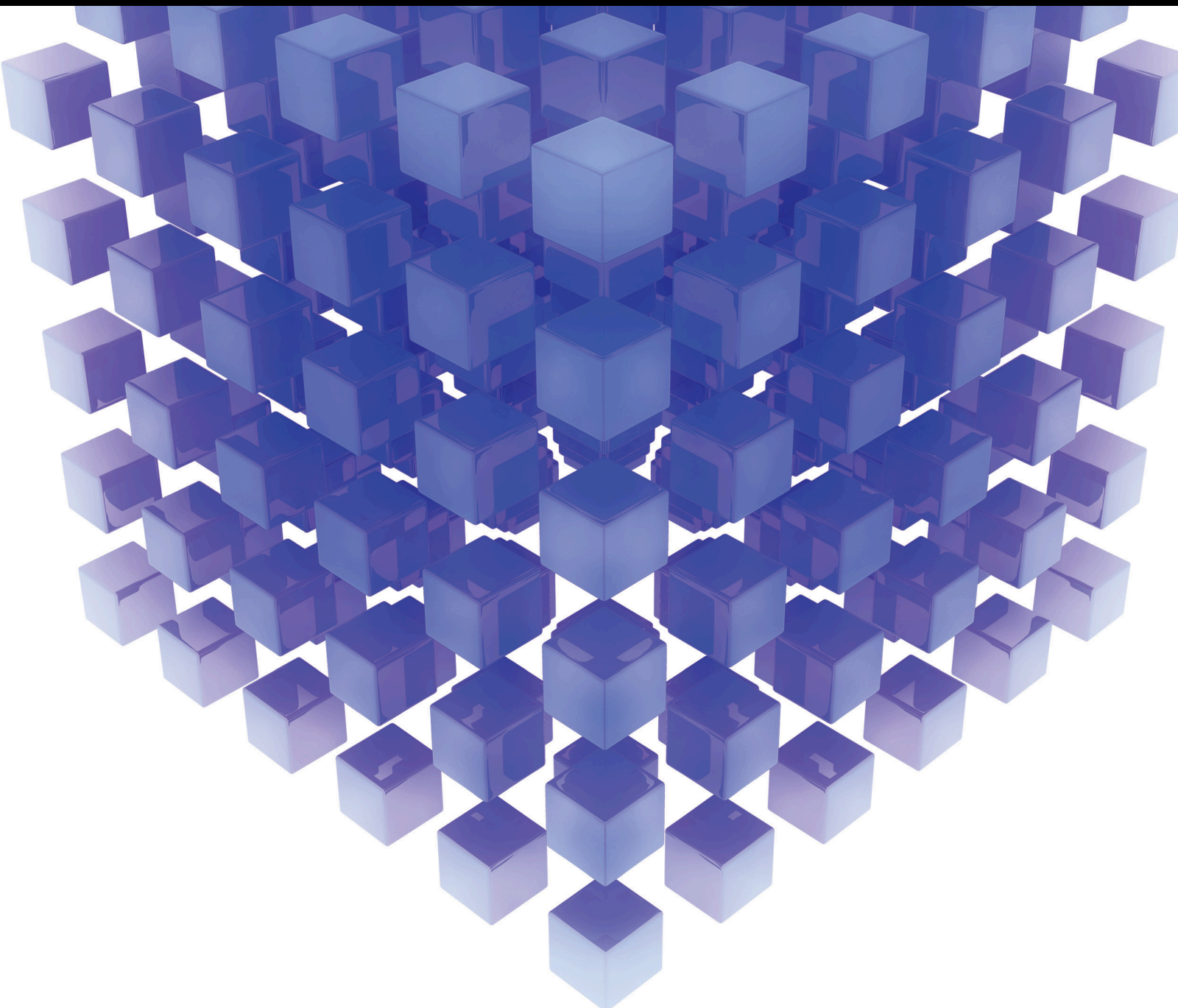


Nature-Inspired Intelligence Methods and Applications

Special Issue Editor in Chief: Qingzheng Xu

Guest Editors: Ana Maria A. C. Rocha, Erik Cuevas, and Iztok Fister Jr.





Nature-Inspired Intelligence Methods and Applications

Mathematical Problems in Engineering

Nature-Inspired Intelligence Methods and Applications

Special Issue Editor in Chief: Qingzheng Xu

Guest Editors: Ana Maria A. C. Rocha, Erik Cuevas,
and Iztok Fister Jr.



