5, mutation rate mutp = 0.2, one-point crossover, roulette wheel selection, and elitism replacement. The parameters of PSO are set as follows: population size $N = 20$, inertia weight $w = 1$, acceleration constants $c_1 = 2$ and $c_2 = 2$, and the maximum limited velocity $V_{\max} = 6$. Specially, one incremental univariate EDA is PBIL [51] which is used to perform the probabilistic model, and the probability vector of the best solutions is shifted as follows:

$$p_{l+1}(x) = (1 - \alpha) p_l(x) + \alpha \frac{1}{N} \sum_{k=1}^{N} x_l^k, \qquad (6)$$

where p_l is the probability of generating a 1 in generation l, $x_l^1, x_l^2, \ldots, x_l^N$ denotes the N best individuals, and α is the learning rate.

5.2. Experimental Results and Discussion. In order to evaluate the feasibility and effectiveness of the proposed technique, it is compared with other techniques on all experimental datasets. Table 3 shows the performance metrics of the proposed technique on three datasets, and it lists the accuracy, F-measure, sensitivity, and specificity values, which is quite consistent with training and testing samples. Typically, classification accuracy analysis of the

Table 2: LendingClub dataset with descriptive statistics.

Number	Feature	Statistics	Range
1	Loan_amnt	Avg = 15413	[1000, 40000]
2	Int_rate	Avg = 13.81	[5.31, 30.94]
3	Installment	Avg = 456.51	[30.12, 1556.8]
4	Open_acc	Avg = 11.17	[0, 57]
5	Open_acc_6 m	Avg = 1.07	[0, 12]
6	Total_bal_il	Avg = 36233.93	[0, 923836]
7	. . .	. . .	. . .
8	Open_rv_12 m	Avg = 1.37	[0, 17]
9	Max_bal_bc	Avg = 5255.02	[0, 104320]
10	Acc_open_past_24 mths	Avg = 4.97	[0, 31]
11	Mths_since_recent_inq	Avg = 5.90	[0, 24]
12	Num_bc_tl	Avg = 7.35	[0, 44]
13	Num_tl_op_past_12 m	Avg = 2.36	[0, 18]
14	Total_il_high_credit_limit	Avg = 45162.75	[0, 817057]
15	Annual_inc	Avg = 80672.79	[0, 93000]

Table 3: Performance metrics of VNS-EDA with the optimal feature subset.

Dataset	Accuracy (%)		F-measure (%)		Sensitivity (%)		Specificity (%)	
	Training	Testing	Training	Testing	Training	Testing	Training	Testing
ZhongAn Credit	86.3	85.6	77.2	76.3	61.5	60.1	91.2	90.6
German Bank	76.2	74.5	72.9	71.4	65.6	63.8	87.7	85.1
LendingClub	74.3	75.2	70.6	71.8	60.2	61.5	87.9	89.3

Table 4: Performance results of the comparative method.

Dataset	ZhongAn Credit		German Bank		LendingClub	
	Accuracy (%)	Number of features	Accuracy (%)	Number of features	Accuracy (%)	Number of features
None	75.3	87	70.5	20	68.0	72
GA	82.7	35	74.0	8	72.8	35
PSO	82.9	36	75.5	10	72.7	32
PBIL	83.4	32	73.0	12	73.1	32
Proposed	85.6	30	74.5	10	75.2	28

proposed and other techniques is shown in Table 4 and Figure 2. Similarly, Figures 3–5 depict, respectively, F-measure, sensitivity, and specificity of the proposed and other techniques. We can draw the following observations.

Firstly, not surprisingly, the performance metrics of the original features is the worst. The proposed technique outperforms others in terms of accuracy, F-measure, sensitivity, and specificity on ZhongAn and LendingClub datasets. Additionally, PBIL is slightly better than GP and PSO in these two datasets, and it might be because of its advantages for the searching direction of optimization process in evolutionary computing. Furthermore, the proposed technique generates the fewest reduced features among these comparison techniques, as shown in Table 4. Although the number obtained by PBIL is only a little more than the proposed technique, its performance metrics fall. It appears that PBIL can create new individuals (solutions) in each generation by evolving from fine individuals (solutions) of the previous generation. The solutions, because of the predefined solution searching space, might get into the local

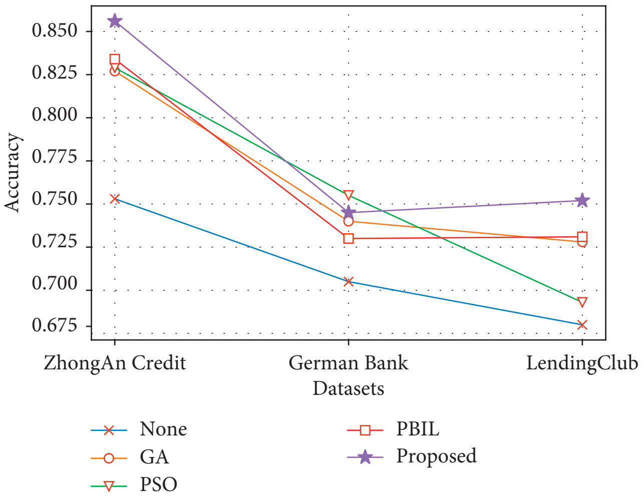


Figure 2: Comparative analysis of accuracy.

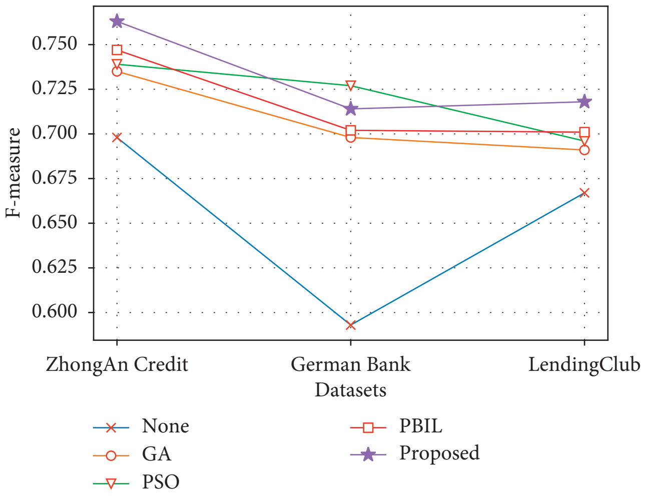


Figure 3: Comparative analysis of F-measure.

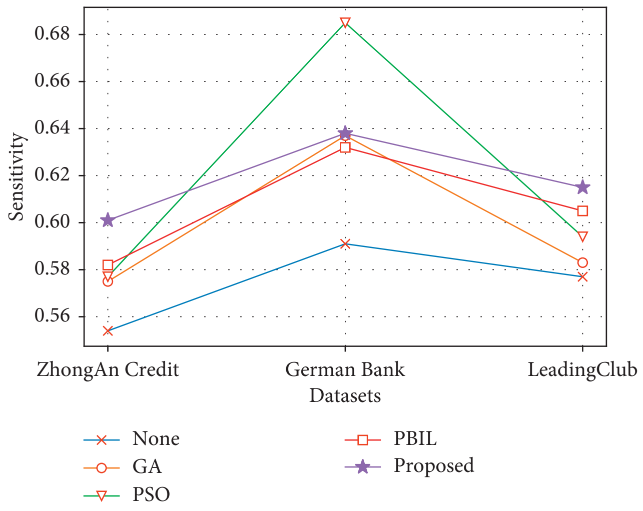


Figure 4: Comparative analysis of sensitivity.

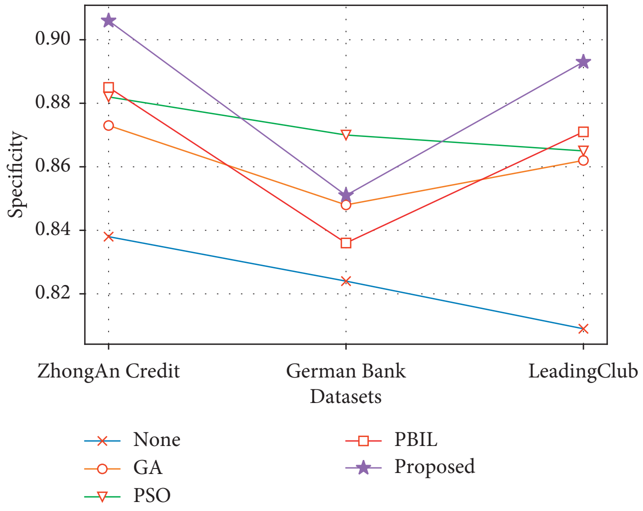


Figure 5: Comparative analysis of specificity.

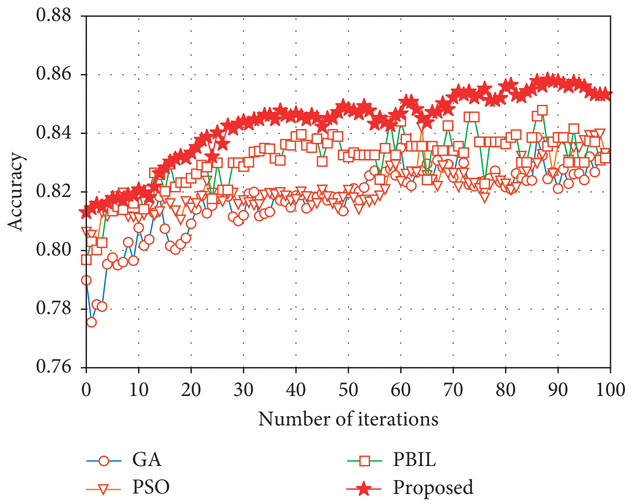


Figure 6: Comparative results of iteration on ZhongAn dataset.

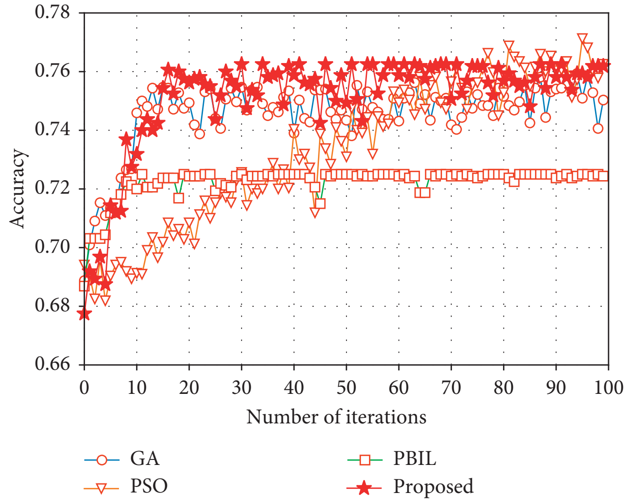


Figure 7: Comparative results of iteration on the German dataset.

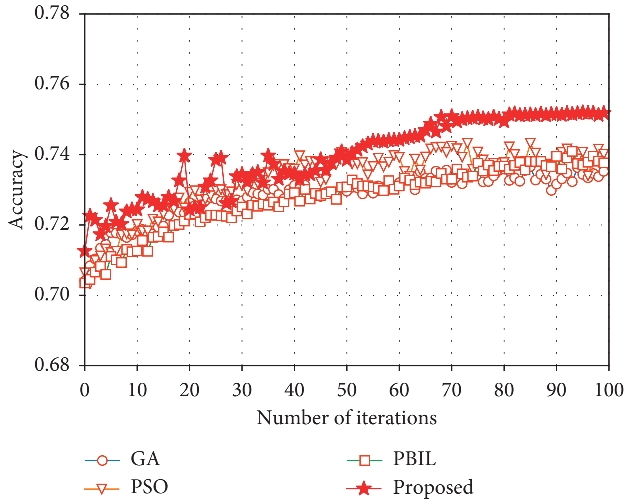
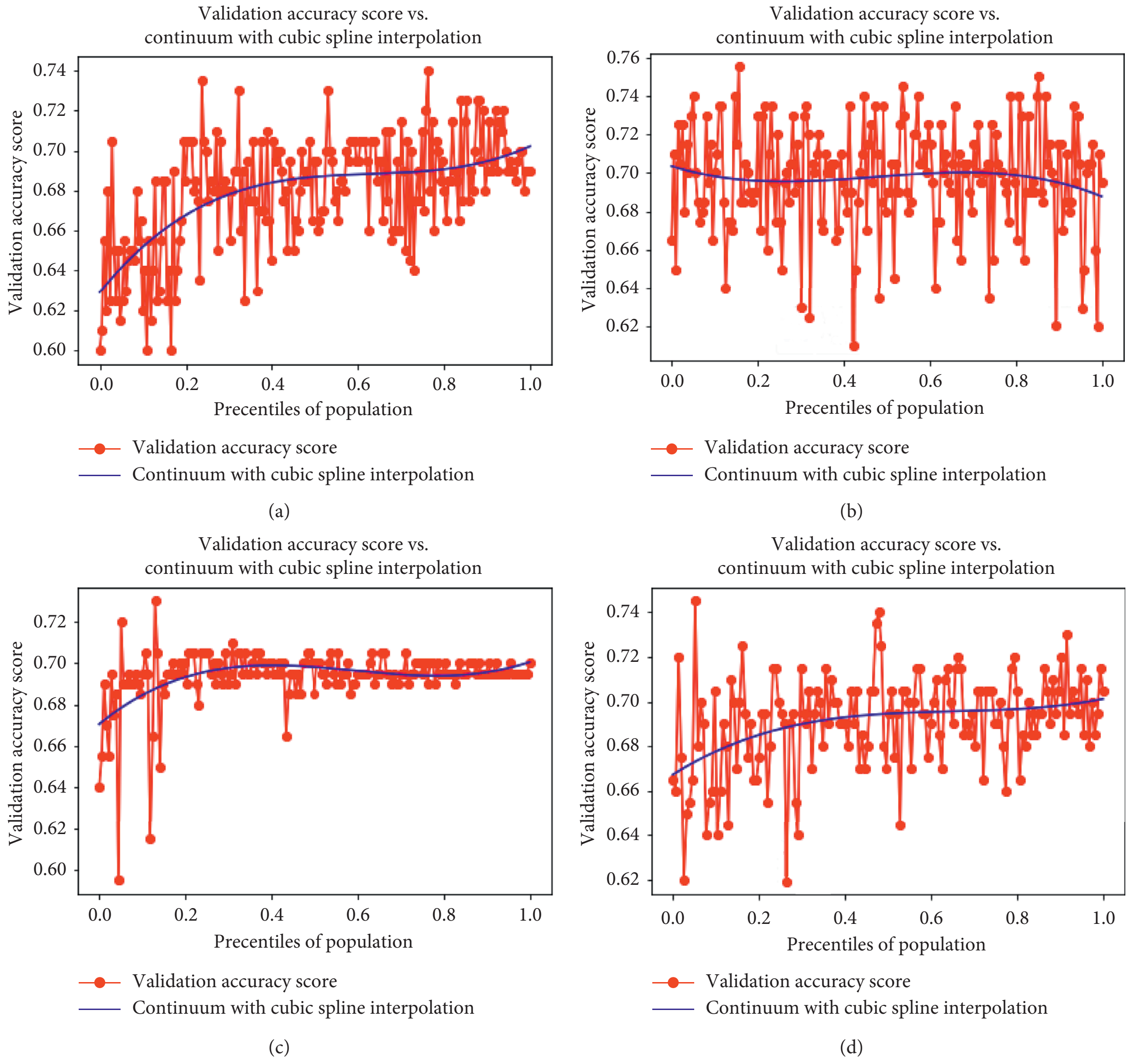


Figure 8: Comparative results of iteration on the LendingClub dataset.



(a)

(b)

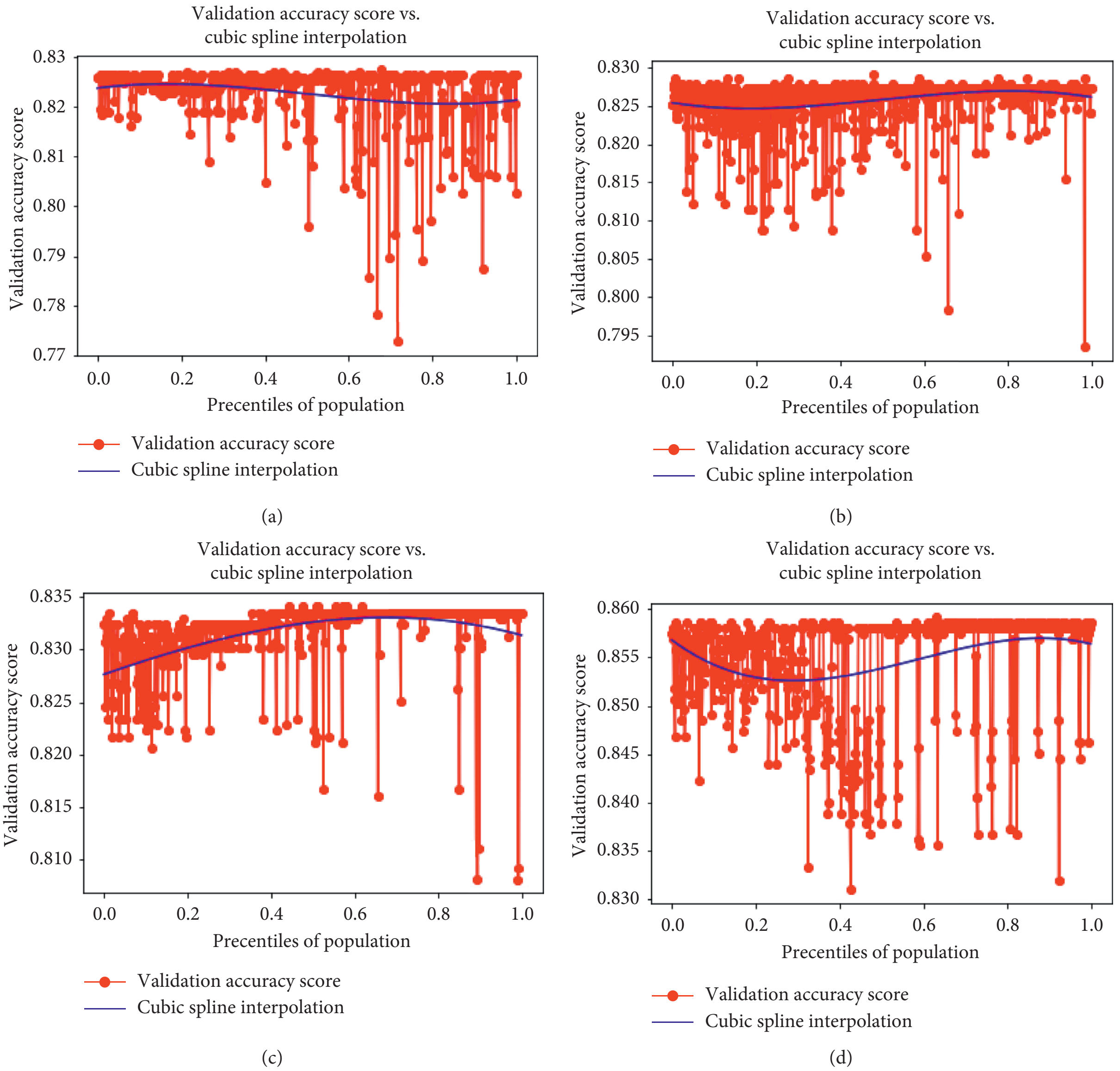
(c)

(d)

Figure 9: Comparative accuracy of all solutions on the German dataset: (a) genetic algorithm; (b) particle swarm optimization; (c) population-based incremental learning; (d) proposed.

optimal solution. However, the proposed technique uses local searching with the elitist population in noncomplete evolutionary process, so as to jump out of the local optimum and reach the global optimum. For German Credit dataset, the results suggest the proposed technique is inefficient. PSO outperforms other techniques in performance metrics although the proposed technique achieved the same 10 reduced features as the number of PSO.

Secondly, the convergent curves of ZhongAn, German, and LendingClub datasets are depicted, respectively, in Figures 6–8, which show accuracy difference in iterative searching. Taking iterative 100 times as the end, the proposed technique obviously outperforms others in searching efficiency. On the ZhongAn dataset, it can be seen the curve of the proposed technique is improved by 4.5% in the first 30 generations and by 0.3% between the 30th and the 100th generations, which indicate that the curve from 30th and 100th is a nearly straight line. However, the difference of GA and PSO remains the same between the 60th and 100th generations, and PBIL is between the 40th and 100th generations, as shown in Figure 6. Moreover, from Figure 7, it is observed that the curve of GA is similar to the proposed technique in convergent trend, and these two techniques have a higher convergence rate compared with PSO. Although they offer a slower convergence rate than PBIL, higher values are achieved. The reason behind this phenomenon may be that PBIL with poor stochastic searching results in not being able to jump out of the local optimum, as can be seen from Figure 9(c). The convergent curves on the LendingClub dataset in Figure 8 demonstrate the value of the proposed technique is entering the phase of saturation after the 60th generation. However, others are nearly identical from the 50th to 100th execution. The trends between the curves closely coincided with the presentation in



(a)

(b)

(c)

(d)

Figure 10: Comparative accuracy of all solutions on the ZhongAn dataset: (a) genetic algorithm; (b) particle swarm optimization; (c) population-based incremental learning; (d) proposed.

Figure 6. Besides, the interesting thing about other techniques is that the curves are almost the same, which is probably caused by the inner properties of this dataset. Based on these results, it reveals that the proposed technique can make a proper tradeoff be realized between diversity and accuracy in iterative searching.

Lastly, the proposed technique is compared with other techniques in terms of the performance generated by all reduced subsets, and the results are shown in Figures 9–11. Each value represents an accuracy obtained by each reduced feature subset. It can be seen that the measuring scales on the Y-axis are different with respect to the anticipated results, and most of values generated by the proposed technique are better than others. The figures also illustrate that the proposed technique can effectively produce more promising solutions and maintain population diversity to prevent premature convergence, except on the German dataset.