Copyright © 2022 Hindawi Limited. All rights reserved.

This is a special issue published in “Mathematical Problems in Engineering.” All articles are open access articles distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Chief Editor

Guangming Xie, China

Editorial Board

Dr. Kumaravel A, India
Waqas Abbasi, Pakistan
Mohamed Abd El Aziz, Egypt
Ahmed A. Abd El-Latif, Egypt
Mahmoud Abdel-Aty, Egypt
Mohammed S. Abdo, Yemen
Mohammad Yaghoub Abdollahzadeh
Jamalabadi, Republic of Korea
Rahib Abiyev, Turkey
Leonardo Acho, Spain
José Ángel Acosta, Spain
Daniela Addressi, Italy
Paolo Addresso, Italy
Claudia Adduce, Italy
Ramesh Agarwal, USA
Francesco Aggogeri, Italy
Ricardo Aguilar-Lopez, Mexico
Shabir Ahmad, Pakistan
Ali Ahmadian, Malaysia
Naveed Ahmed, Pakistan
Tarek Ahmed-Ali, France
Elias Aifantis, USA
Akif Akgul, Turkey
Guido Ala, Italy
Andrea Alaimo, Italy
Reza Alam, USA
Osamah Albahri, Malaysia
Nicholas Alexander, United Kingdom
Salvatore Alfonzetti, Italy
Nouman Ali, Pakistan
Ghous Ali, Pakistan
Dr. Jehad Ali, Republic of Korea
Mohammad D. Aliyu, Canada
Mohammed Almalahi, Yemen
Juan A. Almendral, Spain
Watheq Al-Mudhafar, Iraq
A.K. Alomari, Jordan
Tareq Al-shami, Yemen
Ali Saleh Alshomrani, Saudi Arabia
José Domingo Álvarez, Spain
Cláudio Alves, Portugal
Juan P. Amezcua-Sanchez, Mexico
Lionel Amodeo, France
Sebastian Anita, Romania

Renata Archetti, Italy
Muhammad Arif, Pakistan
Sabri Arik, Turkey
Francesco Aristodemo, Italy
Fausto Arpino, Italy
Alessandro Arsie, USA
Edoardo Artioli, Italy
Rashad Asharabi, Saudi Arabia
Farhad Aslani, Australia
Mohsen Asle Zaeem, USA
Andrea Avanzini, Italy
Richard I. Avery, USA
Viktor Avrutin, Germany
Mohammed A. Awadallah, Malaysia
Muhammad Uzair Awan, Pakistan
Francesco Aymerich, Italy
Sajad Azizi, Belgium
Michele Bacciocchi, Italy
Seungik Baek, USA
Khaled Bahlali, France
M.V.A Raju Bahubalendruni, India
Pedro Balaguer, Spain
P. Balasubramaniam, India
Stefan Balint, Romania
Ines Tejado Balsera, Spain
Alfonso Banos, Spain
Jerzy Baranowski, Poland
Tudor Barbu, Romania
Andrzej Bartoszewicz, Poland
Sergio Baselga, Spain
S. Caglar Baslamisli, Turkey
David Bassir, France
Chiara Bedon, Italy
Azeddine Beghdadi, France
Andriette Bekker, South Africa
Francisco Beltran-Carbajal, Mexico
Abdellatif Ben Makhlof, Saudi Arabia
Denis Benasciutti, Italy
Ivano Benedetti, Italy
Rosa M. Benito, Spain
Elena Benvenuti, Italy
Giovanni Berselli, Italy
Giorgio Besagni, Italy
Michele Betti, Italy