All in all, it is found that the ultimate feature number is far less than that of the original feature using the aforementioned metaheuristic techniques. The proposed technique outperforms other techniques on ZhongAn and LendingClub datasets, while it is not supported on the German dataset. It is probably that the effectiveness and reliability of the proposed technique is more significant when dealing with the large-scale data because it is required to use the metaheuristic subspace searching to optimize the ergodicity, but not necessary for small-scale data like German dataset with only 20 original features. The proposed

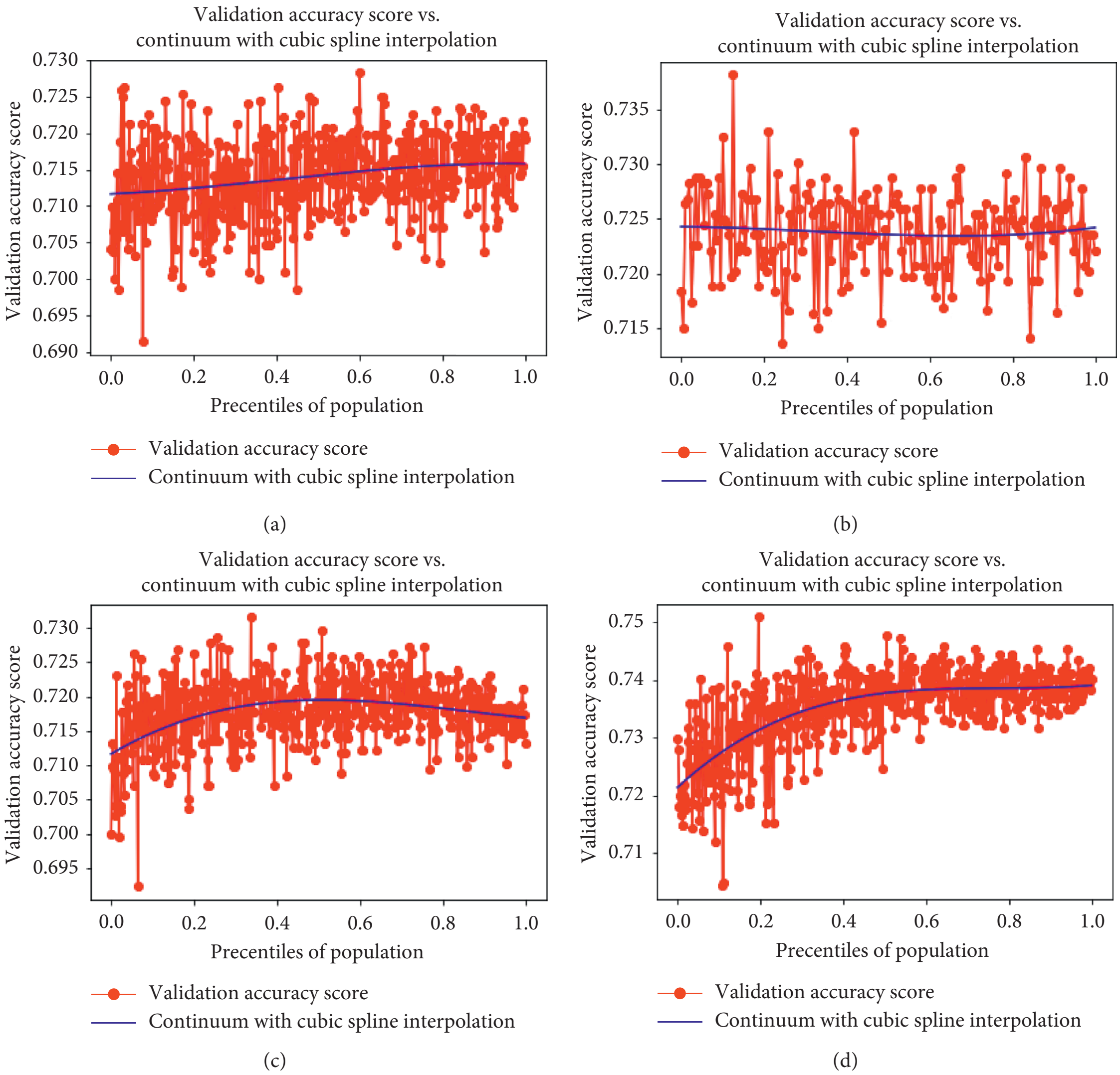


FIGURE 11: Comparative accuracy of all solutions on the LendingClub dataset: (a) genetic algorithm; (b) particle swarm optimization; (c) population-based incremental learning; (d) proposed.

VNS-EDA with the elitist population strategy is well suitable for large-scale credit data with high dimensionality.

6. Conclusion and Future Directions

As data availability is significantly enhanced by big data technology, as well as a significantly reduced profit margin in the credit loan industry, credit risk scoring model should be more accurate and effective. Feature selection for credit features is one of the challenging issues to deal with many irrelevant features. Previous research has shown that it still needs more sophisticated techniques to enhance the accuracy and generalization of risk classification. This article has proposed a hybrid VNS-EDA technique combining feature correlations, multiple neighborhood structures with elitist population strategy, and the probabilistic model. It includes three steps: the original feature preselection, the reduced feature refining, and the reduced feature learning. The original feature preselection can quickly extract relevant features to ameliorate the metaheuristic searching. The reduced feature refining can further generate predominant population with restricted neighborhoods to maintain population diversity and prevent premature convergence. The reduced feature learning builds an accurate classifier. The proposed technique has been tested on three credit datasets to evaluate its effectiveness. Comparisons have been drawn between the proposed and three state-of-the-art techniques by considering classification performance metrics. The mean improvement in terms of accuracy, F-measure, sensitivity, and specificity is found to be 2.8%, 2.1%, 1.9%, and 2.6% on the ZhongAn dataset, respectively. It is consistent with the results of improvement which are, respectively, 3.4%, 2.2%, 2.1%, and 2.7% obtained by the LendingClub dataset. Therefore, the proposed technique is more efficient in large-scale credit data.

Future directions of the proposed technique are discussed as follows: (1) the proposed technique can be also utilized to remove the irrelevant risk features when handling a large dimensionality number of samples for SMEs and individual financing. (2) The initial hyperparameters affect the performance. The prediction results can be further optimized to improve model quality by an efficient tuning of the hyperparameters. (3) This feature selection technique is based on subspace searching. Therefore, in the near future, we can make use of advanced space reduction techniques to enhance the performance and reduce the execution time such as the combination of generative adversarial networks (GANs) with metaheuristic techniques. The population evolution can be implemented by the operator competition or cooperation through GAN with the dynamic expanding neighborhoods and be realized as a learning process with their own evolution, which is expected to obtain an approximate optimum as quickly as possible.

Data Availability

The private experimental data used to support the findings of this study have not been made available because these data belong to a private financial company, while the public experimental data are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (no. 71721001) and in part by the Natural Science Research Team of Guangdong Province of China (no. 2014A030312003).

References

[1] M. Šušteršič, D. Mramor, and J. Zupan, "Consumer credit scoring models with limited data," *Expert Systems with Applications*, vol. 36, no. 3, pp. 4736–4744, 2009.

[2] A. E. Khandani, A. J. Kim, and A. W. Lo, "Consumer credit-risk models via machine-learning algorithms," *Journal of Banking & Finance*, vol. 34, no. 11, pp. 2767–2787, 2010.

[3] F. Louzada, A. Ara, and G. B. Fernandes, "Classification methods applied to credit scoring: systematic review and overall comparison," *Surveys in Operations Research and Management Science*, vol. 21, no. 2, pp. 117–134, 2016.

[4] S. Bhatia, P. Sharma, R. Burman, S. Hazari, and R. Hande, "Credit scoring using machine learning techniques," *International Journal of Computer Applications*, vol. 161, no. 11, pp. 1–4, 2017.

[5] I. Guyon and A. Elisseeff, "An introduction to feature extraction," in *Feature Extraction*, pp. 1–25, Springer, Berlin, Germany, 2006.

[6] J. Tang, S. Alelyani, and H. Liu, "Feature selection for classification: a review," in *Data Classification: Algorithms and Applications*, pp. 37–64, CRC Press, Boca Raton, FL, USA, 2014.

[7] S. Khalid, T. Khalil, and S. Nasreen, "A survey of feature selection and feature extraction techniques in machine learning," in *Proceedings of the IEEE Science and Information Conference*, pp. 372–378, London, UK, August 2014.

[8] R. Sandy, "Survey on dimension reduction techniques," *Journal of Computer Applications*, vol. 8, no. 5, pp. 704–710, 2006.

[9] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.

[10] A. I. Hall and L. A. Smith, "Feature selection for machine learning: comparing a correlation-based filter approach to the wrapper," in *Proceedings of the FLAIRS Conference*, pp. 235–239, Orlando, FL, USA, May 1999.

[11] P. Somol, B. Baesens, P. Pudil, and J. Vanthienen, "Filter-versus wrapper-based feature selection for credit scoring," *International Journal of Intelligent Systems*, vol. 20, no. 10, pp. 985–999, 2005.

[12] A. G. Karegowda, A. S. Manjunath, and M. A. Jayaram, "Feature subset selection problem using wrapper approach in supervised learning," *International Journal of Computer Applications*, vol. 1, no. 7, pp. 13–17, 2010.

[13] P. Danenas, G. Garsva, and S. Gudas, "Credit risk evaluation model development using support vector based classifiers," *Procedia Computer Science*, vol. 4, pp. 1699–1707, 2011.

[14] C. Ding and H. Peng, "Minimum redundancy feature selection from microarray gene expression data," *Journal of Bioinformatics and Computational Biology*, vol. 3, no. 2, pp. 185–205, 2005.

[15] Y. Liu and M. Schumann, "Data mining feature selection for credit scoring models," *Journal of the Operational Research Society*, vol. 56, no. 9, pp. 1099–1108, 2005.

[16] H. Yu, X. Huang, X. Hu, and H. Cai, "A comparative study on data mining algorithms for individual credit risk evaluation," in *Proceedings of the IEEE International Conference on Management of e-Commerce and e-Government*, pp. 35–38, Chengdu, China, October 2010.

[17] F. N. Koutanaei, H. Sajedi, and M. Khanbabaei, "A hybrid data mining model of feature selection algorithms and ensemble learning classifiers for credit scoring," *Journal of Retailing and Consumer Services*, vol. 27, pp. 11–23, 2015.

[18] C. Dhaenens and L. Jourdan, *Metaheuristics for Big Data*, Wiley, Hoboken, NJ, USA, 2016.

[19] L. Yu and H. Liu, "Efficient feature selection via analysis of relevance and redundancy," *Journal of Machine Learning Research*, vol. 5, pp. 1205–1224, 2004.

[20] M. A. Hall, *Correlation-based feature subset selection for machine learning*, Ph.D. thesis, University of Waikato, Hamilton, New Zealand, 1999.

[21] M. A. Hall and G. Holmes, "Benchmarking attribute selection techniques for discrete class data mining," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 6, pp. 1437–1447, 2012.

[22] D. Boughaci and A. A. K. Alkhawaldeh, "A new variable selection method applied to credit scoring," *Algorithmic Finance*, vol. 7, no. 1-2, pp. 43–52, 2018.

[23] S. Piramuthu, "On preprocessing data for financial credit risk evaluation," *Expert Systems with Applications*, vol. 30, no. 3, pp. 489–497, 2006.

[24] P. M. Narendra and K. Fukunaga, "A branch and bound algorithm for feature subset selection," *IEEE Transactions on Computers*, vol. C-26, no. 9, pp. 917–922, 1977.

[25] L. Wang, H. Ni, R. Yang, V. Pappu, M. B. Fenn, and P. M. Pardalos, "Feature selection based on meta-heuristics for biomedicine," *Optimization Methods and Software*, vol. 29, no. 4, pp. 703–719, 2014.

[26] R. Diao and Q. Shen, "Nature inspired feature selection meta-heuristics," *Artificial Intelligence Review*, vol. 44, no. 3, pp. 311–340, 2015.

[27] I. S. Oh, J. S. Lee, and B. R. Moon, "Hybrid genetic algorithms for feature selection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 11, pp. 1424–1437, 2004.

[28] Y. Marinakis and M. Magdalene, "A hybridized particle swarm optimization with expanding neighborhood topology for the feature selection problem," in *Hybrid Metaheuristics*, pp. 37–51, Springer, Berlin, Germany, 2013.

[29] H. Banka and S. Dara, "A Hamming distance based binary particle swarm optimization (HDBPSO) algorithm for high dimensional feature selection, classification and validation," *Pattern Recognition Letters*, vol. 52, pp. 94–100, 2015.

[30] M. M. Mafarja and S. Mirjalili, "Hybrid whale optimization algorithm with simulated annealing for feature selection," *Neurocomputing*, vol. 260, pp. 302–312, 2017.

[31] U. Singh and S. N. Singh, "Optimal feature selection via NSGA-II for power quality disturbances classification," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 7, pp. 2994–3002, 2018.

[32] H. S. Pannu, D. Singh, and A. K. Malhi, "Multi-objective particle swarm optimization-based adaptive neuro-fuzzy inference system for benzene monitoring," *Neural Computing and Applications*, vol. 31, no. 7, pp. 2195–2205, 2017.

[33] S. Oreski and G. Oreski, "Genetic algorithm-based heuristic for feature selection in credit risk assessment," *Expert Systems with Applications*, vol. 41, no. 4, pp. 2052–2064, 2014.

[34] J. Zhou and T. Bai, "Credit risk assessment using rough set theory and GA-based SVM," in *Proceedings of the 3rd International Conference on Grid and Pervasive Computing-Workshops*, pp. 320–325, Kunming, China, May 2008.

[35] Y. Marinakis, M. Marinaki, M. Doumpos, N. Matsatsinis, and C. Zopounidis, "Optimization of nearest neighbor classifiers via metaheuristic algorithms for credit risk assessment," *Journal of Global Optimization*, vol. 42, no. 2, pp. 279–293, 2008.

[36] C.-M. Wang and Y.-F. Huang, "Evolutionary-based feature selection approaches with new criteria for data mining: a case study of credit approval data," *Expert Systems with Applications*, vol. 36, no. 3, pp. 5900–5908, 2009.

[37] J. Wang, A.-R. Hedar, S. Wang, and J. Ma, "Rough set and scatter search metaheuristic based feature selection for credit scoring," *Expert Systems with Applications*, vol. 39, no. 6, pp. 6123–6128, 2012.

[38] H. Altinbas and G. C. Akkaya, "Improving the performance of statistical learning methods with a combined meta-heuristic for consumer credit risk assessment," *Risk Management*, vol. 19, no. 4, pp. 1–26, 2017.

[39] M. Hauschild and M. Pelikan, "An introduction and survey of estimation of distribution algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 3, pp. 111–128, 2011.

[40] U. Aickelin and J. Li, "An estimation of distribution algorithm for nurse scheduling," *Annals of Operations Research*, vol. 155, no. 1, pp. 289–309, 2007.

[41] R. Shah and P. Reed, "Comparative analysis of multiobjective evolutionary algorithms for random and correlated instances of multiobjective d-dimensional knapsack problems," *European Journal of Operational Research*, vol. 211, no. 3, pp. 466–479, 2011.

[42] R. ArmaPñanzas, I. Inza, R. Santana et al., "A review of estimation of distribution algorithms in bioinformatics," *BioData Mining*, vol. 1, pp. 1–6, 2008.

[43] P. Hansen and N. Mladenoviú, "Variable neighborhood search: principles and applications," *European Journal of Operational Research*, vol. 130, no. 3, pp. 449–467, 2008.

[44] A. Janecek, W. Gansterer, M. Demel, and G. Ecker, "On the relationship between feature selection and classification accuracy," in *Proceedings of the Workshop on New Challenges for Feature Selection in Data Mining and Knowledge Discovery*, pp. 90–105, Antwerp, Belgium, 2008.

[45] J. Schwarz and J. Ocenasek, "A problem knowledge-based evolutionary algorithm KBOA for hypergraph bisectioning," in *Proceedings of the 4th Joint Conference on Knowledge-Based Software Engineering*, pp. 51–58, Brno, Czech Republic, 2000.

[46] M. Pelikan and D. E. Goldberg, "Hierarchical BOA solves Ising spin glasses and MAXSAT," in *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 1275–1286, Chicago, IL, USA, July 2003.

[47] K. Bache and M. Lichman, *UCI Machine Learning Repository*, University of California, School of Information and Computer Science, Irvine, CA, USA, 2013, http://archive.ics.uci.edu/ml.

[48] F. Fortin, F. Rainville, M. Gardner, M. Parizeau, and C. Gagné, "DEAP: evolutionary algorithms made easy," *Journal of Machine Learning Research*, vol. 13, pp. 2171–2175, 2012.

[49] A. Kalousis, J. Prados, and M. Hilario, "Stability of feature selection algorithms: a study on high-dimensional spaces," *Knowledge and Information Systems*, vol. 12, no. 1, pp. 95–116, 2007.

[50] J. Hamon, "Combinatorial optimization for variable selection in high dimensional regression: application in animal genetic," Thesis, University of Science and Technology, Lille, France, 2013.

[51] S. Baluja, "Population-based incremental learning. a method for integrating genetic search based function optimization and competitive learning," Technical report, Carnegie Mellon University, Pittsburgh, PA, USA, 1999.

Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 1957812, 17 pages
https://doi.org/10.1155/2020/1957812

Research Article

Sequential Hybrid Particle Swarm Optimization and Gravitational Search Algorithm with Dependent Random Coefficients

Shanhe Jiang, Chaolong Zhang, and Shijun Chen

Department of Physics and Power Engineering, Anqing Normal University, Anqing 246011, China

Correspondence should be addressed to Shanhe Jiang; jshxlxlw@163.com

Received 30 December 2019; Revised 1 March 2020; Accepted 30 March 2020; Published 21 April 2020

Guest Editor: Dr. Dilbag Singh

Particle swarm optimization (PSO) has been proven to show good performance for solving various optimization problems. However, it tends to suffer from premature stagnation and loses exploration ability in the later evolution period when solving complex problems. This paper presents a sequential hybrid particle swarm optimization and gravitational search algorithm with dependent random coefficients called HPSO-GSA, which first incorporates the gravitational search algorithm (GSA) with the PSO by means of a sequential operating mode and then adopts three learning strategies in the hybridization process to overcome the aforementioned problem. Specifically, the particles in the HPSO-GSA enter into the PSO stage and update their velocities by adopting the dependent random coefficients strategy to enhance the exploration ability. Then, the GSA is incorporated into the PSO by using fixed iteration interval cycle or adaptive evolution stagnation cycle strategies when the swarm drops into local optimum and fails to improve their fitness. To evaluate the effectiveness and feasibility of the proposed HPSO-GSA, the simulations were conducted on benchmark test functions. The results reveal that the HPSO-GSA exhibits superior performance in terms of accuracy, reliability, and efficiency compared to PSO, GSA, and other recently developed hybrid variants.

1. Introduction

As many real-world optimization problems become increasingly complex, traditional optimization algorithms cannot sufficiently satisfy the problem requirements and better effective optimization algorithms are needed. Hence, various kinds of metaheuristic algorithms that are inspired by natural phenomena have launched into a center stage in recent decades for solving complex optimization problems. Genetic algorithm (GA) [1], particle swarm optimization (PSO) [2], artificial immune system (AIS) [3], differential evolution (DE) [4], ant colony optimization (ACO) [5], glowworm swarm optimization (GSO) [6], artificial bee colony (ABC) [7], gravitational search algorithm (GSA) [8], grey wolf optimization (GWO) [9], cat swarm optimization (CSO) [10], harmony search algorithm (HS) [11], and bacterial foraging optimization algorithm (BFOA) [12] have been developed in recent years by researchers and have shown superior performance for solving a wide range of optimization problems, such as function optimization [4–9, 11–15], fuzzy inference system [16, 17], image processing [18, 19], economic dispatch [20, 21], and neural networks training [22, 23].

Overall, the review of the presented literature states that there is no single superior method for solving optimization problems. That is, no one algorithm can solve all of the optimization problems, but each algorithm can solve a special class of problems. Although the aforementioned algorithms that have been proposed to solve optimization problems do achieve good performance, there are still undesirable shortcomings. For instance, PSO often suffers from premature convergence, whereas it tends to be trapped into local optima due to the rapid convergence speed [24]. GSA requires a long computation time to find the solution for some problems [21]. Hence, there is a lot of room for improvement in finding the better optimization algorithm.

Another issue is how to balance the exploration/exploitation search ability for a single metaheuristic algorithm including PSO or GSA. The key operation of metaheuristic optimization algorithms is how to keep a better trade-off between exploitation and exploration abilities in the searching process. A good algorithm should have the capability of these two abilities to seek the global optimal solution. However, some algorithms present more outstanding advantage on one of these abilities. For instance, PSO has a tendency to rapid convergence in a multivariable optimization problem. By comparison, GSAs global exploration performance is particularly conspicuous. Hence, PSO and GSA approaches possess respective advantages and potentialities. It encourages us to develop an appropriate hybridization technique of different metaheuristic algorithms to mitigate the weakness of the original algorithm and obtain the outstanding optimization performance against a single algorithm, thereby acquiring rapid response and avoiding premature convergence.

Inspired by abovementioned ideas, to further improve the respective drawbacks of PSO and GSA, a novel combination strategy that integrates PSO with GSA, sequential hybrid particle swarm optimization and gravitational search algorithm with dependent random coefficients called HPSO-GSA, is proposed in this paper based on a sequential hybrid pattern. To be specific, we first propose a new velocity updating equation for PSO based on dependent random coefficients strategy to enhance the balance between exploitation and exploration search. Second, the existing PSO evolution framework is improved, and the GSA is incorporated into the PSO by using a sequential mode when the swarm drops into local optimum and fails to improve their fitness. That is, the HPSO-GSA first enters into the PSO phase to update its velocity and position. Then, the GSA operator is carried out on condition that the fixed iteration interval cycle or adaptive evolution stagnation cycle strategies are met in the process of evolution. Finally, the performance of the proposed algorithm is evaluated against PSO, GSA, and state-of-the-art hybrid variants by using a set of benchmark test functions. The results reveal that the proposed HPSO-GSA can achieve better optimization performance compared with the involved algorithms.

The remainder of this paper is organized as follows. A brief review of related works on PSO and GSA is given in Section 2. Section 3 introduces the proposed HPSO-GSA approach. In Section 4, the experiments, comparisons, and discussion for the used benchmark test problems are carried out to evaluate the performance of the proposed algorithm. Finally, the conclusion and future work are given in Section 5.

2. Related Works

In this section, we first introduce the relevant backgrounds including the PSO and GSA algorithms. Next, the state-of-the-art PSO and GSA hybrid variants are reviewed.

2.1. Particle Swarm Optimization (PSO). PSO is a population-based metaheuristic optimization method, which was originally introduced by Kennedy and Eberhart [2]. Since its development, PSO has become one of the most promising optimizing techniques for solving global optimization problems. The algorithm is motivated by intelligent collective behavior like the movement of a flock of birds, a school of fish, and a group of ants to seek for foods. Compared to other optimization techniques, PSO is easy to implement with few parameters to adjust and is computationally inexpensive. PSO does not require any gradient information from the objective functions, and it uses only primitive mathematical operators through the exchange of information among candidate individuals (particles), in order to attain desirable optimization performance. In the past decades, the PSO has been shown to successfully solve a wide range of optimization fields. It achieves better results more speedily and more cheaply than other methods, such as GA, DE, and ACO [25].

Suppose that each particle which represents a potential solution of the problem in the population size N flies through a D-dimensional search space. It is associated with two vectors, namely, a position vector $x_i(t) = [x_i^1(t), x_i^2(t), \ldots, x_i^D(t)]$ and a velocity vector $v_i(t) = [v_i^1(t), v_i^2(t), \ldots, v_i^D(t)]$ for the current iteration t. The personal best experience of the ith particle is represented by $p_i(t) = [p_i^1(t), p_i^2(t), \ldots, p_i^D(t)]$, and the best global position of the swarm found so far is stored in $p_g(t) = [p_g^1(t), p_g^2(t), \ldots, p_g^D(t)]$. The initial position for each particle $x_i(0) = [x_i^1(0), x_i^2(0), \ldots, x_i^D(0)]$ in the population is randomly generated from in the range of the decision space of the problem. The initial velocity $v_i(0) = [v_i^1(0), v_i^2(0), \ldots, v_i^D(0)]$ for each dimension is set as a zero vector. Then, the particle's velocity and its new position are updated by the following equations:

$$\begin{aligned} v_i^d(t+1) = {} & w(t)v_i^d(t) + c_1(t)r_1(t)\left(p_i^d(t) - x_i^d(t)\right) \\ & + c_2(t)r_2(t)\left(p_g^d(t) - x_i^d(t)\right), \end{aligned} \tag{1}$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1), \tag{2}$$

where $d \in \{1, 2, \ldots, D\}, i \in \{1, 2, \ldots, N\}.r_1(t)$ and $r_2(t)$ are two mutually independent random coefficients drawn from uniform distributed within the range $[0, 1]$. $w(t)$ is the linear time-varying inertia weight (TVIW) factor within the interval $[0.4, 0.9]$ suggested by Shi and Eberhart [26]. Its goal is to control the impact of the previous velocity on the current velocity, thereby influencing the trade-off between global exploration and local search of the particles. $c_1(t)$ and $c_2(t)$ are two linear time-varying acceleration coefficients (TVAC) suggested by Ratnaweera et al. [27]. Their objectives are to control the influence of personal and swarm best experiences, respectively.

2.2. Gravitational Search Algorithm (GSA). GSA inspired by the Newton law of gravity and mass interactions is a newly developed swarm-based metaheuristic optimization method [8]. In GSA, all agents are regarded as objects including different masses, and their performances are evaluated by their masses using a fitness function. Each agent attracts each other agent through the gravity force which is directly proportional to the product of their masses and inversely proportional to the square of the distance between them. This force leads to global movement of all agents towards heavier masses with the aid of the Newtonian law of motion. The heavier agent that corresponds to better solution to a problem moves more slowly than the lighter one. Then, it is concluded that masses should be attracted by the heaviest mass which represents an optimum solution in the search space by lapse of time.

Let us consider a population including N agents (masses) in a D-dimensional decision space, and the position vector of the ith agent is described as $x_i(t) = [x_i^1(t), x_i^2(t), \ldots, x_i^D(t)]$ at iteration t, where $x_i^d(t)$ represents the position of the ith agent in the dth dimension at iteration t. The algorithm initializes the N agents randomly in the given decision space. During the evolution process, the gravitational force acting on agent i from agent j at iteration t is defined as (3) and the whole force that acts on agent i in the dth dimension with randomly weighted sum of the dth component of the forces coming from other agents are given as (4):

$$F_{ij}^d(t) = G(t)\frac{M_{pi}(t) \times M_{aj}(t)}{R_{ij}(t) + \varepsilon}\left(x_j^d(t) - x_i^d(t)\right), \tag{3}$$

$$F_i^d(t) = \sum_{j \in \text{kbest}, j \neq i} \text{rand}_j \times F_{ij}^d(t), \tag{4}$$

where $M_{aj}(t)$ represents the active gravitational mass related to agent j and $M_{pi}(t)$ represents the passive gravitational mass related to agent i. $G(t)$ is the gravitational constant defined by (5). ε is a small constant value. $R_{ij}(t)$ is the Euclidian distance between agents i and j defined by (6). rand_j is uniform random number in the interval [0, 1]. kbest represents the set of first K agents with the best fitness value and the biggest mass, which is defined as a function of time with the original value K_0 at the beginning, and it is linearly decreased to 1 by lapse of iteration. Finally, there will be just one agent applying force to the others:

$$G(t) = G_0 \exp\left(-\alpha \frac{t}{t_{\max}}\right), \tag{5}$$

$$R_{ij}(t) = \left\| x_i(t), x_j(t) \right\|. \tag{6}$$

According to the Newton law of motion, the acceleration of an agent is proportional to the resultant force and inverse of its mass, so the acceleration of the ith agent in dth dimension at iteration t is denoted as follows:

$$a_i^d(t) = \frac{F_i^d(t)}{M_{ii}(t)}. \tag{7}$$

The next velocity of agent i is defined as a fraction of its current velocity added to its acceleration. Sequentially, its next position is calculated based on the corresponding velocity:

$$\begin{aligned} v_i^d(t+1) &= \text{rand}_i v_i^d(t) + a_i^d(t), \\ x_i^d(t+1) &= x_i^d(t) + v_i^d(t+1). \end{aligned} \tag{8}$$

The mass of agent i is calculated by (9), and the normalization of the calculated mass is given as (10):

$$m_i(t) = \frac{\text{fit}_i(t) - \text{worst}(t)}{\text{best}(t) - \text{worst}(t)}, \tag{9}$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)}, \tag{10}$$

$$M_{ai} = M_{pi} = M_{ii} = M_i, \quad i = 1, 2, \ldots, N, \tag{11}$$

where $\text{fit}_i(t)$ represents the fitness value of agent i at iteration t and best (t) and worst (t) represent the best and worst fitness value in the current population at iteration t, respectively. The gravitational mass $M_i(t)$ represents the mass of agent i at iteration t which embodies the fitness evaluation value of agent i.

For a minimization optimization problem, best (t) and worst (t) are defined as follows:

$$\begin{aligned} \text{best}(t) &= \min_{i \in \{1,\ldots,N\}} \text{fit}_i(t), \\ \text{worst}(t) &= \max_{i \in \{1,\ldots,N\}} \text{fit}_i(t). \end{aligned} \tag{12}$$

2.3. State-of-the-Art PSO and GSA Hybrid Variants. Extensive amounts of works have been performed to improve the original PSO and GSA performance. Among these works, studies on hybrid systems that combine skilled metaheuristic optimization algorithms to obtain good compromise between exploration and exploitation have gained extensive popularity. The most classical PSO variants have been reported in [14, 15, 23, 28–36]. Kao and Zahara [28] proposed the hybridization strategy of PSO and GA (GAPSO) for solving multimodal test functions. In GAPSO, individuals in a new generation are drawn not only from crossover and mutation operation in GA but also from movement mechanism in PSO. The results show the superiority of the hybrid GAPSO approach in terms of solution quality and convergence speed. Esmin et al. [29] introduced a PSO algorithm coupled with GA mutation operator, namely, HPSOM, for solving unconstrained global optimization problems. Shunmugalatha and Slochanal [30] proposed a hybrid particle swarm optimization (HPSO), which incorporated the crossover, mutation operators, and subpopulation process in the genetic algorithm into particle swarm optimization. The implementation of HPSO on test functions shows that it converges to better solution much faster. Zhang and Xie [31] introduced a hybrid particle swarm optimization with a differential evolution operator (DEPSO), which provides the bell-shaped mutation with consensus on the population diversity along with the evolution. A set of benchmark functions was applied to evaluate its efficiency. A hybrid algorithm named DE-PSO was proposed by Zhang et al. [32], which incorporates concepts from DE and PSO, updating particles not only by DE operators but also by mechanisms of PSO. The proposed algorithm was tested on several benchmark functions. Liu et al. [14] proposed a novel hybrid algorithm named PSODE, where DE is incorporated to update the previous best positions of PSO particles to force them to jump out of local attractor in order to prevent stagnation of population. Besides GA and DE, PSO has been hybridized with extremal optimization (EO) [15], central force optimization (CFO) [23], estimation of distribution algorithm (EDA) [33], artificial immune system (AIS) [34], gravitational search algorithm [35], and teaching-learning-based optimization (TLBO) [36]. Overall, these PSO-based hybrid variants have been successfully utilized for solving global optimization problems.

Similarly, a novel hybrid version of GSA variants has also been reported in [18, 22, 37–39]. For instance, Mirjalili et al. [13, 22] proposed a novel hybrid PSOGSA algorithm by adopting a parallel model for solving benchmark function optimization and feedforward neural networks. A hybrid approach that integrated differential evolution into gravitational search algorithm (DE-GSA) for unconstrained optimization was introduced by Li et al. [37]. Chen et al. [38] proposed a hybrid gravitational search algorithm combined with simulated annealing (GSA-SA) for the traveling salesman problem. A new hybrid approach, namely, genetic algorithm-based gravitational search algorithm (GA-GSA), was proposed to solve image segmentation [18]. A novel GSA-SVM hybrid system which hybridizes the GSA with support vector machine (SVM) was proposed to improve classification accuracy with an appropriate feature subset in binary problems [39]. Apparently, these hybrid systems of GSA have demonstrated powerful results when compared with other approaches such as the original GSA itself, DE, GA, and PSO.

2.4. Comparison of Particle Swarm Optimization and Gravitational Search Algorithm. To thoroughly understand the two metaheuristic optimization methods, we have identified three similarities and four differences between the PSO and the GSA. The similarities are as follows: (1) both are population-based metaheuristic algorithms; (2) particles/agent positions are updated by iteration; and (3) both algorithms use velocity formulations for position updating. On the other hand, they differ in the following aspects: (1) PSO simulates the social behavior of birds, whereas GSA was inspired by a physical phenomenon; (2) PSO employs fitness values for the two best positions pbest and gbest, while GSA uses fitness values to calculate masses that are proportional to gravitational forces; (3) PSO particles update their positions by means of dynamic velocities with cognitive and social behaviors, while GSA agents calculate their positions using changing accelerations with the concept of Newtonian gravity; and (4) PSO uses memory to store and update the velocity with the pbest and gbest, while GSA is memoryless and is concerned exclusively with the current status. Therefore, PSO and GSA have their respective specialties and potentialities to find optimum solutions. It encourages us to further design a hybridization of these two techniques to obtain better optimization performance.

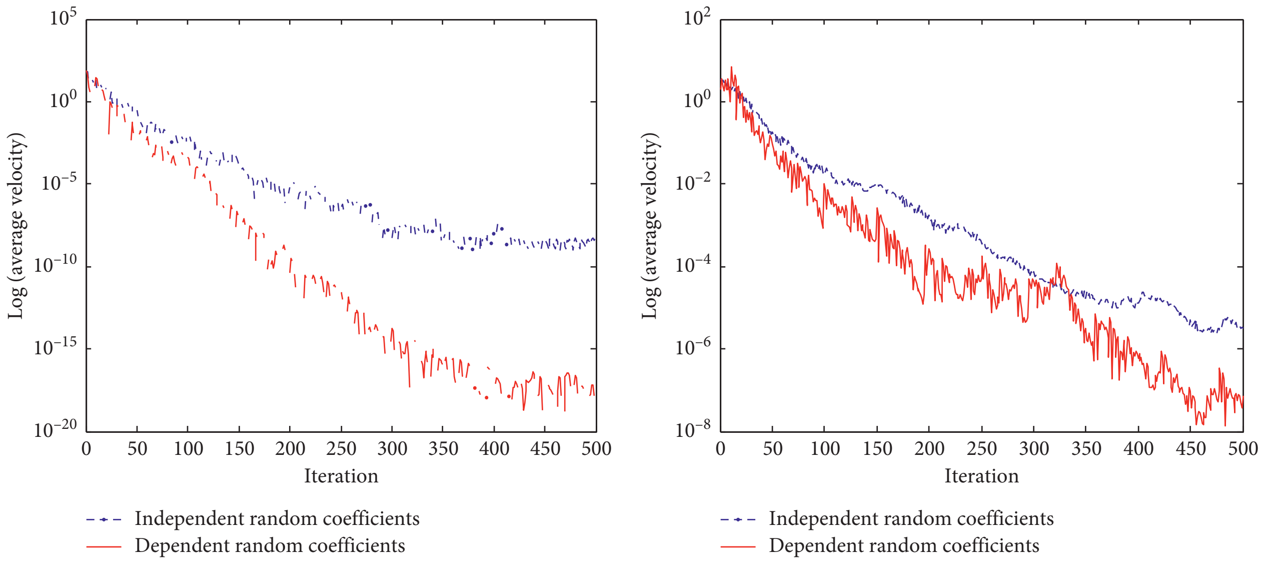
3. Sequential Hybrid Particle Swarm Optimization and Gravitational Search Algorithm

As is well known, PSO ensures that the optimization process converges faster, whereas GSA assures that the search can jump out of local optima by maintaining the diversity in the swarm [40]. Moreover, they have different search properties and movement mechanisms. Hence, we propose the new sequential hybrid version HPSO-GSA, where PSO is integrated with GSA to combine the merits of both algorithms. To be specific, first, to further balance between global and local search of the PSO, dependent random coefficients (DRCs) strategy (Section 3.1) is introduced into the HPSO-GSA. Second, to decrease the computational cost due to GSAs integration in the hybridization, two GSA-embedded strategies, namely, fixed iteration interval cycle (FIIC) (Section 3.2) and adaptive evolution stagnation cycle (AESC) (Section 3.3), are introduced into the algorithm. Finally, computational complexity of the algorithm in Section 3.4 is theoretically analyzed based on main operators.

3.1. Dependent Random Coefficients (DRCs). As shown in equation (1), the two random coefficients $r_1(t)$ and $r_2(t)$ are generated independently, so in some cases, the values of the $r_1(t)$ and $r_2(t)$ are too large or too small. For the former case, both the personal and social influences are excessively evaluated and the particles are driven too far away from the suboptimum solution. In the latter case, both the personal and social influences are negligible and the convergence speed of the algorithm is sharply reduced. To alleviate these phenomena, the dependent random coefficients strategies based on these random variables are introduced as follows:

$$\begin{aligned} v_i^d(t+1) = {} & w(t)v_i^d(t) + c_1(t)r_1(t)\left(1-r_2(t)\right)\left(p_i^d(t)-x_i^d(t)\right) \\ & + c_2(t)r_2(t)\left(1-r_1(t)\right)\left(p_g^d(t)-x_i^d(t)\right). \end{aligned} \tag{13}$$

To demonstrate the impact of the DRC strategy on the evolution of the population, an experiment was performed on the Rosenbrock and Ackley test functions. The average velocity of particles varying throughout iterations is shown in Figure 1. On the one hand, it is desirable that the particles with high velocities can explore large areas in the decision space to find new regions. From Figure 1, we can see that the DRC strategy relatively increases the average velocity of the population in the early iterations, thereby improving the diversity of the swarm that provides the particles with the ability to jump out of premature convergence. On the other hand, in the later stage, the particles in the population need to exploit local regions more precisely to improve their performances. It is obvious from the results that the average velocity of the algorithm with the DRC strategy has a lower value than that of the algorithm with independent random coefficients. In this case, the particles can find a better solution with a faster convergence speed in the last iterations.



(a)

(b)

Figure 1: Comparison results of average velocity of particles for (a) Rosenbrock and (b) Ackley function.

Instead, the velocity of the algorithm with independent random variables decreases slowly. Then, the particles need more iterations or runtimes to seek an optimum solution.

3.2. Fixed Iteration Interval Cycle (FIIC). A FIIC strategy is that the GSA is implanted into the PSO evolution framework according to a fixed iteration interval frequency T_f. For instance, T_f is set to 10; that is, the GSA can be introduced into the PSO every ten iterations. However, the optimal setting of the parameter T_f is relatively challenging and dependent on the function of termination condition such as maximum iterations. In most cases, the parameter value T_f is usually determined through the experience or the parameter sensitivity analysis. Hence, in this paper, the parameter sensitivity analysis by means of the experiments was carried out and the proper parameter value is determined. Too large or too small values of T_f are not desirable, as the former may be difficult to utilize the contribution of the GSA, whereas the latter leads to waste computation resources, thereby degrading the algorithm convergence speed. According to the results of the multigroup experiments, the T_f can be set to a large value in the range [30, 50] to save the convergence time on condition that the test function is unimodal or multimodal with relatively less local optima. Similarly, the T_f can be set to a small value within the range [1, 32] to enhance the global convergence ability on condition that the optimization function is multimodal with many local optima. In this work, the T_f is fixed at 20 according to the parameter sensitivity analysis shown in Section 4.3.

3.3. Adaptive Evolution Stagnation Cycle (AESC). To adaptively implant the GSA into the PSO framework to guide the particles access to potential regions during the entire evolution process, we employ an AESC strategy in the HPSO-GSA to effectively judge whether the algorithm reaches the premature stagnation stage. That is, the particles are found to be trapped into local optima. The AESC strategy is defined as follows:

Definition 1 (evolution stagnation). Suppose the fitness evaluation function for a given optimization problem is defined as $f(\cdot)$, and the global best position of the population found so far at iteration t is described as $p_g(t)$. For a small positive constant $\delta \geq 0$, if and only if $|f(p_g(t)) - f(p_g(t+1))| \leq \delta$ is satisfied, in such a way, the population presents evolution stagnation situation at iteration t+1 in the evolving process, where δ is the evolution stagnation radius.

Definition 2 (evolution stagnation cycle). For a small position constant $\delta \geq 0$, if the population is unvaryingly kept in evolution stagnation at minimum successive iterations $S_{\min}$ for radius δ, then the value $S_{\min}$ is defined as evolution stagnation cycle for radius δ.

The evolution stagnation cycle is calculated by (14) at the successive evolving process:

$$S(t+1) = \begin{cases} S(t)+1, & \left|f\left(p_g(t)\right) - f\left(p_g(t+1)\right)\right| \leq \delta, \\ 0, & \left|f\left(p_g(t)\right) - f\left(p_g(t+1)\right)\right| > \delta. \end{cases} \tag{14}$$

In the HPSO-GSA, the AESC strategy is applied when the condition $S(t+1) \geq S_{\min}$ is satisfied; that is, evolution stagnation cycle $S(t+1)$ reaches or exceeds its threshold value $S_{\min}$. In this case, the GSA operator is added to the PSO to increase its flexibility for solving more complicated problems. Hence, the parameter $S_{\min}$ has a direct impact on the performance of the HPSO-GSA. Too small or too large value of $S_{\min}$ is undesirable for the HPSO-GSA. In our work, we set $S_{\min} = 4$ in terms of the result of the parameter sensitivity analysis shown in Section 4.3.

3.4. Computational Complexity Analysis. Computational complexity is usually regarding the analysis of storage space requirements and computational time costs. In most cases, the time complexity analysis is the main issue for population-based metaheuristic algorithms [15, 23]. Generally, the time complexity of an algorithm is proportional to the number of dimensions D and the number of particles or agents N in the swarm when only considering a main loop and therefore can be calculated according to their main operations and worst case in a full iteration cycle. A greater number of dimensions D or a larger population size N will directly result in higher running time. As a result, the computational step analysis for PSO, GSA, and HPSO-GSA in an iterative loop is given in Table 1. From Table 1, the time consumed considering their main operations and worst case in an iterative loop for the PSO and the GSA is $O(2N + 2ND)$ and $O(2N + 3ND)$, respectively. Therefore, the worst-case consuming time for the HPSO-GSA in a single iterative loop is $O(3N + 3ND)$. Apparently, the consuming time of the algorithm is proportional to the number of particle N and the dimension of decision space D in the population. A larger population size N and/or a greater dimension D will directly result in more running time. Then, the time complexity for the HPSO-GSA is greater than that of the PSO or the GSA when the GSA is entered into the PSO. However, the hybrid approach can avoid premature convergence and thus is capable of escaping from local optima with the help of an increase in diversity. Therefore, the HPSO-GSA is still a very competitive optimization approach at the expense of a little higher time resource.

In order to describe clearly the steps of the proposed algorithm, the detailed pseudocode of the HPSO-GSA algorithm is summarized in Figure 2. It is obvious that the HPSO-GSA procedure is mainly dependent on the PSO algorithm with DRCS, and the GSA can be allowed to perform when the AESC and/or the FIIC strategies are satisfied.

4. Experimental Setup, Results, and Discussion

In this section, the experimental studies that have been performed to investigate the performance of the proposed HPSO-GSA method for classical benchmark test functions are presented. We first describe the benchmark test functions in Section 4.1. Second, the experimental design including parameter settings of the involved algorithms for comparison is described in Section 4.2. The parameter sensitivity analysis of T_f and $S_{\min}$ and the effect of different strategies in the HPSO-GSA are discussed in Sections 4.3 and 4.4, respectively. Finally, the performance investigation and comparison of the proposed algorithm is evaluated between the PSO, the GSA, and other variants of hybrid algorithm.