Pietro Bia, Italy
Carlo Bianca, France
Vittorio Bianco, Italy
Vincenzo Bianco, Italy
Simone Bianco, Italy
David Bigaud, France
Sardar Muhammad Bilal, Pakistan
Antonio Bilotta, Italy
Dr. Kishore Bingi, India
Sylvio R. Bistafa, Brazil
Bartłomiej Błachowski, Poland
Chiara Boccaletti, Italy
Guido Bolognesi, United Kingdom
Rodolfo Bontempo, Italy
Alberto Borboni, Italy
Marco Bortolini, Italy
Paolo Boscariol, Italy
Daniela Boso, Italy
Guillermo Botella-Juan, Spain
Boulaïd Boulkroune, Belgium
Abdesselem Boulkroune, Algeria
Fabio Bovenga, Italy
Francesco Braghin, Italy
Ricardo Branco, Portugal
Maurizio Brocchini, Italy
Julien Bruchon, France
Matteo Bruggi, Italy
Michele Brun, Italy
Maria Elena Bruni, Italy
Vasilis Burganos, Greece
Maria Angela Butturi, Italy
Dhanamjayulu C, India
Raquel Caballero-Águila, Spain
Guillermo Cabrera-Guerrero, Chile
Filippo Cacace, Italy
Pierfrancesco Cacciola, United Kingdom
Salvatore Caddemi, Italy
zuowei cai, China
Roberto Caldelli, Italy
Alberto Campagnolo, Italy
Eric Campos, Mexico
Salvatore Cannella, Italy
Francesco Cannizzaro, Italy
Maosen Cao, China
Javier Cara, Spain
Raffaele Carli, Italy
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
Nicola Caterino, Italy
Alberto Cavallo, Italy
Gabriele Cazzulani, Italy
Luis Cea, Spain
Fatih Vehbi Celebi, Turkey
Song Cen, China
Miguel Cerrolaza, Venezuela
M. Chadli, France
Gregory Chagnon, France
Ludovic Chamoin, France
Xiaoheng Chang, China
Qing Chang, USA
Ching-Ter Chang, Taiwan
Kuei-Lun Chang, Taiwan
Dr. Prasenjit Chatterjee, India
Kacem Chehdi, France
Peter N. Cheimets, USA
Chih-Chiang Chen, Taiwan
Shyi-Ming Chen, Taiwan
Xinkai Chen, Japan
Xizhong Chen, Ireland
Xue-Bo Chen, China
Kebing Chen, China
Mengxin Chen, China
Xiao Chen, China
He Chen, China
Chien-Ming Chen, China
Zhiwen Chen, China
Zeyang Cheng, China
Qiang Cheng, USA
Luca Chiapponi, Italy
Ryoichi Chiba, Japan
Francisco Chicano, Spain
Nicholas Chileshe, Australia
Tirivanhu Chinyoka, South Africa
Adrian Chmielewski, Poland
Seongim Choi, USA
Dr Gautam Choubey, India
Ioannis T. Christou, Greece

Hung-Yuan Chung, Taiwan
Yusheng Ci, China
Simone Cinquemani, Italy
Roberto G. Citarella, Italy
Joaquim Ciurana, Spain
John D. Clayton, USA
Francesco Clementi, Italy
Piero Colajanni, Italy
Giuseppina Colicchio, Italy
Vassilios Constantoudis, Greece
Francesco Conte, Italy
Enrico Conte, Italy
Alessandro Contento, USA
Mario Cools, Belgium
Gino Cortellessa, Italy
Juan Carlos Cortés, Spain
Carlo Cosentino, Italy
Paolo Crippa, Italy
Erik Cuevas, Mexico
Guozeng Cui, China
Maria C. Cunha, Portugal
Mehmet Cunkas, Turkey
Peter Dabnichki, Australia
Luca D'Acierno, Italy
Weizhong Dai, USA
Zhifeng Dai, China
Pei Dai, China
Purushothaman Damodaran, USA
Bhabani S. Dandapat, India
Giuseppe D'Aniello, Italy
Sergey Dashkovskiy, Germany
Adiel T. de Almeida-Filho, Brazil
Fabio De Angelis, Italy
Samuele De Bartolo, Italy
Abílio De Jesus, Portugal
Pietro De Lellis, Italy
Alessandro De Luca, Italy
Stefano de Miranda, Italy
Filippo de Monte, Italy
José António Fonseca de Oliveira Correia, Portugal
Jose Renato de Sousa, Brazil
Michael Defoort, France
Alessandro Della Corte, Italy
Laurent Dewasme, Belgium
Sanku Dey, India
Gianpaolo Di Bona, Italy