4.1. Benchmark Test Functions. The classical benchmark test functions with different complexities of the fitness landscape are shown in Table 2. These functions were considered in the study [15, 41] as well. This table consists of the brief descriptions of function expressions, their feasible domains, their optimal positions, and global minimum. All the functions are minimization problems. The variable D denotes the dimensions of the test functions. These functions, grouped into three sets, were designed to evaluate various aspects of algorithms. The first set of $f_1(x)$ to $f_3(x)$ consists of unimodal test functions. The second set of $f_4(x)$ to $f_9(x)$ is multimodal high-dimensional test functions with many local optima. Moreover, $f_8(x)$ and $f_9(x)$ are hybrid composition test functions. The third set, given by $f_{10}(x)$, consists of multimodal test functions with fixed dimensions.

4.2. Parameter Settings of the Involved Algorithms. In this section, we employ PSO [26], GSA [8], and five hybrid variants such as DEPSO [31], GAPSO [28], DE-GSA [37], GA-GSA [18], and PSOGSA [22] for comparison with the HPSO-GSA. These algorithms have a few parameters, some of which are common and others are specific to the algorithms.

Common parameters are the number of dimensions for the search space, the maximum number of iterations, the population size, and the total number of trials. For all test functions, except the 2D function Schaffer, we test the experiments with 30 dimensions, that is, $D = 30$. To assure the fair assessment between HPSO-GSA and its peers, all PSO variants are run independently 100 times on the test functions employed. The population size is also set to 60. The maximum number of function evaluations $\text{FE}_{\max}$ is 5000 for the Schaffer function and 30,000 for the remainders, respectively. The evolution stagnation radius δ is chosen as $1.0e-2$. The convergence criterion ε for test functions in 30 dimensions is fixed at $1.0e-3$. Each run stopped when the maximum number of iterations or the convergence criterion is reached. Similarly, the population is initialized with its position and velocity, both of which are randomly selected from the range $[x_{\max}]$. We set $v_{\max} = 0.5x_{\max}$. The upper and lower bounds of the particle's position are limited to the interval $[x_{\min}, x_{\max}]$, and the maximum velocity is restricted to $v_{\max}$. Note that all the experiments are conducted in a Windows XP Professional OS environment using Intel Core i5, 2.67 GHz, 2G RAM, and the codes are performed in Matlab 7.0.1.

Besides the common parameters, the parameter configurations for all variants employed are extracted from their optimized suggestions in the corresponding publications and are described in Table 3. For our HPSO-GSA, it is important to note that the evolution stagnation cycle $S_{\min}$ is selected as 4 and the fixed iteration interval cycle T_f is set to 20 according to the analysis of parameter sensitivity observation.

4.3. Parameter Sensitivity Analysis. The key parameters T_f and $S_{\min}$ have a direct effect on the performance of the HPSO-GSA. Hence, the experiments are performed to

Table 1: Computational complexity analysis for PSO, GSA, and HPSO-GSA.

Main step for PSO	Complexity	Main step for GSA	Complexity	Main step for HPSO-GSA	Complexity
Evaluate objectives	N	Evaluate objectives	N	Evaluate objectives	N
Update best positions	N	Compute masses	N	Update best positions	N
Update velocities	$N \times D$	Compute accelerations	$N \times D$	Compute masses	N
Compute new positions	$N \times D$	Update velocities	$N \times D$	Compute accelerations	$N \times D$
		Compute new positions	$N \times D$	Update velocities	$N \times D$
				Compute new positions	$N \times D$

```
Pseudo-code of the HPSO-GSA algorithm:
Initialization
  1: Initialize the parameters: $x_{\max}$, $x_{\min}$,$v_{\max}$, $N$, $D$, $w_{\min}$, $w_{\max}$, $c_{1\min}$, $c_{1\max}$, $c_{2\min}$, $c_{2\max}$, $\alpha$ ,$G_0$, $T_f$, $S_{\min}$, $iter_{\max}$ , $\varepsilon$, and $\delta$
  2: for each particle do
  3:     Initialize randomly the position $X_i^0 \in \Omega$
  4:     Initialize randomly the velocity $V_i^0 \le V_{\max}$
  5: end for
  6: Evaluate the fitness value of each agent (particle) $f_i^0$
  7: Find the personal best positions $p_i^0 \leftarrow f_i^0$ and the global best position $p_g^0 \leftarrow f_{best}^0$
  8: Set $t = 0$                // $t$ for iterations
  9: Set $S(t) = 0$             // $S(t)$ for evolution stagnation cycle
Loop
  10: while (termination condition is not met) do
  11:   $t = t + 1$
  12:   for ($i$ = 1 to $N$)   do          // $i$ for agents
  13:       for ($j$ = 1 to $D$) do       // $j$ for dimensions
  14:           Update the velocity ( $v_{id}$ ) and position ( $x_{id}$ ) of each particle using equations (15) and (2), respectively
  15:       end for
  16:           Evaluate the fitness value of each agent $f_i^t$
  17:           Update $p_i^t \leftarrow f_i^t$ and $p_g^t \leftarrow f_{best}^t$
  18:   end for
  19:   Compute the evolution stagnation cycle $S(t)$ by Eq. (16)
  20:   if (($t$ mod $T_f$) = 0) or ($S(t) \geqslant S_{\min}$)) then      // $T_f$ for fixed interval cycle and $S_{\min}$ for adaptive interval cycle
                                                        //GSA is added to PSO when the conditions are met
  21:           Perform the GSA processes using equations (8) and (9), respectively
  22:           Evaluate the fitness value of each agent $f_i^{t+1}$
  23:           Update $p_i^{t+1} \leftarrow f_i^{t+1}$ and $p_g^{t+1} \leftarrow f_{best}^{t+1}$
  24:   end if
  25: when the termination condition is met, output results. Otherwise, go to step 10
  26: end while
Termination
```

Figure 2: Pseudocode of the HPSO-GSA algorithm.

investigate the effect of different parameters on the proposed HPSO-GSA. For simplicity, the maximum number of iterations is set as 1000, and other parameters are the same as previously mentioned. Four well-known test functions, namely, the Sphere, Rosenbrock, Rastrigrin, and Griewank problems, have been employed to observe how the algorithm is affected by these parameters.

First, the parameter T_f of the proposed HPSO-GSA is tuned. Each test function has been tested on HPSO-GSA with different values of T_f; however, it is impossible to evaluate all the cases of the parameter. Hence, the eight selected values for this parameter $T_f = 2$, $T_f = 5$, $T_f = 10$, $T_f = 15$, $T_f = 20$, $T_f = 30$, $T_f = 40$, and $T_f = 50$ are considered, respectively. The results are averaged over 100 independent runs, and the normalized average best fitness values achieved for each parameter T_f are shown in Figure 3. Note that the normalized average best fitness is defined by means of the variable ($H_i(\tau) \in [0, 1]$) calculated as follows:

$$\mathrm{H}_i(\tau) = \frac{\mathrm{fit}_i(\tau) - \mathrm{fit}_{\min}(\tau)}{\mathrm{fit}_{\max}(\tau) - \mathrm{fit}_{\min}(\tau)}, \tag{15}$$

where i represents the group index of different parameters T_f (apparently, $i = 1, 2, \ldots, 8$) and τ denotes the Sphere, Rosenbrock, Rastrigrin, and Griewank functions. $\mathrm{fit}_i(\tau)$ is the average best fitness under the ith parameter T_f for function τ. $\mathrm{fit}_{\min}(\tau)$ and $\mathrm{fit}_{\max}(\tau)$ denote the minimum and maximum fitness under all of the cases for function τ, respectively. The aim of normalizing is to reduce the influence of different magnitudes in the same coordinate. From Figure 3, it is apparent that the searching capabilities of the HPSO-GSA are influenced by different parameter T_f. It is an interesting conclusion that too small or two large values of T_f tend to compromise the convergence accuracy of the HPSO-GSA. The best results are obtained when T_f is fixed at 20 for most of test functions. Hence, in our simulations, the HPSO-GSA uses the parameter $T_f = 20$.

Table 2: Benchmark test functions used in the experiments.

Function name	Function expression	Domain	Opt. position	Opt. value	Trait
Sphere	$f_1(x) = \sum_{i=1}^{D} x_i^2$	[−100, 100]	(0, 0, . . ., 0)	0	Unimodal
Rosenbrock	$f_2(x) = \sum_{i=1}^{D-1} 100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2$	[−30, 30]	(1, 1, . . ., 1)	0	Unimodal
Quartic noise	$f_3(x) = \sum_{i=1}^{D} i x_i^4 + \text{random}[0, 1)$	[−1.28, 1.28]	(0, 0, . . ., 0)	0	Unimodal
Schwefel	$f_4(x) = 418.98 \times D - \sum_{i=1}^{D} x_i \sin(\sqrt{\lvert x_i \rvert})$	[−500, 500]	(420.96,. . .,420.96)	0	Multimodal
Rastrigin	$f_5(x) = \sum_{i=1}^{D} (x_i^2 - 10\cos(2\pi x_i) + 10)$	[−5.12, 5.12]	(0, 0, . . ., 0)	0	Multimodal
Ackley	$f_6(x) = -20\exp(-0.2\sqrt{(1/D)\sum_{i=1}^{D} x_i^2}) - \exp((1/D)\sum_{i=1}^{D}\cos(2\pi x_i)) + 20 + e$	[−30, 30]	(0, 0, . . ., 0)	0	Multimodal
Griewank	$f_7(x) = (1/4000)\sum_{i=1}^{D}(x_i)^2 - \prod_{i=1}^{D}\cos(x_i/\sqrt{i}) + 1$	[−600, 600]	(0, 0, . . ., 0)	0	Multimodal
Penalized1	$f_8(x) = (\pi/D)\left\{10\sin^2(\pi y_1) + \sum_{i=1}^{D-1}(y_i - 1)^2 \cdot [1 + 10\sin^2(\pi y_{i+1})] + (y_D - 1)^2\right\} + \sum_{i=1}^{D} u_i(x_i, 10, 100, 4)$	[−50, 50]	(1, 1, . . ., 1)	0	Multimodal
Penalized2	$f_9(x) = 0.1\left\{\sin^2(3\pi x_1) + \sum_{i=1}^{D}(x_i - 1)^2 \cdot [1 + \sin^2(3\pi x_i + 1)] + (x_D - 1)^2 \cdot [1 + \sin^2(2\pi x_D)]\right\} + \sum_{i=1}^{D} u_i(x_i, 5, 100, 4)$	[−50, 50]	(1, 1, . . ., 1)	0	Multimodal
Schaffer	$f_{10}(x) = 0.5 + (sin\sqrt{(x_1^2 + x_2^2)})^2 - 0.5/(1 + 0.001(x_1^2 + x_2^2))^2$	[−100, 100]	(0, 0, . . ., 0)	0	Multimodal

Remark: in the Penalized1 and Penalized2, $y_i = 1 + (1/4(x_i + 1))$ and $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & \text{if} \quad x_i > a, \\ 0 & \text{if} \quad -a \le x_i \le a, \\ k(-x_i - a)^m & \text{if} \quad x_i < -a. \end{cases}$

Table 3: Parameter settings of the involved algorithms.

Algorithms	Parameter settings	Reference
PSO	$w_{\max} = 0.9$, $w_{\min} = 0.4$, $c_1 = c_2 = 2.0$	[26]
GSA	$G_0 = 100$, $\alpha = 20$	[8]
DEPSO	$c_1 = c_2 = 2.0$, $w = 0.4$, $\text{CR} = 0.9$, $F = 0.5$	[31]
GA-PSO	$c_1 = c_2 = 2.0$, $w = 0.4$, $P_c = 0.55$, $P_m = 0.01$	[28]
DE-GSA	$G_0 = 100$, $\alpha = 20$, $\text{CR} = 0.9$, $F = 0.5$	[37]
GA-GSA	$G_0 = 100$, $\alpha = 20$, $P_c = 0.65$, $P_m = 0.01$	[18]
PSOGSA	$G_0 = 1.0$, $\alpha = 20$, $c_1 = c_2 = 1.0$, $w_{\max} = 0.9$, $w_{\min} = 0.4$	[22]
HPSO-GSA	$G_0 = 100$, $\alpha = 20$, $w_{\max} = 0.9$, $w_{\min} = 0.4$, $c_{1\min} = 0.5$, $c_{1\max} = 2.5$, $c_{2\min} = 0.5$, $c_{2\max} = 2.5$, $T_f = 20$, $S_{\min} = 4$	Present work

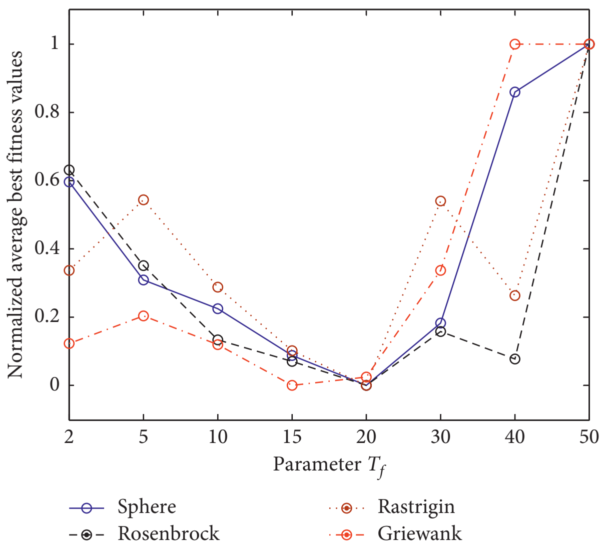


Figure 3: Results of the HPSO-GSA for different parameter T_f.

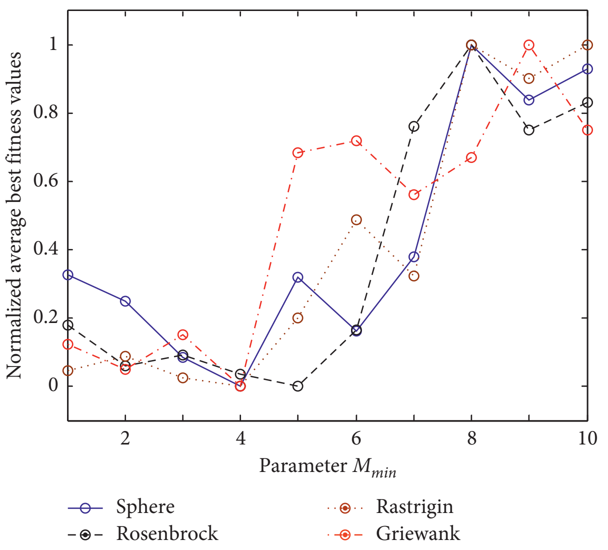


Figure 4: Results of the HPSO-GSA for different parameter $M_{\min}$.

Second, the parameter $S_{\min}$ is similarly considered. The possible ten scenarios are tested when the range of this parameter is selected from [1, 10] by step size 1. The results obtained by the HPSO-GSA for test functions are given in Figure 4. It is clear that, for the Rosenbrock function, $S_{\min} = 5$ produces the best average best fitness. For other functions, the best results are obtained for $S_{\min} = 4$. Hence, $S_{\min} = 4$ is a desirable choice for the proposed algorithm in the following experiments.

4.4. Effectiveness of Different Strategies. The proposed algorithm employs three strategies, namely, DRC, FIIC, and AESC. To evaluate the impact on the performance improvement incurred by each of these strategies, we investigate the performance of (1) HPSO-GSA without the DRC strategy and with the FIIC strategy (HPSO-GSA1), (2) HPSO-GSA without the DRC strategy and with the FIIC strategy (HPSO-GSA2), (3) HPSO-GSA with the DRC strategy and the FIIC strategy (HPSO-GSA3), (4) HPSO-GSA with the DRC strategy and the AESC strategy (HPSO-GSA4), and (5) the complete HPSO-GSA. For HPSO-GSA, the three strategies are integrated with the algorithm. In this experiment, seven test functions, except for function Schaffer, in 10 dimensions are considered. The maximum number of iterations is set to 500, and a total of 30 runs for each algorithm are conducted.

To assure a fair comparison, we compare the average best fitness $\text{fit}_{\text{average}}$ values obtained by HPSO-GSA1, HPSO-GSA2, HPSO-GSA3, HPSO-GSA4, and HPSO-GSA with those obtained by the PSO. The comparison results are expressed according to the percentage improvement (%improve) computed as follows [36]:

$$\%\text{improve} = \frac{\text{fit}_{\text{average}}(\text{PSO}) - \text{fit}_{\text{average}}(\eta)}{\left|\text{fit}_{\text{average}}(\text{PSO})\right|} \times 100\%, \quad (16)$$

where η represents HPSO-GSA variants. If η has better performance (that is smaller average fitness) than PSO, %improve is positive. Otherwise, it is negative. The results of $\text{fit}_{\text{average}}$ and %improve for all involved algorithms are given in Table 4.

From Table 4, all of HPSO-GSA variants have obtained performance improvement in comparison with the PSO, implying that the utilization of any strategies, namely, DRC, FIIC, and AESC, can contribute to improving the PSOs searching accuracy. Among all the HPSO-GSA variants, the HPSO-GSA shows best performance according to the largest average %improve, followed by the HPSO-GSA4, HPSO-GSA3, HPSO-GSA2, and HPSO-GSA1. This conclusion suggests that the integration of three strategies into the

Table 4: F_{average} and %improve results obtained by PSO and HPSO-GSA variants for 10 dimensions in test functions.

Function	F_{average} (%improve)					
	PSO	HPSO-GSA1	HPSO-GSA2	HPSO-GSA3	HPSO-GSA4	HPSO-GSA
f_{Sph}	$1.68e-13$	$1.62e-13$ (3.02)	$1.56e-13$ (7.16)	$1.36e-13$ (20.04)	$1.02e-13$ (40.28)	**2.61e − 14 (84.52)**
f_{Ros}	$2.92e+01$	$6.87e-01$ (97.26)	$4.63e-01$ (98.40)	$3.42e-03$ (100.00)	$3.56e-03$ (100.00)	**2.18e − 03 (100.00)**
f_{Qua}	$8.63e+00$	$8.21e+00$ (4.86)	$6.52e+00$ (24.45)	$4.06e+00$ (52.95)	$3.62e+00$ (58.05)	**1.06e + 00 (87.72)**
f_{Schw}	−3521.6	−3862.6 (9.68)	−4104.3 (16.55)	−4043.1 (14.81)	−4176.4 (18.59)	**−4189.8 (18.97)**
f_{Ras}	$6.04e-02$	$2.63e-02$ (56.46)	$3.62e-04$ (99.42)	$5.26e-04$ (99.15)	$6.02e-06$ (100.00)	**5.08e − 08 (100.00)**
f_{Ack}	$2.68e-03$	$1.93e-03$ (27.99)	$3.72e-04$ (86.12)	$3.11e-06$ (99.88)	$8.63E-08$ (100.00)	**7.82e − 10 (100.00)**
f_{Gri}	$4.28e-03$	$8.32e-04$ (80.56)	$2.36e-05$ (99.45)	$5.87e-06$ (99.86)	$2.31E-06$ (99.95)	**2.08e − 10 (100.00)**
Average %improve		39.97	61.65	69.52	73.84	**84.45**

HPSO-GSA has significantly improved its optimizing performance. Hence, we use the HPSO-GSA for comparative study in the following section.

As shown in Table 4, HPSO-GSA3 and HPSO-GSA4 obtain higher average %improve than HPSO-GSA1 and HPSO-GSA2, especially in multimodal functions. This implies that the DRC strategy helps to escape from the local optima. On the other hand, HPSO-GSA2 and HPSO-GSA4 show better results against HPSO-GSA1 and HPSO-GSA3, respectively. This observation suggests that our AESC strategy performs better than the FIIC strategy. Finally, we observe some performance deteriorations in the functions f_{Schw} and f_{Ras} for HPSO-GSA2 and HPSO-GSA3, as the %improve value of HPSO-GSA3 is lower than that of HPSO-GSA2. It indicates that the AESC strategy can effectively improve the searching accuracy in certain functions in the absence of having the DRC strategy. The AESC strategy is more likely to help the HPSO-GSA to enhance the performance in some cases.

4.5. Comparative Study. To assess the performance of the HPSO-GSA compared to seven other optimization algorithms, the simulation experiments based on different measures are presented. These measures provide the ability to evaluate algorithms from different points. First, the performance results of the HPSO-GSA compared to PSO, GSA, and other state-of-the-art hybrid variants like DEPSO, GAPSO, DE-GSA, GA-GSA, and PSOGSA are presented in Section 4.5.1. Following the experiments, the statistical comparisons among the involved algorithms are also given to determine whether improvements of the proposed method are significant, as shown in Section 4.5.2.

4.5.1. Performance Results. The simulation results for each test function are recorded in Tables 5 and 6 based on three evaluation criteria such as accuracy, reliability, and efficiency by means of the mean best fitness (mean), standard deviation (std. dev.), success rate (SR), and searching time (ST). In these tables, mean is defined as the average result of best fitness generated by each algorithm for each function in 100 independent trials. A low mean is desirable as it indicates the algorithm has better optimizing accuracy. Std. dev. measures the amount of variation from the average. A small std. dev. indicates an algorithm has good stability. SR represents the consistency of an algorithm to achieve a predefined convergence level ε among the maximum iterations. A larger SR implies that an algorithm is more reliable as it can consistently solve a problem with the accuracy level ε. Finally, the computational cost can be evaluated by the mean ST which represents the algorithm's convergence speed with the predefined solution accuracy. The best results obtained by the algorithms are shown in bold for each metric.

From Table 5, we observe that HPSO-GSA has the lowest optimizing accuracy for solving all test functions except for functions quartic noise and Schwefel. Specifically, for the previous three unimodal functions, HPSO-GSA shows no much superior searching accuracy for function quartic noise, as the smallest mean values have been obtained by GA-GSA. The HPSO-GSA significantly outperforms other algorithms for function Rosenbrock. It implies that the proposed algorithm provides better exploration ability for avoiding the premature convergence, as the global optimum of this function is located in a narrow, long, and parabolic shaped flat valley. So it is often used to evaluate the ability of an algorithm in mitigating the stagnation problem. For the great majority of multimodal functions including complex hybrid composition, the HPSO-GSA surpasses all the other contenders as it has the smallest convergence accuracy. Hence, the HPSO-GSA can provide an appropriate level of global search escaping from many local optima.

Meanwhile, we present the SR and ST results produced by all the involved algorithms, as shown in Table 6, in order to compare the algorithm's reliability and computational cost, respectively. First, we observe that the HPSO-GSA and DE-GSA have more superior searching reliability than their peers for all the problems, as it converges successfully to the acceptable accuracy level with success rate 1. It is mentioned that the PSO never converges to the criteria for test functions at the predefined level ε. The remaining algorithms, especially the DEPSO, GAPSO, GA-GSA, and PSOGSA, are able to partially solve all the test functions. As a consequence, the proposed algorithm provides appropriate balance between exploration and exploitation abilities, guaranteeing to converge towards the predefined criteria. Second, as for the searching times, it is clear from Table 6 that the involved algorithms show various ST values for each test function. For instance, the computational overload of the PSO is the lowest, as it has advantage of fast convergence with simple implementation and global searching guide, whereas the second lowest ST values are achieved by the HPSO-GSA for all the employed functions. This is because GSA operator is

Table 5: Mean best fitness (mean) and standard deviation (std. dev.) results for benchmark test functions.

Function		PSO	GSA	DEPSO	GAPSO	DE-GSA	GA-GSA	PSOGSA	HPSO-GSA
f_{Sph}	Mean	1.47e + 00	5.72e − 07	1.97e − 02	2.06e − 17	2.18e − 23	5.18e − 21	2.52e − 23	**1.83e − 24**
	Std. dev.	3.36e − 01	1.46e − 07	6.26e − 03	2.82e − 18	4.52e − 23	6.22e − 21	1.35e − 23	**2.42e − 24**
f_{Ros}	Mean	8.92e − 01	2.62e − 08	3.06e − 10	3.69e − 09	1.36e − 13	1.11e − 14	1.84e − 16	**5.19e − 19**
	Std. dev.	6.83e − 01	3.26e − 08	2.02e − 10	6.26e − 09	2.74e − 13	2.41e − 14	2.76e − 17	**9.25e − 20**
f_{Qua}	Mean	3.38e + 00	1.36e − 04	5.42e − 02	4.05e − 04	9.72e − 04	**3.12e − 16**	2.47e − 06	2.12e − 06
	Std. dev.	1.56e + 00	8.54e − 05	1.63e − 02	4.61e − 04	6.82e − 04	**2.41e − 16**	6.25e − 06	8.35e − 06
f_{Schw}	Mean	2.88e − 02	3.81e + 03	1.18e + 02	8.72e + 00	**2.61e − 08**	2.88e − 05	1.11e − 02	2.55e − 05
	Std. dev.	5.42e − 01	5.52e + 02	4.54e + 02	3.65e + 00	**9.52e − 08**	3.51e − 06	7.26e − 02	1.37e − 06
f_{Ras}	Mean	3.12e + 02	5.87e + 01	1.09e + 01	1.76e − 15	6.91e − 12	1.98e + 00	8.95e + 00	**4.02e − 22**
	Std. dev.	3.57e + 01	5.52e + 00	7.26e + 00	3.25e − 15	4.21e − 12	4.56e − 01	2.73e + 00	**2.84e − 22**
f_{Ack}	Mean	2.01e + 01	3.56e − 10	2.59e − 09	2.66e − 15	1.92e − 10	8.77e − 10	8.88e − 16	**1.86e − 19**
	Std. dev.	5.42e − 00	4.65e − 10	3.81e − 09	6.26e − 15	4.26e − 10	2.15e − 10	3.55e − 16	**2.37e − 19**
f_{Gri}	Mean	1.67e − 01	1.97e − 02	5.91e − 02	1.21e − 01	3.06e − 07	7.39e − 03	9.85e − 03	**1.21e − 09**
	Std. dev.	6.83e − 02	8.21e − 02	3.31e − 02	6.24e − 01	4.65e − 07	1.26e − 04	5.73e − 03	**6.52e − 09**
f_{Pen1}	Mean	2.68e + 00	6.22e − 01	6.39e − 09	9.37e − 23	3.26e − 21	1.52e − **19**	5.21e − 23	**2.54e − 32**
	Std. dev.	1.34e + 00	6.37e − 01	3.62e − 09	5.36e − 23	4.25e − 21	5.62e − 19	4.26e − 23	**1.53e − 32**
f_{Pen2}	Mean	3.11e − 01	1.09e − 02	1.29e − 18	4.08e − 18	2.18e − 22	1.52e − 08	2.86e − 20	**1.25e − 22**
	Std. dev.	9.64e − 02	4.22e − 02	4.62e − 18	6.85e − 18	3.25e − 22	5.23e − 08	3.65e − 20	**9.64e − 22**
f_{Sch}	Mean	9.72e − 02	5.77e − 04	1.57e − 04	3.94e − 09	1.54e − 09	2.24e − 05	1.99e − 17	**9.13e − 19**
	Std. dev.	2.23e − 03	4.62e − 04	3.64e − 04	2.12e − 09	6.73e − 09	6.42e − 05	2.12e − 17	**8.35e − 19**

Best results are provided in bold.

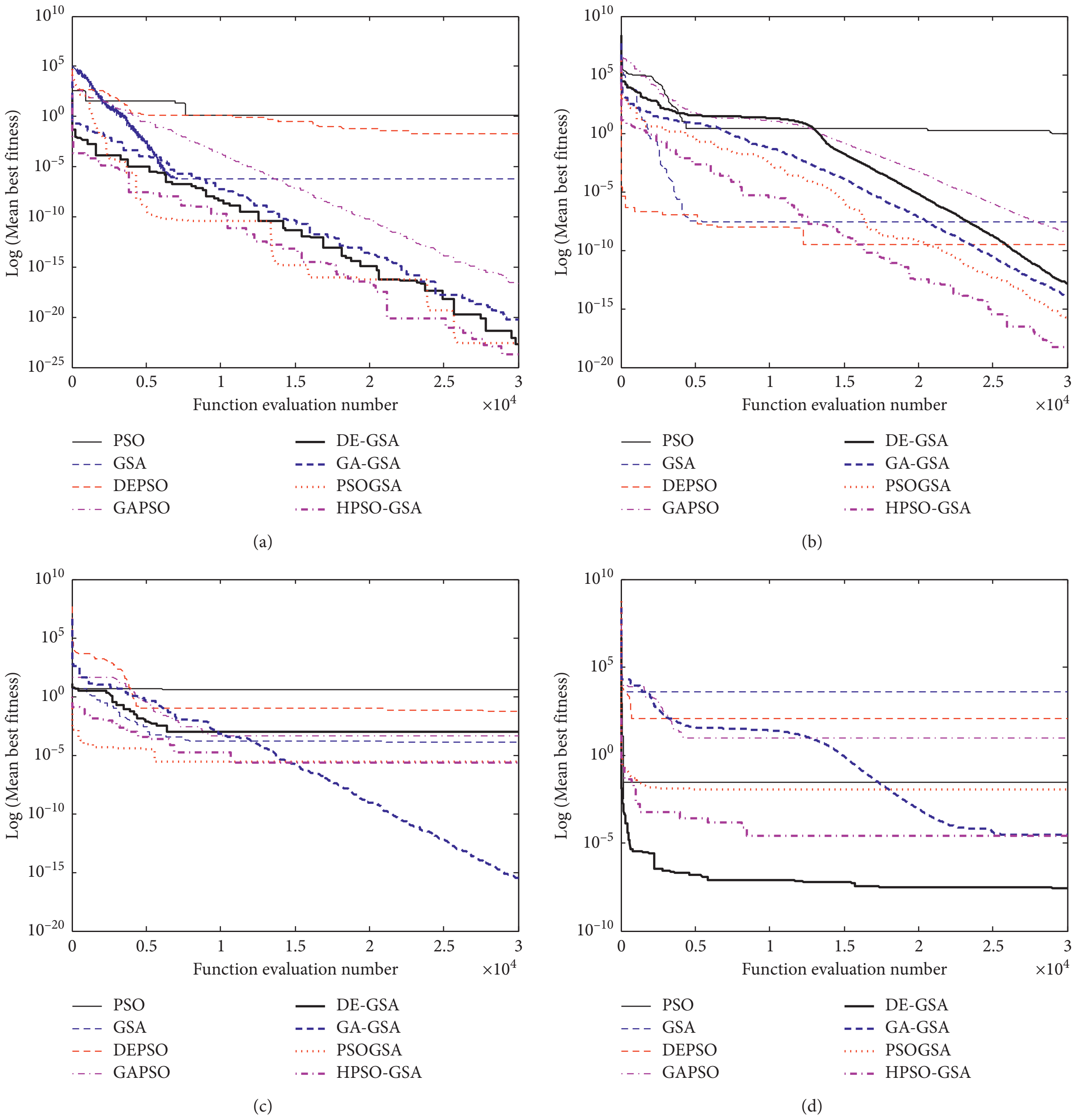
Table 6: Success rate (SR) and searching time (ST, in seconds) results for benchmark test functions.

Function		PSO	GSA	DEPSO	GAPSO	DE-GSA	GA-GSA	PSOGSA	HPSO-GSA
f_{Sph}	SR	0.58	**1.00**	0.86	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	ST	**2.54**	15.62	18.46	12.62	22.85	20.26	20.24	4.83
f_{Ros}	SR	0.12	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	ST	**2.66**	16.06	26.21	16.52	25.85	21.34	19.42	6.33
f_{Qua}	SR	0.00	**1.00**	0.52	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	ST	**4.23**	23.74	22.46	16.88	30.22	24.62	25.42	8.32
f_{Schw}	SR	0.34	0.00	0.00	0.00	**1.00**	**1.00**	0.75	**1.00**
	ST	4.63	18.22	24.86	15.32	24.52	20.68	20.66	4.68
f_{Ras}	SR	0.00	0.00	0.00	**1.00**	**1.00**	0.00	0.00	**1.00**
	ST	**3.22**	19.68	24.63	16.22	26.53	23.52	20.08	5.26
f_{Ack}	SR	0.00	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	ST	**3.56**	18.86	20.31	13.94	25.44	20.28	19.51	4.36
f_{Gri}	SR	0.00	0.85	0.32	0.00	**1.00**	0.90	0.92	**1.00**
	ST	**3.34**	19.08	22.35	14.82	26.73	23.90	19.06	5.43
f_{Pen1}	SR	0.00	0.00	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	ST	**5.83**	20.52	16.85	19.73	25.06	22.52	22.05	9.17
f_{Pen2}	SR	0.00	0.26	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	ST	**4.32**	16.53	15.66	16.08	25.23	23.55	20.04	8.26
f_{Sch}	SR	0.72	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	ST	**0.92**	3.06	4.52	3.44	5.21	4.81	4.52	1.08

Best results are provided in bold.

added to our algorithm, resulting in the smaller computational overload. Other algorithms such as GSA, DEPSO, GAPSO, DE-GSA, GA-GSA, and PSOGSA require higher computational times compared to PSO and HPSO-GSA. The excellent performance of the HPSO-GSA in terms of accuracy and success rate also confirms that the HPSO-GSA is more computationally efficient than other hybrid variants.

To further evaluate the algorithm's convergence speed qualitatively, the convergence curves of all algorithms for all test functions are presented in Figure 5. The evolution tendency of an algorithm represents its convergence behavior and its convergence speed throughout the iterations. From Figure 5, the rapid convergence properties of HPSO-GSA are reflected by the convergence curves except for quartic noise and Schwefel functions; that is, HPSO-GSA



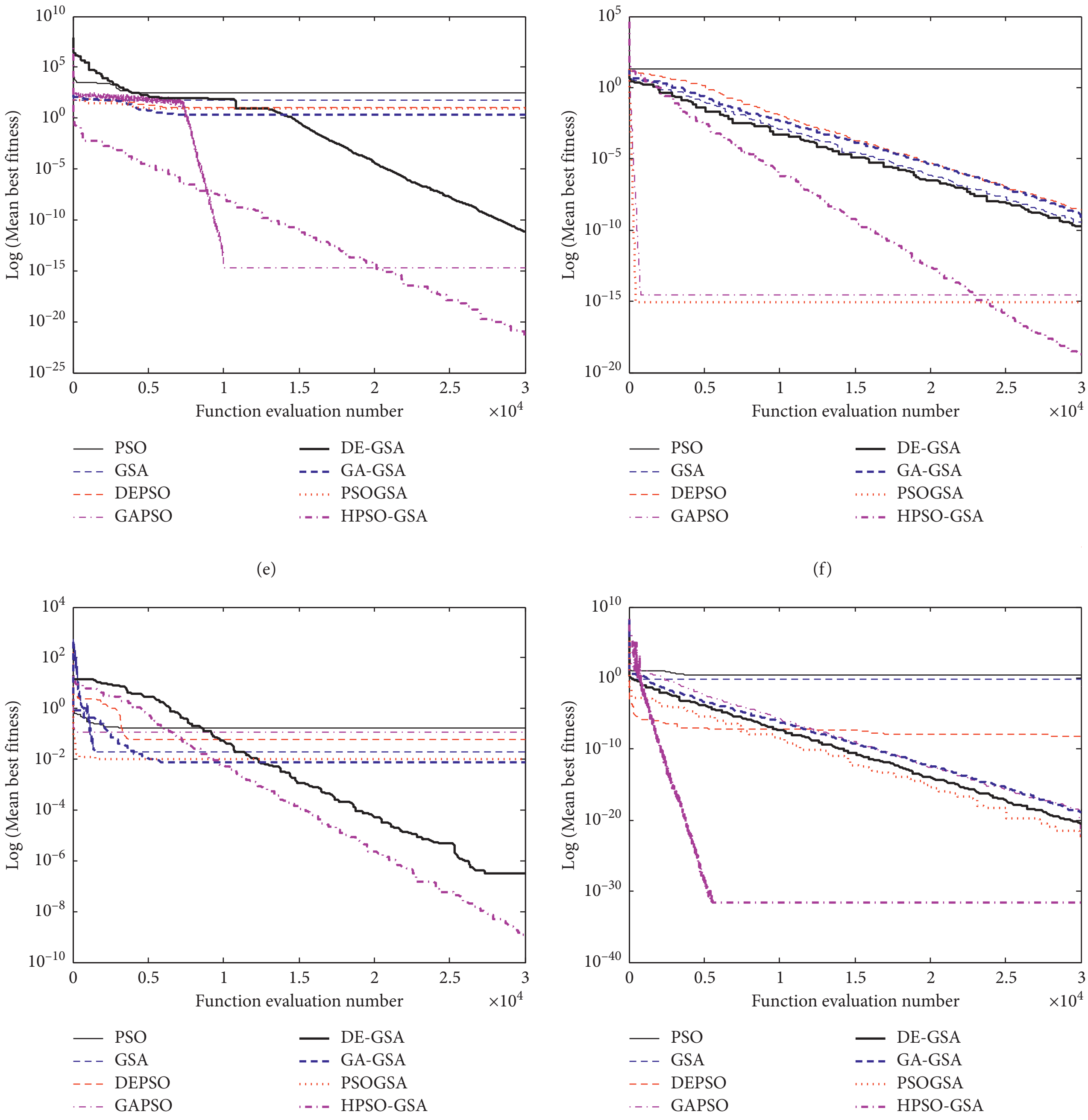
(a)

(b)

(c)

(d)

Figure 5: Continued.



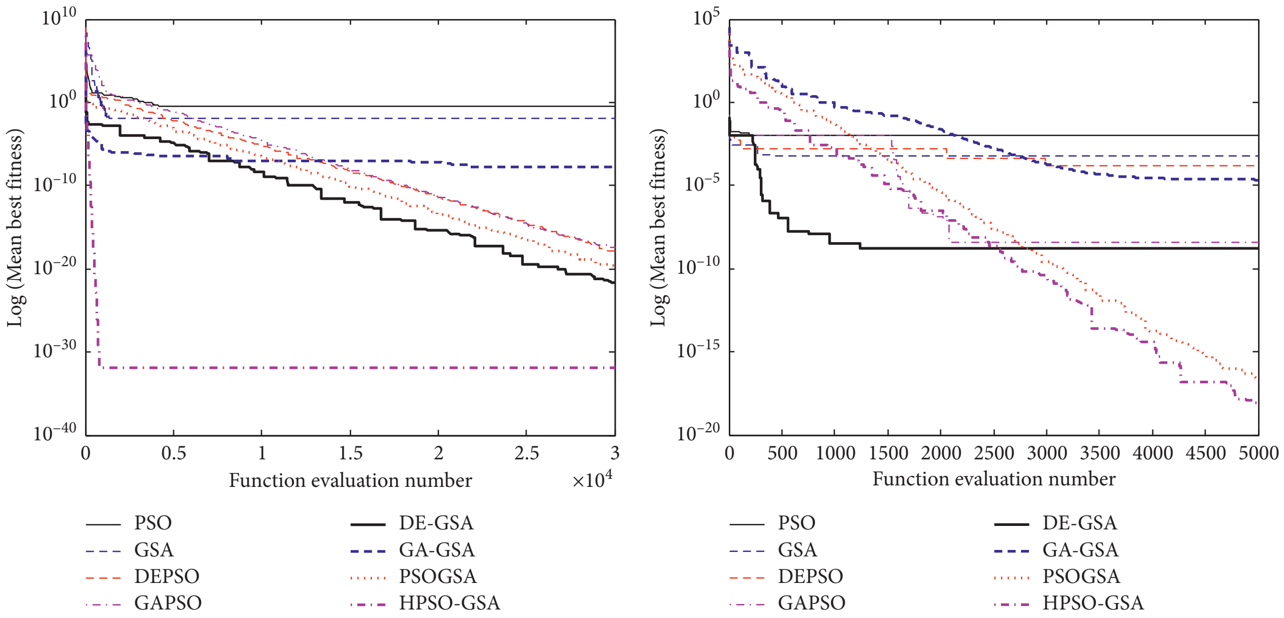
(e)

(f)

(g)

(h)

Figure 5: Continued.



(i)

(j)

Figure 5: Convergence curves of mean best fitness obtained by all algorithms for benchmark test functions: (a) Sphere; (b) Rosenbrock; (c) quartic noise; (d) Schwefel; (e) Rastrigin; (f) Ackley; (g) Griewank; (h) Penalized1; (i) Penalized2; (j) Schaffer.

requires the less computational time to converge to acceptable level ε with less iteration. To be specific, PSO and DEPSO do not reach the specified convergence level if the function evaluation numbers are extended for function sphere. For Rosenbrock function, all the algorithms except for PSO converge to reach the criterion. Moreover, the best results are obtained by HPSO-GSA. As for multimodal functions, DE-GSA and HPSO-GSA show similar convergence curves for function Rastrigin; however, HPSO-GSA achieves better optimization accuracy. Meanwhile, GSA, DEPSO, DE-GSA, GA-GSA, and HPSO-GSA can continuously optimize the function Ackley throughout the function evaluation numbers, though they only provide a slower convergence speed. It reveals that the convergence accuracy may improve if the evolutionary process is continued. It is worth mentioning that more than half of hybrid variants including GAPSO, DE-GSA, GA-GSA, PSOGSA, and HPSO-GSA exhibit better convergence characteristics by extending the evolutionary process. They can converge to the predefined level ε in the early or middle stages of optimization. This implies that the incorporation of different algorithms can enhance optimization ability of the original algorithm, as it provides more mechanisms to mitigate the stagnation problem. Generally, the HPSO-GSA provides more competitive performance in terms of global accuracy and success rate, compared to other hybrid variants.

4.5.2. Statistical Comparisons of Different Algorithms. To thoroughly compare the HPSO-GSA with its competitors, we perform a two-tailed Taillard test (t-test) [42] with 58 degrees of freedom at a 0.05 level of significance and the Wilcoxon test [43]. Results of t-test (T) achieved by all the involved algorithms based on mean best fitness and success rate are reported in Tables 7 and 8, respectively. Additionally, we rank (R) the algorithms from the smallest mean value to the largest one for each function. The algorithms that are not statistically different from each other are given the same rank. The corresponding ranking R values are also listed in Table 7. To obtain the overall performance, we summarize the T values among HPSO-GSA and other peers as "+/ = /−" in the last row of the table. The "+/ = /−" denotes the number of test functions that HPSO-GSA performs significantly better, almost the same as and considerably worse than its competitors, respectively. Meanwhile, the overall average ranks over the number of test functions and the order of the average ranks are listed in Table 7.

From Table 7, we observe that the number of test functions where HPSO-GSA performs considerably better than its contenders (T = "+") is much larger than the number of test functions where HPSO-GSA obtains significantly worse results than its peers (T = "−"). Then, the best searching accuracy of HPSO-GSA among 8 algorithms is further validated by the t-test results. Particularly, the HPSO-GSA significantly surpasses all of its peers for functions Sphere, Rosenbrock, Rastrigin, Ackley, Griewank, Penalized1, Penalized2, and Schaffer. Moreover, the t-test results indicate that HPSO-GSA is statistically different in all the compared algorithms including PSO, GSA, DEPSO, and GAPSO. Also, based on the average ranks, the algorithms are sorted into the following order: HPSO-GSA, PSOGSA, DE-GSA, GA-GSA, GAPSO, DEPSO, GSA, and PSO. Due to the total and average ranks of HPSO-GSA are smaller than those of other algorithms, HPSO-GSA obtained a better overall performance than all other algorithms. Moreover, the

Table 7: Results of t-test (T) and ranking (R) of all the involved algorithms based on mean best fitness.