Angelo Di Egidio, Italy
Roberta Di Pace, Italy
Francesca Di Puccio, Italy
Ramón I. Diego, Spain
Yannis Dimakopoulos, Greece
Rossana Dimitri, Italy
Hasan Dinçer, Turkey
Alexandre B. Dolgui, France
José M. Domínguez, Spain
Georgios Dounias, Greece
Bo Du, China
Z. Du, China
George S. Dulikravich, USA
Emil Dumic, Croatia
Bogdan Dumitrescu, Romania
Madalina Dumitriu, United Kingdom
Saeed Eftekhar Azam, USA
Said El Kafhali, Morocco
Antonio Elipe, Spain
R. Emre Erkmen, Canada
John Escobar, Colombia
Francisco Periago Esparza, Spain
Gilberto Espinosa-Paredes, Mexico
Leandro F. F. Miguel, Brazil
Andrea L. Facci, Italy
Shahla Faisal, Pakistan
Giovanni Falsone, Italy
Hua Fan, China
Jianguang Fang, Australia
Nicholas Fantuzzi, Italy
Muhammad Shahid Farid, Pakistan
Hamed Faroqi, Iran
Mohammad Fattahi, Iran
Yann Favennec, France
Fiorenzo A. Fazzolari, United Kingdom
Giuseppe Fedele, Italy
Roberto Fedele, Italy
Zhongyang Fei, China
Baowei Feng, China
Mohammad Ferdows, Bangladesh
Arturo J. Fernández, Spain
Jesus M. Fernandez Oro, Spain
Massimiliano Ferraioli, Italy
Massimiliano Ferrara, Italy
Francesco Ferrise, Italy
Constantin Fetecau, Romania
Eric Feulvarch, France

Iztok Fister Jr., Slovenia
Thierry Floquet, France
Eric Florentin, France
Gerardo Flores, Mexico
Antonio Forcina, Italy
Alessandro Formisano, Italy
FRANCESCO FOTI, Italy
Francesco Franco, Italy
Elisa Francomano, Italy
Juan Frausto-Solis, Mexico
Shujun Fu, China
Juan C. G. Prada, Spain
Matteo Gaeta, Italy
Mauro Gaggero, Italy
Zoran Gajic, USA
Jaime Gallardo-Alvarado, Mexico
Mosè Gallo, Italy
Akemi Gálvez, Spain
Rita Gamberini, Italy
Maria L. Gandarias, Spain
Zhong-Ke Gao, China
Xingbao Gao, China
Yan Gao, China
Hao Gao, Hong Kong
Shangce Gao, Japan
Zhiwei Gao, United Kingdom
Giovanni Garcea, Italy
José García, Chile
Luis Rodolfo Garcia Carrillo, USA
Jose M. Garcia-Aznar, Spain
Harish Garg, India
Akhil Garg, China
Alessandro Gasparetto, Italy
Gianluca Gatti, Italy
Oleg V. Gendelman, Israel
Stylios Georgantzinis, Greece
Fotios Georgiades, India
Parviz Ghadimi, Iran
Ștefan Cristian Gherghina, Romania
Ganesh Ghorai, India
Georgios I. Giannopoulos, Greece
Agathoklis Giaralis, United Kingdom
Pablo Gil, Spain
Anna M. Gil-Lafuente, Spain
Ivan Giorgio, Italy
Gaetano Giunta, Luxembourg
Alessio Gizzi, Italy

Alireza Goli, Iran
Jefferson L.M.A. Gomes, United Kingdom
HECTOR GOMEZ, Chile
José Francisco Gómez Aguilar, Mexico
Emilio Gómez-Déniz, Spain
Antonio M. Gonçalves de Lima, Brazil
Qunxi Gong, China
Chris Goodrich, USA
Rama S. R. Gorla, USA
Veena Goswami, India
Xunjie Gou, Spain
Jakub Grabski, Poland
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
NALLAPPAN GUNASEKARAN, Japan
Dong-Sheng Guo, China
Hu Guo, China
Zhaoxia Guo, China
Jian-Ping Guo, China
Yusuf Gurefe, Turkey
Quang Phuc Ha, Australia
Li Haitao, China
Petr Hájek, Czech Republic
Mohamed Hamdy, Egypt
Muhammad Hamid, United Kingdom
Shigeyuki Hamori, Japan
Renke Han, United Kingdom
Weimin Han, USA
Zhen-Lai Han, China
Xingsi Han, China
Thomas Hanne, Switzerland
Xinan Hao, China
Mohammad A. Hariri-Ardebili, USA
Khalid Hattaf, Morocco
Defeng He, China
Xiao-Qiao He, China
Fu-Qiang He, China
Yanchao He, China
Yu-Ling He, China
salim HEDDAM, Algeria
Ramdane Hedjar, Saudi Arabia