Function		PSO	GSA	DEPSO	GAPSO	DE-GSA	GA-GSA	PSOGSA	HPSO-GSA
f_{Sph}	T	+	+	+	+	+	+	+	
	R	8	6	7	5	2	4	2	1
f_{Ros}	T	+	+	+	+	+	+	+	
	R	8	7	5	6	4	3	2	1
f_{Qua}	T	+	+	+	+	+	−	=	
	R	8	4	7	4	4	1	2	2
f_{Schw}	T	+	+	+	+	−	=	+	
	R	6	8	6	5	1	2	4	2
f_{Ra}	T	+	+	+	+	+	+	+	
	R	8	6	6	2	3	4	4	1
f_{Ack}	T	+	+	+	+	+	+	+	
	R	8	4	7	3	4	4	2	1
f_{Gri}	T	+	+	+	+	+	+	+	
	R	7	5	5	7	2	3	3	1
f_{Pen1}	T	+	+	+	+	+	+	+	
	R	8	7	6	2	4	5	2	1
f_{Pen2}	T	+	+	+	+	+	+	+	
	R	8	7	4	4	1	6	3	1
f_{Sch}	T	+	+	+	+	+	+	+	
	R	8	6	6	3	3	5	2	1
Average ranks		8 (9.63)	7 (7.50)	6 (7.37)	5 (5.13)	3 (3.50)	4 (4.63)	2 (3.25)	**1 (1.50)**
+/=/−		**10/0/0**	**10/0/0**	**10/0/0**	**10/0/0**	9/0/1	8/1/1	9/1/0	

Table 8: Results of t-test (T) of all the involved algorithms based on success rate.

Function	PSO	GSA	DEPSO	GAPSO	DE-GSA	GA-GSA	PSOGSA	HPSO-GSA
f_{Sph}	+	=	+	=	=	=	=	
f_{Ros}	+	=	=	=	=	=	=	
f_{Qua}	+	=	+	=	=	=	=	
f_{Schw}	+	+	+	+	=	=	+	
f_{Ras}	+	+	+	=	=	+	+	
f_{Ack}	+	=	=	=	=	=	=	
f_{Gri}	+	+	+	+	=	+	+	
f_{Pen1}	+	+	=	=	=	=	=	
f_{Pen2}	+	+	=	=	=	=	=	
f_{Sch}	+	=	=	=	=	=	=	
+/=/−	**10/0/0**	5/5/0	5/5/0	2/8/0	0/10/0	2/8/0	3/7/0	

analysis indicates that HPSO-GSA (with both the private thinking parts of the PSO and sequential mode) performs better than PSOGSA (without the personal thinking part of the PSO).

It is apparent from Table 8 that the number of test functions where HPSO-GSA performs considerably better than and almost the same as its contenders (T = "+" and "=") is much larger than the number of test functions where HPSO-GSA obtains significantly worse results than its peers (T = "−"). It reveals that HPSO-GSA is statistically different from GAPSO, DE-GSA, GA-GSA, and PSOGSA. If we increase the convergence level in the experiments, the "+" values in this table would increase and HPSO-GSA exhibits more excellent performance compared to other algorithms.

Table 9: Results of Wilcoxon test between HPSO-GSA and other algorithms on test functions.

HPSO-GSA	p values
PSO	**0.003**
GSA	**0.016**
DEPSO	**0.012**
GAPSO	**0.036**
DE-GSA	0.263
GA-GSA	**0.042**
PSOGSA	0.292

The p values below 0.05 are shown in bold.

To compare the performance difference between HPSO-GSA and the other nine algorithms, we also conduct a Wilcoxon signed-rank test. Table 9 shows the resultant p values when comparing HPSO-GSA with other algorithms. The p values below 0.05 are shown in bold. The results show that HPSO-GSA is significantly better than other algorithms except for DE-GSA and PSOGSA. However, HPSO-GSA significantly outperforms DE-GSA and PSOGSA according to the searching accuracy in Table 5 and the average ranks in Table 7.

Based on the aforementioned performance evaluation and statistical results, we conclude that the proposed hybrid algorithm performs better overall in the involved test functions compared to PSO, GSA, DEPSO, GAPSO, DE-GSA, GA-GSA, and PSOGSA.

5. Conclusion

In this paper, a novel hybridization approach of PSO and GSA through sequential pattern, namely, HPSO-GSA, which consists of three learning strategies, dependent random coefficients, fixed iteration interval cycle, and adaptive evolution stagnation cycle, is proposed to solve the global optimization problems. The employment of the dependent random coefficients enhances the better balance between global and local searches, as it improves the diversity of the swarm. The fixed iteration interval cycle and adaptive evolution stagnation cycle is proposed for seamlessly integrating PSO and GSA in a less computational cost, thereby enhancing the algorithm's convergence speed. Meanwhile, the GSA operators encourage exploration and thus improve the premature stagnation problem. The experimental studies were performed to assess the performance of the proposed HPSO-GSA for solving benchmark test functions, as well as the impact of each employed strategy on the performance of the algorithm. The results indicate that the HPSO-GSA achieves better performance than its contenders investigated in this paper in terms of searching accuracy, algorithm reliability, and computational cost. Hence, the HPSO-GSA is a promising alternative solution to optimization problem. Possible future work includes extending the application of HPSO-GSA in the real-world optimization problem. In addition, we will investigate the suitable hybridization strategies to alleviate the stagnation tendency of HPSO-GSA.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant no. 61572238, Anhui Provincial Natural Science Foundation under Grant nos. 1608085QF157 and 1708085ME132, and Key Project of Natural Science Research of Anhui Provincial Department of Education under Grant no. KJ2016A431.

References

[1] A. Gupta, D. Singh, and M. Kaur, "An efficient image encryption using non-dominated sorting genetic algorithm-III based 4-D chaotic maps," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 3, pp. 1309–1324, 2020.

[2] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference Neural Networks*, IEEE Press, Perth, Australia, pp. 1942–1948, November 1995.

[3] M. Wang, S. Feng, C. He, Z. Li, and Y. Xue, "An artificial immune system algorithm with social learning and its application in industrial PID controller design," *Mathematical Problems in Engineering*, vol. 2017, Article ID 3959474, 13 pages, 2017.

[4] M. Kaur, H. K. Gianey, D. Singh, and M. Sabharwal, "Multi-objective differential evolution based random forest for e-health applications," *Modern Physics Letters B*, vol. 33, no. 5, Article ID 1950022, 2019.

[5] S. Tabakhi, A. Najafi, R. Ranjbar, and P. Moradi, "Gene selection for microarray data classification using a novel ant colony optimization," *Neurocomputing*, vol. 168, pp. 1024–1036, 2015.

[6] D. Nelson Jayakumar and P. Venkatesh, "Glowworm swarm optimization algorithm with topsis for solving multiple objective environmental economic dispatch problem," *Applied Soft Computing*, vol. 23, pp. 375–386, 2014.

[7] F. Wahid and D. H. Kim, "An efficient approach for energy consumption optimization and management in residential building using artificial bee colony and fuzzy logic," *Mathematical Problems in Engineering*, vol. 2016, Article ID 9104735, 13 pages, 2016.

[8] E. Rashedi, H. Nezamabadi-pour, and S. Saeid, "GSA: a gravitational search algorithm," *Information Sciences*, vol. 179, no. 1, pp. 2232–2248, 2009.

[9] T. Jiang, C. Zhang, and H. Zhu, "Energy-efficient scheduling for a job shop using grey wolf optimization algorithm with double-searching mode," *Mathematical Problems in Engineering*, vol. 2018, Article ID 8574892, 12 pages, 2018.

[10] L. Guo, Z. Meng, Y. Sun, and L. Wang, "Parameter identification and sensitivity analysis of solar cell models with cat swarm optimization algorithm," *Energy Conversion and Management*, vol. 108, no. 2, pp. 520–528, 2016.

[11] Z. Li and D. Li, "An improved global harmony search algorithm for the identification of nonlinear discrete-time systems based on volterra filter modeling," *Mathematical Problems in Engineering*, vol. 2016, Article ID 3102845, 13 pages, 2016.

[12] Y. Tian and Z. Lu, "Chaotic s-box: intertwining logistic map and bacterial foraging optimization," *Mathematical Problems in Engineering*, vol. 2017, Article ID 6969312, 11 pages, 2017.

[13] S. Mirjalili and S. Z. M. Hashim, "A new hybrid PSOGSA algorithm for function optimization,," in *Proceeding of the IEEE International Conference on Computer and Information Application*, pp. 374–377, Tianjin, China, 2010.

[14] H. Liu, Z. Cai, and Y. Wang, "Hybridizing particle swarm optimization with differential evolution for constrained numerical and engineering optimization," *Applied Soft Computing*, vol. 10, no. 2, pp. 629–640, 2010.

[15] M.-R. Chen, X. Li, X. Zhang, and Y.-Z. Lu, "A novel particle swarm optimizer hybridized with extremal optimization," *Applied Soft Computing*, vol. 10, no. 2, pp. 367–373, 2010.

[16] H. S. Pannu, D. Singh, and A. K. Malhi, "Multi-objective particle swarm optimization-based adaptive neuro-fuzzy

inference system for benzene monitoring," *Neural Computing and Applications*, vol. 31, no. 7, pp. 2195–2205, 2019.

[17] H. S. Pannu, D. Singh, and A. K. Malhi, "Improved particle swarm optimization based adaptive neuro-fuzzy inference system for benzene detection," *Clean-Soil, Air, Water*, vol. 46, no. 5, Article ID 1700162, 2018.

[18] G. Sun and A. Zhang, "A hybrid genetic algorithm and gravitational search algorithm for image segmentation using multilevel thresholding,," in *Pattern Recognition and Image Analysis*, Springer, Heidelberg, Germany, 2013.

[19] M. Kaur, D. Singh, and R. Singh Uppal, "Parallel strength pareto evolutionary algorithm-II based image encryption," *IET Image Processing*, 2019.

[20] S. Jiang, Z. Ji, and Y. Shen, "A novel hybrid particle swarm optimization and gravitational search algorithm for solving economic emission load dispatch problems with various practical constraints," *International Journal of Electrical Power & Energy Systems*, vol. 55, pp. 628–644, 2014.

[21] S. Mallick, S. P. Ghoshal, P. Acharjee, and S. S. Thakur, "Optimal static state estimation using improved particle swarm optimization and gravitational search algorithm," *International Journal of Electrical Power & Energy Systems*, vol. 52, pp. 254–265, 2013.

[22] S. Mirjalili, S. Z. Mohd Hashim, and H. Moradian Sardroudi, "Training feedforward neural networks using hybrid particle swarm optimization and gravitational search algorithm," *Applied Mathematics and Computation*, vol. 218, no. 22, pp. 11125–11137, 2012.

[23] R. C. Green II, L. Wang, and M. Alam, "Training neural networks using central force optimization and particle swarm optimization: insights and comparisons," *Expert Systems with Applications*, vol. 39, no. 1, pp. 555–563, 2012.

[24] F. vandenBergh and A. P. Engelbrecht, "A cooperative approach to particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 225–239, 2004.

[25] R. Thangaraj, M. Pant, A. Abraham, and P. Bouvry, "Particle swarm optimization: hybridization perspectives and experimental illustrations," *Applied Mathematics and Computation*, vol. 217, no. 12, pp. 5208–5226, 2011.

[26] Y. Shi and R. C. Eberhart, "Empirical study of particle swarm optimization,," in *Proceedings of the 1999 Congress on Evolutionary Computation*, pp. 1945–1950, Washington, DC, USA, 1999.

[27] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 240–255, 2004.

[28] Y.-T. Kao and E. Zahara, "A hybrid genetic algorithm and particle swarm optimization for multimodal functions," *Applied Soft Computing*, vol. 8, no. 2, pp. 849–857, 2008.

[29] A. A. A. Esmin, G. Lambert-Torres, and G. B. Alvarenga, "Hybrid evolutionary algorithm based on PSO and GA mutation,," in *Proceedings of 6th International Conference on Hybrid Intelligent Systems*, pp. 57–62, Rio de Janeiro, Brazil, 2006.

[30] A. Shunmugalatha and S. M. R. Slochanal, "Optimum cost of generation for maximum load ability limit of power system using hybrid particle swarm optimization," *International Journal of Electrical Power & Energy Systems*, vol. 30, no. 8, pp. 486–490, 2008.

[31] W. J. Zhang and X. F. Xie, "DEPSO: Hybrid particle swarm with differential evolution operator," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics (SMCC)*, pp. 3816–3821, Washington, DC, USA, 2003.

[32] C. Zhang, J. Ning, S. Lu, D. Ouyang, and T. Ding, "A novel hybrid differential evolution and particle swarm optimization algorithm for unconstrained optimization," *Operations Research Letters*, vol. 37, no. 2, pp. 117–122, 2009.

[33] I. Mendialdua, A. Arruti, E. Jauregi, E. Lazkano, and B. Sierra, "Classifier Subset Selection to construct multi-classifiers by means of estimation of distribution algorithms," *Neurocomputing*, vol. 157, pp. 46–60, 2015.

[34] H. W. Ge, L. Sun, Y. C. Liang, and F. Qian, "An effective PSO and AIS-based hybrid intelligent algorithm for job-shop scheduling," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 38, no. 2, pp. 358–368, 2008.

[35] S. Jiang, Y. Wang, and Z. Ji, "A new design method for adaptive IIR system identification using hybrid particle swarm optimization and gravitational search algorithm," *Nonlinear Dynamics*, vol. 79, no. 4, pp. 2553–2576, 2015.

[36] W. H. Lim and N. A. Mat Isa, "Teaching and peer-learning particle swarm optimization," *Applied Soft Computing*, vol. 18, pp. 39–58, 2014.

[37] X. T. Li, M. Yin, and Z. Q. Ma, "Hybrid differential evolution and gravitation search algorithm for unconstrained optimization," *International Journal of Physical Sciences*, vol. 6, no. 25, pp. 5961–5981, 2011.

[38] H. Q. Chen, S. Li, and Z. Tang, "Hybrid gravitational search algorithm with random-key encoding scheme combined with simulated annealing," *International Journal of Computer Science and Network Security*, vol. 11, no. 6, pp. 208–217, 2011.

[39] S. Sarafrazi and H. Nezamabadi-pour, "Facing the classification of binary problems with a GSA-SVM hybrid system," *Mathematical and Computer Modelling*, vol. 57, no. 1-2, pp. 270–278, 2013.

[40] S. H. Jiang, C. L. Zhang, W. W. Wu, and Y. M. Li, "An improved hybrid particle swarm optimization with dependent random coefficients for global optimization," in *Proceedings of the 2018 33rd Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, pp. 666–672, Nanjing, China, 2018.

[41] H. Huang, H. Qin, Z. Hao, and A. Lim, "Example-based learning particle swarm optimization for continuous optimization," *Information Sciences*, vol. 182, no. 1, pp. 125–138, 2012.

[42] H. Wang, Z. J. Wu, S. Rahnamayan, Y. Liu, and M. Ventresca, "Enhancing particle swarm optimization using generalized opposition-based learning," *Information Sciences*, vol. 181, no. 2, pp. 4699–4714, 2011.

[43] S. García, D. Molina, M. Lozano, and F. Herrera, "A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 special session on real parameter optimization," *Journal of Heuristics*, vol. 15, no. 6, pp. 617–644, 2009.

Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 9464593, 11 pages
https://doi.org/10.1155/2020/9464593

Research Article

Improved Chaotic Quantum-Behaved Particle Swarm Optimization Algorithm for Fuzzy Neural Network and Its Application

Yuexi Peng,[1] Kejun Lei,[2] Xi Yang,[2] and Jinzhang Peng[3]

[1]*School of Physics and Electronics, Central South University, Changsha 410083, China*
[2]*College of Information Science and Engineering, Jishou University, Jishou 416000, China*
[3]*College of Physics and Mechatronics Engineering, Jishou University, Jishou 416000, China*

Correspondence should be addressed to Kejun Lei; 1258489458@qq.com

Received 7 February 2020; Accepted 5 March 2020; Published 28 March 2020

Guest Editor: Kehui Sun

Traditional fuzzy neural network has certain drawbacks such as long computation time, slow convergence rate, and premature convergence. To overcome these disadvantages, an improved quantum-behaved particle swarm optimization algorithm is proposed as the learning algorithm. In this algorithm, a new chaotic search is introduced, and benchmark function experiments prove it outperforms the other five existing algorithms. Finally, the proposed algorithm is presented as the learning algorithm for Takagi–Sugeno fuzzy neural network to form a new neural network, and it is utilized in the water quality evaluation of Dongjiang Lake of Hunan province. Simulation results demonstrated the effectiveness of the new neural network.

1. Introduction

Artificial neural network (ANN) [1] is an effective method to deal with nonlinear problems. Because of the powerful fitting ability, the ANN can nearly simulate any complex nonlinear functional relationship without knowing the correlation between the input and the output. At present, the ANN has been widely used in practical applications [2–5]. In fact, fuzzy neural network (FNN) [6] can handle complex engineering problem as well [7–9]. It combines the advantages of fuzzy mathematics and ANN. Due to the introduction of the fuzzy logic concept, it is very suitable for the classification of nonlinear or highly uncertain information. However, the FNN also inherits the disadvantages of the ANN, such as long computation time and slow convergence rate. Therefore, how to solve these shortcomings becomes a problem to be solved urgently [10–12].

It is an effective way to improve the FNN performance by replacing the traditional learning algorithm by metaheuristic algorithms such as the particle swarm optimization (PSO) algorithm [13–15]. However, when the considered problem is a complex high-dimensional problem, PSO algorithm has the disadvantage of premature convergence [16–19]. After studying the results of particle convergence behavior, Sun et al. [20] proposed a novel metaheuristic algorithm called quantum-behaved particle swarm optimization (QPSO) algorithm. It combines PSO algorithm with the quantum mechanic and has better global searching ability than that of the PSO algorithm [21, 22]. Since the QPSO algorithm was proposed, many researchers devoted to apply it for practical applications [23–26] or improve the algorithm itself [27–31]. For example, Mariani et al. [29] proposed a novel chaotic QPSO algorithm for the image matching, but this algorithm only introduces chaos variables into the particle position initialization. In fact, the effectiveness should be better if the chaos variables are introduced both at the beginning and the end of the algorithm search stage. Mariani et al. [29] proposed a QPSO algorithm combined with the Zaslavskii chaotic map and applied it to the optimization of shell and tube heat exchangers. Then, Turgut et al. [30] proposed another chaotic QPSO algorithm for solving nonlinear system of equations, and the benchmark function experiments are proved that the algorithm using Logistic chaotic map has the best performance. On the basis of Ref. [30], Turgut [31] proposed a hybrid chaotic QPSO algorithm for thermal design of plate fin heat exchangers. However, this algorithm is a little complicated for setting up three different populations to search simultaneously, and the convergence rate is not fast enough.

Therefore, to accelerate the convergence rate and improve the optimization precision further, an algorithm called improved chaotic quantum-behaved particle swarm optimization (ICQPSO) algorithm is proposed. Compared with the other chaotic PSO algorithms [17, 29–31], it is simple and easy to implement. The ICQPSO algorithm is introduced as the learning algorithm in the Takagi–Sugeno fuzzy neural network (TSFNN) [32], which can get accurate output value by inputting most forms of information, to form the improved neural network called ICQPSO-TSFNN. Finally, to demonstrate the effectiveness of the proposed method, we focus on the topic of the FNN for the water quality evaluation. The ICQPSO-TSFNN is applied to evaluate the water quality of the Dongjiang Lake in the Hunan Province from 2002 to 2013.

The rest of this paper is organized as follows. Section 2 mainly introduces the proposed ICQPSO algorithm. The performance test of ICQPSO algorithm are shown in Section 3. The application of ICQPSO-TSFNN for water quality evaluation is presented in Section 4. Finally, we summarize the results and indicate future directions.

2. ICQPSO Algorithm

The proposed ICQPSO algorithm aims at improving the performance of dealing with high-dimensional complex problems by introducing a new chaotic search method, and it also lays the foundation for the application of TSFNN in Section 4.

2.1. QPSO Algorithm. For the QPSO algorithm [20], the particle movement is completely different from that of the PSO algorithm [33]. Newtonian principles are invalid in quantum world as the velocity and position update cannot be determined simultaneously. Hence, there is no velocity vector in the particle of QPSO algorithm. The new state of each particle is determined by the wave function $\psi(\overrightarrow{x}, t)$, and there also exists $|\psi(t)|^2$ which is the probability density function of the position of each particle. Considering a one-dimensional optimization problem, the Monte Carlo stochastic simulation is employed to obtain the position equation of the particle i:

$$X_i(t+1) = p_i(t) \pm \frac{L_i(t)}{2}\ln\left(\frac{1}{\mu}\right), \qquad (1)$$

where $p_i(t)$ is the local attractor of the particle i ($i = 1, 2, \ldots, N$, N is the total number of particle swarm), and it is defined by

$$p_i(t) = \varphi P_{bi}(t) + (1-\varphi) P_g(t), \qquad (2)$$

$$\varphi = \frac{c_1 r_1}{c_1 r_1 + c_2 r_2}, \qquad (3)$$

where $P_{bi}(t)$ represents the optimal position of the particle i at the tth iteration and $P_g(t)$ represents the global optimal position in the particle swarm at the tth iteration. c_1 and c_2 are the learning factors, which provide the optimal selection function. r_1 and r_2 are the random numbers between $(0, 1)$. $L_i(t)$ is defined as

$$L_i(t+1) = 2\beta(t)\left|\text{mbest} - X_i(t)\right|, \qquad (4)$$

$$\text{mbest} = \sum_{i=1}^{N} \frac{P_{bi}}{N}, \qquad (5)$$

where $\beta(t)$ is the contraction expansion coefficient that determines the convergence rate of the algorithm [22], and it is calculated by

$$\beta(t) = \beta_{\min} + \frac{(T-t)(\beta_{\max} - \beta_{\min})}{T}, \qquad (6)$$

where t and T are current and maximum iteration number, respectively. According to Ref. [22], $\beta_{\max} = 1.0$ and $\beta_{\min} = 0.5$. The position equation of the particle i is updated by

$$X_i(t+1) = p_i(t) + \beta(t)\left|\text{mbest} - X_i(t)\right|\ln\left(\frac{1}{u}\right), \quad r_3 \geq 0.5, \qquad (7)$$

$$X_i(t+1) = p_i(t) - \beta(t)\left|\text{mbest} - X_i(t)\right|\ln\left(\frac{1}{u}\right), \quad r_3 < 0.5, \qquad (8)$$

where u and r_3 are generated according to a uniform probability distribution at the range (0, 1).

Assuming that the considered problem is a minimum optimization problem, the implementation of the QPSO algorithm is summarized as follows:

(a) Initialize particles in the population with random position vectors.

(b) Evaluate the fitness values of each particle.

(c) Calculate the local attractor point as defined in equation (2).

(d) Calculate *mbest* vector according to equation (5).

(e) Compare the fitness of the $P_i(t)$ with the fitness of the P_{bi}. If the fitness value of $P_i(t)$ is smaller than that of the P_{bi}, then replace the P_{bi} by the $P_i(t)$.

(f) Compare the fitness of the P_{bi} with the fitness of the P_g. If the fitness value of the P_b is smaller than that of the P_g, then replace the P_g by the P_{bi}.

(g) Update the position of the particles according to equation (7) or equation (8).

(h) Repeat Step (b) to Step (f) until the termination criteria are satisfied.

Since there is no velocity limit, the QPSO algorithm can jump out of the local optimum more easily. So, it has stronger global convergence ability than that of the PSO algorithm.

2.2. Chaotic Search. Chaos has inherent randomness and ergodicity [34, 35]. It allows the chaotic search to be programmed and traverses every state in a certain search region, while every state is visited only once. Therefore, introducing chaos sequence generated by the chaotic system into the QPSO algorithm can improve the algorithm performance [28–31]. Here, the logistic map is considered, and it is defined by

$$x(n+1) = \mu x(n)[1 - x(n)], \tag{9}$$

where μ is the control parameter. $x(n)$ is the system variable, and $x(n) \in (0,1)$. A different μ value leads to different dynamical behavior. If $\mu = 4$, the logistic map is chaotic, and the system variable trajectory is dense over the whole search space.

Based on the logistic map, the new chaotic search method is designed. First, the chaotic initialization is presented, and it can make the distribution of all the particles more uniform with respect to the random distribution of the QPSO algorithm. The particle position is initialized as

$$X(n+1) = X_{\min} + (X_{\max} - X_{\min})x(n), \quad n = 1, 2, \ldots, \tag{10}$$

where $x(n)$ is the system variable of the logistic map. $X_{\min}$ and $X_{\max}$ are the minimum and maximum boundary of particle position, respectively.

When the search stage of the QPSO algorithm is over, the chaotic search starts to apply and makes the particles jump out of the local optimal state quickly. The search range is determined by the current global optimal position, and the position of particle i in the chaotic search is updated by

$$\begin{aligned} X_i(n) = {} & \left[P_g(t) - aP_g(t)\right] + \left\{\left[P_g(t) + aP_g(t)\right]\right. \\ & \left. - \left[P_g(t) - aP_g(t)\right]\right\} x(n), \end{aligned} \tag{11}$$

where $P_g(t)$ is the global optimal position at the tth iteration, T is the maximum iteration number of QPSO algorithm, $x(n)$ is the chaotic variable at the nth $(n = 1, 2, \ldots, n_{\max})$ iteration, and $n_{\max}$ is the maximum iteration number of chaotic search. Chaotic search and QPSO algorithm are two independent processes. $a\,(0 < a < 1)$ is a fixed value which is set according to the requirement, and a small value a is helpful for the fast convergence of the algorithm. When the algorithm falls into the local optimal value, it is necessary to set a larger a to expand the range of chaotic search for jumping out of the local optimum. In this paper, chaotic search is defined as Algorithm 1.

2.3. Implementation of the ICQPSO Algorithm. The ICQPSO flowchart is presented, as shown in Figure 1, and the implementation steps are listed as follows:

(a) Chaotic initialize particle positions, and set the QPSO algorithm parameters.

(b) Evaluate the fitness value of each particle.

(c) Calculate local attractor point as defined by equation (2).

(d) Calculate the mbest vector according to equation (5).

(e) Compare the fitness of the $P_i(t)$ with the fitness of the P_{bi}. If the fitness of the $P_i(t)$ is smaller than that of the P_{bi}, then replace it.

(f) Compare the fitness of the P_{bi} with the fitness of the P_g. If the fitness of the P_{bi} is smaller than that of the P_g, then replace it.

(g) Update particle positions according to equation (7) or (8).

(h) Execute the chaotic search.

(i) Repeat Step (b) to Step (h) until the termination criteria are met.

3. Performance Tests of ICQPSO Algorithm

To assess the performance of the proposed ICQPSO algorithm, six benchmark functions are examined, and statistical results are compared with PSO [13], SINPSO [17], APSO [18], QPSO [21], and HCQPSO [31, 32]. QPSO and PSO are the basic traditional algorithms. SINPSO is a hybrid algorithm which combines PSO algorithm with chaotic ant colony algorithm. APSO is a variant PSO algorithm which has the adaptive inertia weight. HCQPSO is an improved QPSO algorithm which is introduced the chaotic sequence.

Here, Ackley, Rastrigin, and Griewank functions are high-dimensional multimodal functions, while Sphere, Rosenbrock, and Schwefel's 1.2 functions are high-dimensional unimodal functions. Formulations and properties of these unimodal and multimodal benchmark functions are listed in Table 1. The dimension of all data is set as 30, and the parameter settings for each algorithm are given in Table 2. Algorithms are developed in Matlab 2014b and run on Intel@Core TM with 2.80 GHz CPU and 4.0 GB RAM. Over 50 consecutive algorithm runs are performed due to the stochastic nature, and 1000 fitness values are calculated for each algorithm.

The statistical results of all function tests are shown in Table 3 and Figure 2. The results reveal that the ICQPSO algorithm is superior over other algorithms in terms of finding the minimum object. In Ackley, Sphere, and Schwefel's 1.2 function tests, only the ICQPSO finds the optimal value 0. In Rastrigin and Griewank function tests, both of the ICQPSO algorithm and HCQPSO algorithm find the optimal value 0, but the ICQPSO converges much faster than that of the HCQPSO. The Rosenbrock function is the most difficult to optimize that all algorithms cannot have the high precision solution; the ICQPSO is the second most efficient algorithm amongst others, just slightly worse than the HCQPSO algorithm. Although the precision of ICQPSO algorithm is lower than that of the HCQPSO algorithm in the Rosenbrock function, its convergence rate is always higher than that of the HCQPSO algorithm. Therefore, it can be concluded that the ICQPSO algorithm is the most efficient among the six algorithms. It lays the foundation for the following water quality evaluation experiment.

```
If $P_g(t) > 1$
   For $n = 1 : n_{\max}$
   $X(n) = [P_g(t) - 0.5P_g(t)] + \{[P_g(t) + 0.5P_g(t)] - [P_g(t) - 0.5P_g(t)]\}x(n)$
   Update $x(n)$ with equation (11)
   Calculate each fitness of $X(n)$, if it is smaller than $P_g(t)$, then replace the $P_g(t)$
Else
   For $n = 1 : n_{\max}$
   $X(n) = [P_g(t) - 0.01P_g(t)] + \{[P_g(t) + 0.01P_g(t)] - [P_g(t) - 0.01P_g(t)]\}x(n)$
   Update $x(n)$ with equation (11)
   Calculate each fitness of $X(n)$, if it is smaller than $P_g(t)$, then replace the $P_g(t)$
End
```

Algorithm 1: Chaotic search in the ICQPSO algorithm.

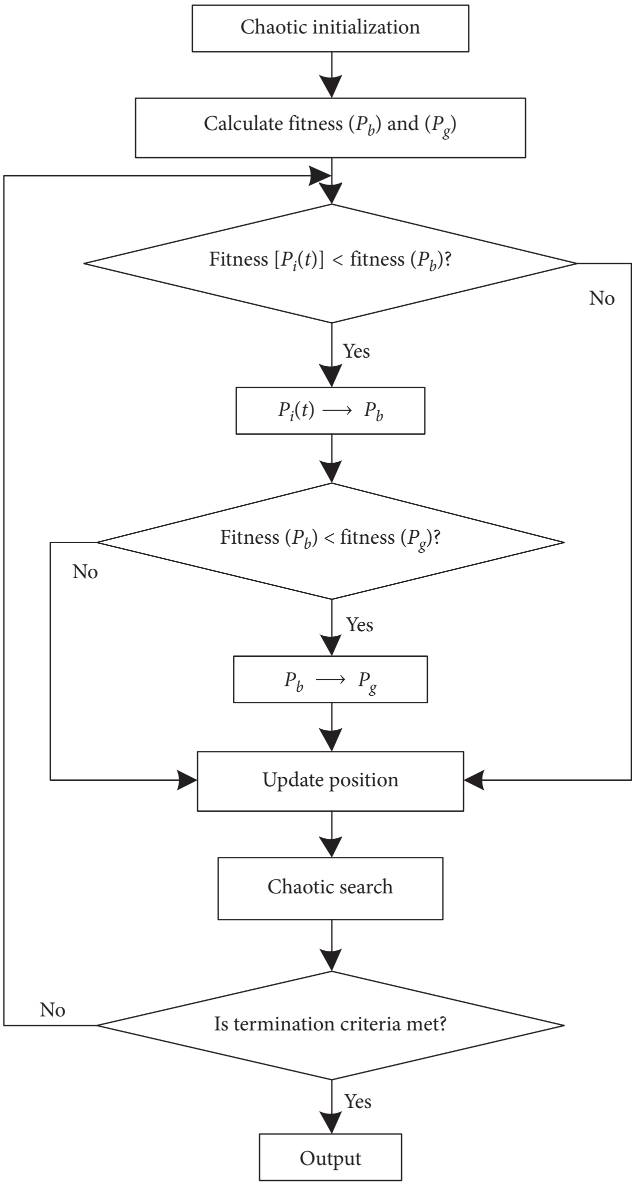

Chaotic initialization
Calculate fitness (P_b) and (P_g)
Fitness [$P_i(t)$] < fitness (P_b)?
No
Yes
$P_i(t) \longrightarrow P_b$
Fitness (P_b) < fitness (P_g)?
No
Yes
$P_b \longrightarrow P_g$
Update position
Chaotic search
No
Is termination criteria met?
Yes
Output

Figure 1: Flowchart of the proposed ICQPSO algorithm.

4. The Application of ICQPSO-TSFNN for Water Quality Evaluation

4.1. ICQPSO-TSFNN. To obtain better evaluation precision, the ICQPSO algorithm is used to replace the error correction learning (ECL) algorithm for TSFNN weights correction. In this paper, there is only one node in the output layer, which is the evaluation value of water quality. The node number of the input layer and hidden layer are n and m, respectively, so the architecture of the TSFNN is $n - m - 1$. The process of the proposed ICQPSO-TSFNN is described as follows:

(a) Set the initial parameters of the TSFNN and ICQPSO algorithm.

(b) Divide data into training sets and testing sets, all the data are normalized to ensure the quality of evaluation results.

(c) The ICQPSO algorithm is applied for the correction of TSFNN weights. Initialize particle positions in the ICQPSO algorithm, and the positions represent the center value, width value, and fuzzy system parameters, respectively.

(d) Input the training data and use TSFNN to calculate the fitness of each particle. Here, the fitness value is the mean square error (MSE) between the output values and the target ones.

(e) Calculate the fitness value and adjust the weights of TSFNN until the termination criteria are met.

(f) Use the trained TSFNN for the testing stage.

(g) If the testing results are satisfactory, output this TSFNN.

The training and testing flowchart of the ICQPSO-TSFNN are shown in Figure 3.

4.2. The Evaluation Criteria of Water Quality. The purpose of water quality evaluation is to determine the water quality rank, which is based on the collected water samples by a certain mathematical model. After considering the water

Table 1: The six benchmark functions.

Function name	Definition formula	Range	Optimal value
Ackley	$f(x) = -20\exp\left(-0.2\sqrt{(1/n)\sum_{i=1}^{n} x_i^2}\right) - \exp\left((1/n)\sum_{i=1}^{n}\cos(2\pi x_i)\right) + 20 + e$	[−32, 32]	0
Rastrigin	$f(x) = \sum_{i=1}^{n}[x_i^2 - 10 \quad \cos(2\pi x_i) + 10]$	[−5.12, 5.12]	0
Griewank	$f(x) = \sum_{i=1}^{n} x_i^2/4000 - \prod_{i=1}^{n}\cos(x_i/\sqrt{i}) + 1$	[−600, 600]	0
Sphere	$f(x) = \sum_{i=1}^{n} x_i^2$	[−100, 100]	0
Rosenbrock	$f(x) = \sum_{i=1}^{n}[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	[−30, 30]	0
Schwefel's 1.2	$f(x) = \sum_{i=1}^{n}(\sum_{j=1}^{i} x_j)^2$	[−100, 100]	0

Table 2: Algorithm parameter settings.

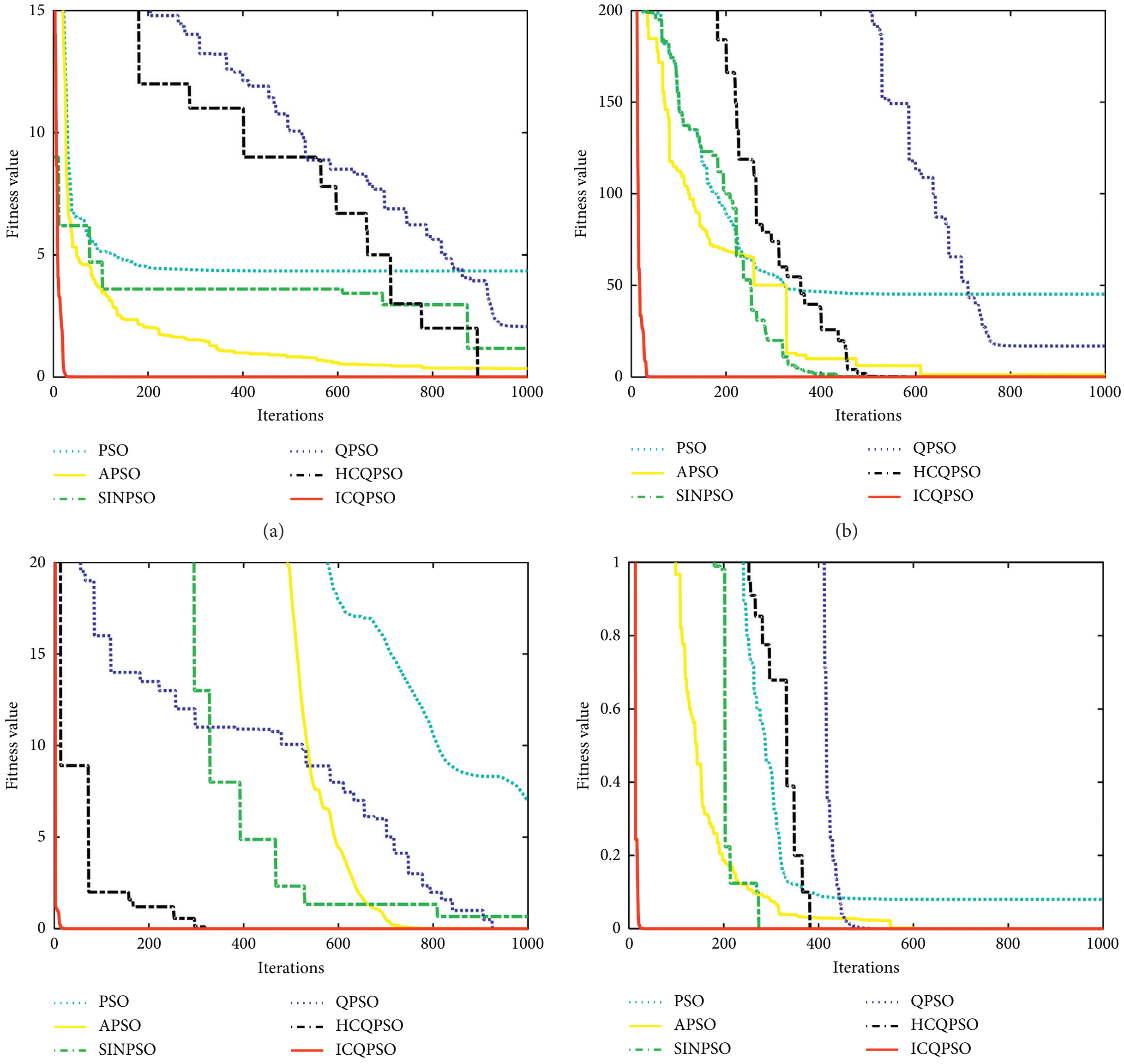
Algorithm	Parameter settings
PSO	$N = 25$, $c_1 = c_2 = 1.4962$, $V_{\max} = 1$, $V_{\min} = -1$, $\omega = 0.7298$
APSO	$N = 25$, $c_1 = c_2 = 2$, $\alpha = 0.1$
SINPSO	$N = 25$, $c_1 = c_2 = 1.4962$, $V_{\max} = 1$, $V_{\min} = -1$, $\omega = 0.7298$ $r(i,d) = 0.5 + (0.005)\text{rand}$, $C_{i\quad d}(0) = 0.999$, $\psi_d = 10.42$, $M_i = 0.5$
QPSO	$N = 25$, $c_1 = c_2 = 2$, $\beta_{\max} = 1.0$, $\beta_{\min} = 0.5$
HCQPSO	$N = 25$, $c_1 = c_2 = 2$, $\beta_{\max} = 1.0$, $\beta_{\min} = 0.5$
ICQPSO	$N = 25$, $c_1 = c_2 = 2$, $\beta_{\max} = 1.0$, $\beta_{\min} = 0.5$, $n_{\max} = 100$

Table 3: Statistical results of different test function.

	Best	Mean	Worst
Ackley			
PSO	$2.0861e+00$	$3.5337e+00$	$5.0738e+00$
APSO	$2.6173e-01$	$1.5059e+00$	$2.5767e+00$
SINPSO	$7.5123e-02$	$1.5673e+00$	$2.2246e+00$
QPSO	$4.8376e-08$	$3.1851e+00$	$5.5752e+00$
HCQPSO	$8.8888e-16$	$8.8888e-16$	$8.8888e-16$
ICQPSO	**0**	$\mathbf{3.3159e-15}$	$\mathbf{3.5527e-15}$
Rastrigin			
PSO	$4.4773e+01$	$5.2945e+01$	$6.3585e+01$
APSO	$3.7384e-01$	$1.6870e+01$	$2.2759e+01$
SINPSO	$3.3364e-04$	$4.1799e-04$	$4.5238e-03$
QPSO	$1.5946e+00$	$2.5262e+00$	$3.4837e+00$
HCQPSO	**0**	**0**	**0**
ICQPSO	0	0	0
Griewank			
PSO	$1.1751e+00$	$3.7878e+00$	$7.9665e+00$
APSO	$4.8418e-03$	$1.0163e-03$	$1.1055e-01$
SINPSO	$8.9135e-01$	$9.8711e-01$	$9.9887e-01$
QPSO	0	$2.4653e-04$	$7.3144e-03$
HCQPSO	0	0	0
ICQPSO	**0**	**0**	**0**
Sphere			
PSO	$7.9163e-03$	$2.2771e-02$	$4.9712e-02$
APSO	$2.2724e-14$	$6.9874e-12$	$6.5521e-11$
SINPSO	$1.2264e-09$	$3.2379e-10$	$4.2347e-10$
QPSO	$4.3948e-19$	$3.2996e-15$	$8.1298e-14$
HCQPSO	$4.3454e-39$	$1.1921e-34$	$8.4116e-33$
ICQPSO	$\mathbf{7.9196e-91}$	$\mathbf{5.3151e-82}$	$\mathbf{6.0556e-81}$
Rosenbrock			
PSO	$2.4057e+01$	$5.4534e+01$	$1.0105e+02$
APSO	$1.1128e+01$	$1.4055e+01$	$3.0779e+01$
SINPSO	$8.5902e+00$	$1.0317e+01$	$1.1547e+01$
QPSO	$1.2473e+01$	$4.0305e+01$	$1.2017e+02$
HCQPSO	$3.5643e+00$	$4.6541e+00$	$8.3287e+00$
ICQPSO	$\mathbf{4.0444e+00}$	$\mathbf{5.2713e+00}$	$\mathbf{6.5344e+00}$

Table 3: Continued.

	Best	Mean	Worst
Schwefel's 1.2			
PSO	$5.3828e+01$	$1.4982\ ee+02$	$3.2404e+02$
APSO	$2.1917e+00$	$9.0975e+00$	$2.1640e+01$
SINPSO	$3.5402e+00$	$8.1004e+00$	$1.2023e+01$
QPSO	$1.2287e+00$	$5.5542e+00$	$7.7122e+00$
HCQPSO	$1.6231e-14$	$1.0412e+00$	$3.2824e+00$
ICQPSO	$\mathbf{1.3000e-24}$	$\mathbf{1.0612e-01}$	$\mathbf{8.0364e+00}$



(a)

(b)

(c)

(d)

Figure 2: Continued.