Jude Hemanth, India
Reza Hemmati, Iran
Nicolae Herisanu, Romania
Alfredo G. Hernández-Díaz, Spain
M.I. Herreros, Spain
Eckhard Hitzer, Japan
Paul Honeine, France
Jaromir Horacek, Czech Republic
S. Hassan Hosseinnia, The Netherlands
Yingkun Hou, China
Xiaorong Hou, China
Lei Hou, China
Jie Hu, China
Yunfeng Hu, China
Yu-Chen Hu, Taiwan
Can Huang, China
Gordon Huang, Canada
Linsheng Huo, China
Sajid Hussain, Canada
ABID HUSSANAN, China
Asier Ibeas, Spain
Wubshet Ibrahim, Ethiopia
Orest V. Iftime, The Netherlands
Przemyslaw Ignaciuk, Poland
Muhammad Imran, Pakistan
Mukherjee IN, India
Giacomo Innocenti, Italy
Emilio Insfran Pelozo, Spain
Azeem Irshad, Pakistan
Alessio Ishizaka, France
Nazrul Islam, USA
Benoit Iung, France
Benjamin Ivorra, Spain
Breno Jacob, Brazil
Tushar Jain, India
Amin Jajarmi, Iran
Payman Jalali, Finland
Mahdi Jalili, Australia
Prashant Kumar Jamwal, Kazakhstan
Chiranjibe Jana, India
Łukasz Jankowski, Poland
Fahd Jarad, Turkey
Samuel N. Jator, USA
Juan C. Jauregui-Correa, Mexico
Kandasamy Jayakrishna, India
Reza Jazar, Australia
Khalide Jbilou, France

Isabel S. Jesus, Portugal
Chao Ji, China
Linni Jian, China
Bin Jiang, China
Qing-Chao Jiang, China., China
Peng-fei Jiao, China
Ricardo Fabricio Escobar Jiménez, Mexico
Emilio Jiménez Macías, Spain
Xiaoliang Jin, Canada
Zhuo Jin, Australia
Maolin Jin, Republic of Korea
Dylan F. Jones, United Kingdom
Ramash Kumar K, India
Viacheslav Kalashnikov, Mexico
Mathiyalagan Kalidass, India
BHABEN KALITA, USA
Tamas Kalmar-Nagy, Hungary
Rajesh Kaluri, India
Zhao Kang, China
Ramani Kannan, Malaysia
Tomasz Kapitaniak, Poland
Julius Kaplunov, United Kingdom
Konstantinos Karamanos, Belgium
J. Kavikumar, Malaysia
Michal Kawulok, Poland
Irfan Kaymaz, Turkey
Vahid Kayvanfar, Qatar
Krzysztof Kecik, Poland
Mohamed Khader, Egypt
Chaudry M. Khaliq, South Africa
Shahid Khan, Pakistan
Umar Khan, Pakistan
Mukhtaj Khan, Pakistan
Abdul Qadeer Khan, Pakistan
Mostafa M. A. Khater, Egypt
MOHAMMAD REZA KHEDMATI, Iran
Kwangki Kim, Republic of Korea
Nam-II Kim, Republic of Korea
Philipp V. Kiryukhantsev-Korneev, Russia
P.V.V Kishore, India
Jan Koci, Czech Republic
Ioannis Kostavelis, Greece
Sotiris B. Kotsiantis, Greece
Frederic Kratz, France
Vamsi Krishna, India
Kamalanand Krishnamurthy, India
Petr Krysl, USA

Edyta Kucharska, Poland
Krzysztof S. Kulpa, Poland
Prof. Ashwani Kumar, India
Kamal Kumar, India
Michal Kunicki, Poland
Cedrick A. K. Kwuimy, USA
Kyandoghere Kyamakya, Austria
Ivan Kyrchei, Ukraine
Davide La Torre, Italy
Márcio J. Lacerda, Brazil
Risto Lahdelma, Finland
Eduardo Lalla, The Netherlands
Giovanni Lancioni, Italy
Jaroslaw Latalski, Poland
Antonino Laudani, Italy
Hervé Laurent, France
Agostino Lauria, Italy
Aimé Lay-Ekuakille, Italy
Nicolas J. Leconte, France
Kun-Chou Lee, Taiwan
Dimitri Lefebvre, France
Eric Lefevre, France
Marek Lefik, Poland
Gang Lei, Saudi Arabia
Yaguo Lei, China
Kauko Leiviskä, Finland
Thibault Lemaire, France
Ervin Lenzi, Brazil
Roman Lewandowski, Poland
Jian Li, USA
Yushuai Li, Norway
Jun Li, China
Yueyang Li, China
Lianhui Li, China
Yuxing Li, China
ChenFeng Li, China
Zhen Li, China
Yang Li, China
Zhiyun Lin, China
Jian Lin, China
Yao-Jin Lin, China
Mingwei Lin, China
Qibin Lin, China
En-Qiang Lin, USA
Jianxu Liu, Thailand
Yuanchang Liu, United Kingdom
Bo Liu, China