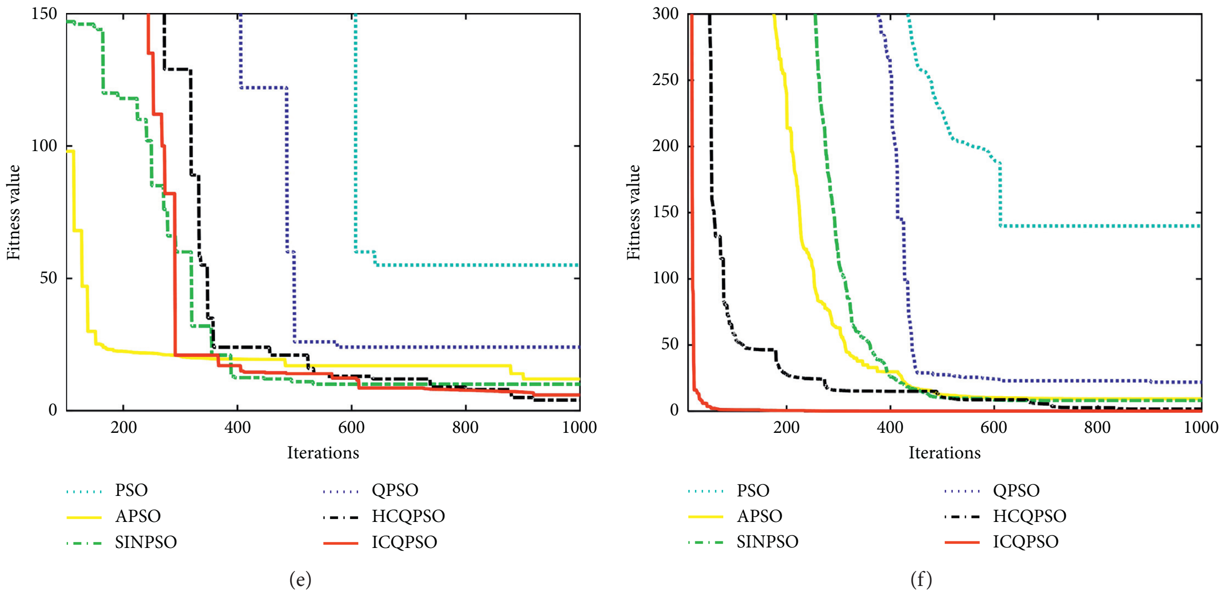
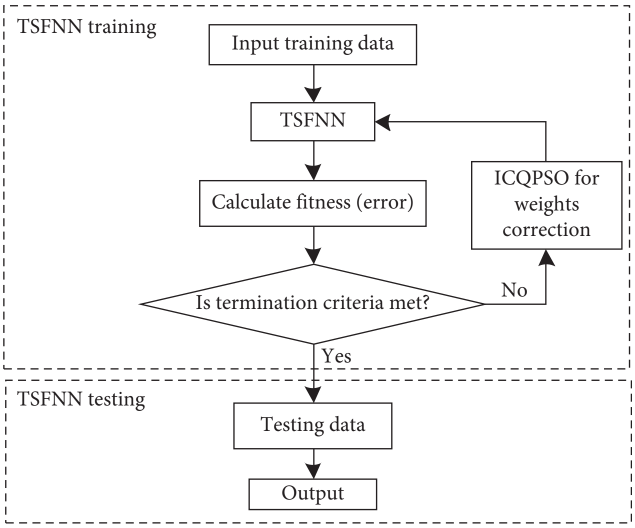


Figure 2: Convergence properties of algorithms: (a) Ackley, (b) Rastrigin, (c) Griewank, (d) Sphere, (e) Rosenbrock, and (f) Schwefel's 1.2.



TSFNN training
Input training data
TSFNN
Calculate fitness (error)
ICQPSO for weights correction
Is termination criteria met?
No
Yes
TSFNN testing
Testing data
Output

Figure 3: Training and testing flowchart of the proposed ICQPSO-TSFNN.

quality monitoring data analysis, we mainly choose the following six indicators as evaluation parameters: ammonia nitrogen, dissolved oxygen, chemical oxygen demand, permanganate index, total phosphorus, and total nitrogen. The water environmental quality index is listed in Table 4. We judged the water quality rank by the TSFNN output values. The classification criteria of water quality rank are shown in Table 5.

4.3. Training and Testing Samples. The experimental data are derived from the environmental hydrological data of the Dongjiang Lake watershed in Zixing, Hunan Province, from 2002 to 2013. 400 datasets from 2002 to 2007 are selected for the training and testing stages. 350 datasets are chosen as the training samples, and the other 50 datasets are taken as the testing samples. Finally, we also select 72 datasets from the Chukou, Dongping, and Bailang hydrological stations from 2008 to 2013. These data are evaluated by the well-trained TSFNN to demonstrate its practicality. The 10 datasets in one of the stations are shown in Table 6.

The water quality evaluation is carried out by the PSO-BP neural network [36] and three TSFNNs which are based on the ECL algorithm, QPSO algorithm, and ICQPSO algorithm, respectively. To verify the effectiveness of the proposed method, the same training and testing samples are used for these neural networks. All the TSFNN architecture is 6-12-1, and BP's architecture is 6-5-1. Set the algorithm population size $N = 25$. Through experiments, we noticed that when the training error is less than $1e-04$, there is almost no error in the water quality rank of the neural network prediction and the actual water quality rank. Therefore, we set the termination criteria as when the training error is less than $1e-04$, or the maximum number of iterations $T = 100$.

4.4. Result Analysis. The results of training stage are shown in Figure 4. Obviously, the convergence rate of the ICQPSO-TSFNN is quicker than that of the other three neural networks. The testing results are given in Table 7. As it is shown, the ICQPSO-TSFNN only needs 19 iterations to reach the termination criteria, but the other three networks still cannot reach it until the maximum iteration is arrived. So, the ICQPSO-TSFNN has much less training runtime than that of the PSO-BP and QPSO-TSFNN, except the ECL-TSFNN. However, the precision of the ICQPSO-TSFNN is much higher than that of the ECL-TSFNN. Finally, from Figure 5, it is shown that the ICQPSO-TSFNN still has the lowest MSE at the testing stage. Therefore, it can be concluded that the ICQPSO-TSFNN is the most efficient among the four neural networks.

Table 4: Chinese surface water environmental quality index (unit: mg/L).

Items	Rank I	Rank II	Rank III	Rank IV	Rank V
Ammonia nitrogen ≤	0.150	0.500	1.000	1.500	2.000
Dissolved oxygen ≥	7.500	6.000	5.000	3.000	2.000
Chemical oxygen demand ≤	15.000	15.000	20.000	30.000	40.000
Permanganate index ≤	2.000	4.000	6.000	10.000	15.000
Total phosphorus ≤	0.020	0.100	0.200	0.300	0.400
Total nitrogen ≤	0.200	0.500	1.000	1.000	2.000
Applicable places	Source of water, national nature reserve	Life drinking water, surface water source protection area, rare aquatic habitat, fish and shrimp production field, etc	Life drinking water, surface water secondary protected areas, fish and shrimp wintering areas, etc	General industrial water and nondirect contact with the human body recreational water	Agricultural water areas and general landscape water

Table 5: Classification criteria of the water quality rank.

Items	Rank I	Rank II	Rank III	Rank IV	Rank V
Output value	(0, 1.5)	[1.5, 2.5)	[2.5, 3.5)	[3.5, 4.5)	≤4.5

Table 6: Data samples (unit: mg/L).

Items	1	2	3	4	5	6	7	8	9	10
Ammonia nitrogen	0.104	0.082	0.182	0.097	0.186	0.186	0.185	1.990	0.502	0.335
Dissolved oxygen	7.190	8.260	8.130	7.860	7.230	7.230	7.350	7.020	6.000	6.700
Chemical oxygen demand	6.600	8.320	10.000	11.26	9.210	9.210	15.000	10.810	9.570	9.150
Permanganate index	3.180	2.550	2.130	1.980	3.260	3.260	2.200	1.900	1.120	1.060
Total phosphorus	0.038	0.014	0.010	0.024	0.046	0.046	0.012	0.044	0.031	0.010
Total nitrogen	0.242	0.132	0.254	0.342	0.156	0.175	0.185	0.196	0.161	0.165

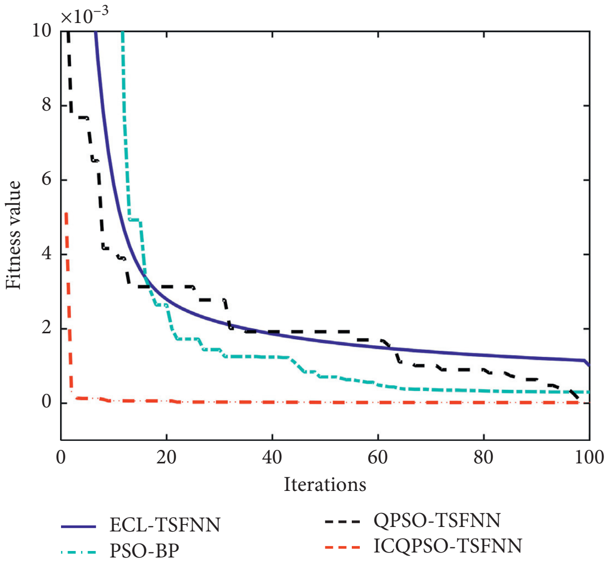
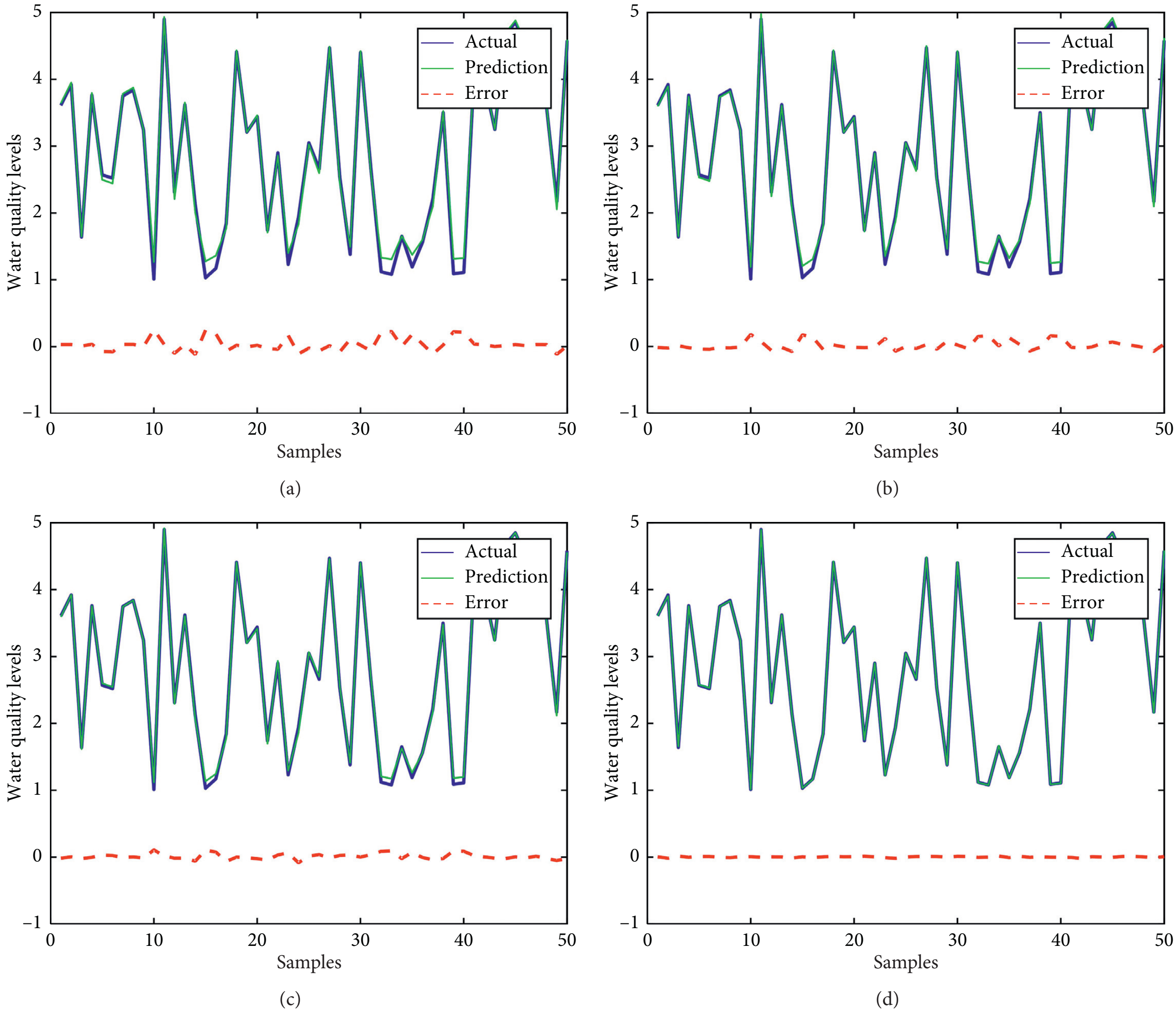


Figure 4: Fitness value at the training stage.

Table 7: Testing results of different models.

Networks	Training MSE	Training runtime (s)	Iterations	Testing MSE
ECL-TSFNN	$7.6224e-03$	22.1163	100	$5.4745e-02$
PSO-BP	$1.9243e-03$	39.2283	100	$9.6671e-03$
QPSO-TSFNN	$6.0541e-04$	57.4561	100	$4.3438e-03$
ICQPSO-TSFNN	$\mathbf{9.6106e-05}$	**24.2267**	**19**	$\mathbf{8.5183e-05}$



(a)

(b)

(c)

(d)

Figure 5: Testing stage error.

After the training and testing, the four networks are used to evaluate the water quality rank of Chukou, Dongping and Bailang hydrological stations form 2008 to 2013. The results are given in Table 8, and it shows that the ICQPSO-TSFNN only has 1 error, while the other three networks have 31, 21, and 16 errors, respectively. Therefore, the proposed ICQPSO-TSFNN has remarkable improvement for water quality evaluation, and it is more suitable for processing the daily hydrological information.

Table 8: Evaluation results for different TSFNN.

Monitoring stations	ECL-TSFNN	PSO-BP	QPSO-TSFNN	ICQPSO-TSFNN
Chukou	11	5	7	**0**
Dongping	11	10	5	**1**
Bailang	9	7	4	**0**
Total errors	31	22	16	**1**

5. Conclusion

In this paper, we focus on the improvement of FNN learning algorithm and its application performance. An improved algorithm called ICQPSO is proposed. It is a hybrid algorithm which combined the QPSO algorithm with a new chaotic search. Six benchmark function experiments are performed in six existing metaheuristic algorithms. The results show that the ICQPSO algorithm has the best performance. Finally, the new algorithm is introduced in TSFNN to form a new neural network called ICQPSO-TSFNN, and it is used to evaluate the water quality of the Dongjiang Lake. The results demonstrate that the new neural network is more efficient than that of three other neural networks. Therefore, it is concluded that the ICQPSO-TSFNN is feasible and effective. The future work is to apply the proposed algorithm to solve other complex nonlinear engineering problems.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant nos. 61861019 and 61362018), the Natural Science Foundation of Hunan Province of China (Grant nos. 2019JJ50483 and 2019JJ50484), the project of Hunan Provincial Department of Education (Grant nos. 18B316 and 18C0557), and the Key Innovation Project of Graduate Student of Central South University (Grant no. 2018ZZTS009).

References

[1] S. G. Rao and A. R. Rao, *Artificial Neural Networks in Hydrology*, Springer Press, Amsterdam, Netherlands, 2000.

[2] L. Kalin, S. Isik, J. E. Schoonover, and B. G. Lockaby, “Predicting water quality in unmonitored watersheds using artificial neural networks,” *Journal of Environmental Quality*, vol. 39, no. 4, pp. 1429–1440, 2010.

[3] G. Carleo and M. Troyer, “Solving the quantum many-body problem with artificial neural networks,” *Science*, vol. 355, no. 6325, pp. 602–606, 2017.

[4] G. W. Kim and K. Y. Lee, “Artificial neural networks-based machine learning for wireless networks: a tutorial,” *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3039–3071, 2019.

[5] A. Tavanaei, M. Ghodrati, S. R. Kheradpisheh, T. Masquelier, and A. Maida, “Deep learning in spiking neural networks,” *Neural Networks*, vol. 111, pp. 47–63, 2019.

[6] H. K. Kwan and Y. Cai, “A fuzzy neural network and its application to pattern recognition,” *IEEE Transactions on Fuzzy Systems*, vol. 2, no. 3, pp. 185–193, 1994.

[7] E. Yucel, M. Syed Ali, N. Gunasekaran, and S. Arik, “Sampled-data filtering of Takagi-Sugeno fuzzy neural networks with interval time-varying delays,” *Fuzzy Sets and Systems*, vol. 316, pp. 69–81, 2017.

[8] D. Krleza and K. Fertalj, “Graph matching using hierarchical fuzzy graph neural networks,” *IEEE Transactions on Fuzzy Systems*, vol. 25, no. 4, pp. 892–904, 2017.

[9] P. Shi, Y. Zhang, M. Chadli, and R. K. Agarwal, “Mixed H-infinity and passive filtering for discrete fuzzy neural networks with stochastic jumps and time delays,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 4, pp. 903–909, 2016.

[10] Y. Hou, L. Zhao, and H. Lu, “Fuzzy neural network optimization and network traffic forecasting based on improved differential evolution,” *Future Generation Computer Systems*, vol. 81, pp. 425–432, 2018.

[11] B. Pizzileo, K. Kang Li, G. W. Irwin, and W. Zhao, “Improved structure optimization for fuzzy-neural networks,” *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 6, pp. 1076–1089, 2012.

[12] H. Han, L. Zhang, X. Wu, and J. Qiao, “An efficient second-order algorithm for self-organizing fuzzy neural networks,” *IEEE Transactions on Cybernetics*, vol. 49, no. 13, pp. 1–13, 2017.

[13] H. Peng, F. Liu, and Z. R. Xu, “Variable universe fuzzy control for vehicle semi-active suspension system with MR damper combining fuzzy neural network and particle swarm optimization,” *Neurocomputing*, vol. 306, pp. 130–140, 2018.

[14] C.-C. Peng and C.-H. Chen, “Compensatory neural fuzzy network with symbiotic particle swarm optimization for temperature control,” *Applied Mathematical Modelling*, vol. 39, no. 1, pp. 383–395, 2015.

[15] R. Cheng and Y. Bai, “A novel approach to fuzzy wavelet neural network modeling and optimization,” *International Journal of Electrical Power & Energy Systems*, vol. 64, pp. 671–678, 2015.

[16] W. D. Annicchiarico and M. Cerrolaza, “Improved dynamical particle swarm optimization method for structural dynamics,” *Mathematical Problems in Engineering*, vol. 2019, Article ID 8250185, 11 pages, 2019.

[17] X. B. Xu, K. F. Zheng, L. I. Dan et al., “New chaos-particle swarm optimization algorithm,” *Journal of Communications*, vol. 33, no. 1, pp. 24–16, 2012.

[18] A. Adeli and A. Broumandnia, “Image steganalysis using improved particle swarm optimization based feature selection,” *Applied Intelligence*, vol. 48, no. 6, pp. 1609–1622, 2018.

[19] M. K. Marichelvam, M. Geetha, and Ö. Tosun, “An improved particle swarm optimization algorithm to solve hybrid flowshop scheduling problems with the effect of human

factors—a case study," *Computers & Operations Research*, vol. 114, Article ID 104812, 2020.

[20] J. Sun, B. Feng, and W. B. Xu, "Particle swam optimization with particles having quantum behavior," in *Proceedings of the IEEE Conference on Evolutionary Computation*, pp. 325–331, Portland, OR, USA, June 2004.

[21] W. Fang, J. Sun, Y. Ding, X. Wu, and W. Xu, "A review of quantum-behaved particle swarm optimization," *IETE Technical Review*, vol. 27, no. 4, pp. 336–348, 2010.

[22] J. Sun, W. Fang, X. Wu, V. Palade, and W. Xu, "Quantum-behaved particle swarm optimization: analysis of individual particle behavior and parameter selection," *Evolutionary Computation*, vol. 20, no. 3, pp. 349–393, 2012.

[23] Y. Li, L. Jiao, R. Shang, and R. Stolkin, "Dynamic-context cooperative quantum-behaved particle swarm optimization based on multilevel thresholding applied to medical image segmentation," *Information Sciences*, vol. 294, pp. 408–422, 2015.

[24] M. Xu, L. Zhang, B. Du, L. Zhang, Y. Fan, and D. Song, "A mutation operator accelerated quantum-behaved particle swarm optimization algorithm for hyperspectral endmember extraction," *Remote Sensing*, vol. 9, no. 3, p. 197, 2017.

[25] C. Huang, D. Zhang, and G. Song, "A novel mapping algorithm for three-dimensional network on chip based on quantum-behaved particle swarm optimization," *Frontiers of Computer Science*, vol. 11, no. 4, pp. 622–631, 2017.

[26] C. T. Cheng, W. J. Niu, Z. K. Feng et al., "Daily reservoir runoff forecasting method using artificial neural network based on quantum-behaved particle swarm optimization," *Water*, vol. 7, no. 8, pp. 4232–4246, 2015.

[27] J. Yang and J. Xie, "An improved quantum-behaved particle swarm optimization algorithm," *Applied Intelligence*, vol. 40, no. 3, pp. 479–496, 2014.

[28] F. Liu, H. Duan, and Y. Deng, "A chaotic quantum-behaved particle swarm optimization based on lateral inhibition for image matching," *Optik*, vol. 123, no. 21, pp. 1955–1960, 2012.

[29] V. C. Mariani, A. R. K. Duck, F. A. Guerra, L. D. S. Coelho, and R. V. Rao, "A chaotic quantum-behaved particle swarm approach applied to optimization of heat exchangers," *Applied Thermal Engineering*, vol. 42, no. 4, pp. 119–128, 2012.

[30] O. E. Turgut, M. S. Turgut, and M. T. Coban, "Chaotic quantum behaved particle swarm optimization algorithm for solving nonlinear system of equations," *Computers & Mathematics with Applications*, vol. 68, no. 4, pp. 508–530, 2014.

[31] O. E. Turgut, "Hybrid chaotic quantum behaved particle swarm optimization algorithm for thermal design of plate fin heat exchangers," *Applied Mathematical Modelling*, vol. 40, no. 1, pp. 50–69, 2016.

[32] Y. Wang, C. Chien, and C. Teng, "Takagi-Sugeno recurrent fuzzy neural networks for identification and control of dynamic systems," *IEEE Transactions on Fuzzy Systems*, vol. 8, no. 4, pp. 349–366, 2001.

[33] J. Kennedy and R. C. Eberhart, "Particle swarms optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, Perth, WA, Australia, December 1995.

[34] C. Li, B. Feng, S. Li, J. Kurths, and G. Chen, "Dynamic analysis of digital chaotic maps via state-mapping networks," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 6, pp. 2322–2335, 2019.

[35] X. Zhao, J. Liu, H. Liu, and F. Zhang, "Dynamic analysis of a one-parameter chaotic system in complex field," *IEEE Access*, vol. 8, no. 1, pp. 28774–28781, 2020.

[36] C. Liu, W. Ding, Z. Li et al., "Prediction of high-speed grinding temperature of titanium matrix composites using BP neural network based on PSO algorithm," *The International Journal of Advanced Manufacturing Technology*, vol. 89, no. 5–8, pp. 2277–2285, 2017.