Lei Liu, China
Heng Liu, China
Wanquan Liu, China
Bin Liu, China
Yu Liu, China
Sixin Liu, China
Bonifacio Llamazares, Spain
Alessandro Lo Schiavo, Italy
Jean Jacques Loiseau, France
Francesco Lolli, Italy
Paolo Lonetti, Italy
Sandro Longo, Italy
António M. Lopes, Portugal
Sebastian López, Spain
Pablo Lopez-Crespo, Spain
Cesar S. Lopez-Monsalvo, Mexico
Luis M. López-Ochoa, Spain
Ezequiel López-Rubio, Spain
Reza Lotfi, Iran
Vassilios C. Loukopoulos, Greece
Jose A. Lozano-Galant, Spain
Gabriele Maria Lozito, Italy
Songtao Lu, USA
Rongxing Lu, Canada
Zhiguo Luo, China
Gabriel Luque, Spain
Valentin Lychagin, Norway
Dazhong Ma, China
Xuanlong Ma, China
Junhai Ma, China
Junwei Ma, China
Praveen Kumar Reddy Maddikunta, India
Antonio Madeo, Italy
Alessandro Magnani, Belgium
Toqeer Mahmood, Pakistan
Fazal M. Mahomed, South Africa
Arunava Majumder, India
Sarfraz Nawaz Malik, Pakistan
Paolo Manfredi, Italy
Muazzam Maqsood, Pakistan
Adnan Maqsood, Pakistan
Giuseppe Carlo Marano, Italy
Damijan Markovic, France
Filipe J. Marques, Portugal
Luca Martinelli, Italy
Guiomar Martín-Herrán, Spain
Denizar Cruz Martins, Brazil

Francisco J. Martos, Spain
Elio Masciari, Italy
Franck Massa, France
Paolo Massioni, France
Alessandro Mauro, Italy
Jonathan Mayo-Maldonado, Mexico
Fabio Mazza, Italy
Pier Luigi Mazzeo, Italy
Laura Mazzola, Italy
Driss Mehdi, France
Dr. Zahid Mehmood, Pakistan
YUE MEI, China
Roderick Melnik, Canada
Debiao Meng, China
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
saleh mobayen, Taiwan, R.O.C., Iran
Nikunja Mohan Modak, India
Ardashir Mohammadzadeh, Iran
Sara Montagna, Italy
Roberto Montanini, Italy
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

Josefa Mula, Spain
Jose J. Muñoz, Spain
Giuseppe Muscolino, Italy
Dino Musmarra, Italy
Marco Mussetta, Italy
Ghulam Mustafa, Pakistan
Hariharan Muthusamy, India
Hakim Naceur, France
Alessandro Naddeo, Italy
Benedek Nagy, Turkey
Omar Naifar, Tunisia
Mariko Nakano-Miyatake, Mexico
Raj Nandkeolyar, India
Keivan Navaie, United Kingdom
Soumya Nayak, India
Adrian Neagu, USA
Erivelton Geraldo Nepomuceno, Brazil
Luís C. Neves, United Kingdom
AMA Neves, Portugal
Ha Quang Thinh Ngo, Vietnam
Dong Ngoduy, New Zealand
Nhon Nguyen-Thanh, Singapore
Majid Niazkar, Italy
Papakostas Nikolaos, Ireland
Jelena Nikolic, Serbia
Mehrbakhsh Nilashi, Malaysia
Tatsushi Nishi, Japan
Shanzhou Niu, China
Xesús Nogueira, Spain
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
Sergio Ortobelli, Italy
Naohisa Otsuka, Japan
Sid Ahmed Ould Ahmed Mahmoud, Saudi Arabia

Taoreed Owolabi, Nigeria
Cenap Özel, Turkey
Pawel Packo, Poland
Arturo Pagano, Italy
Madhumangal Pal, India
Roberto Palma, Spain
Alessandro Palmeri, United Kingdom
Pasquale Palumbo, Italy
Dragan Pamučar, Serbia
Li Pan, China
Weifeng Pan, China
K. M. Pandey, India
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
Petr Páta, Czech Republic
Pubudu N. Pathirana, Australia
Surajit Kumar Paul, India
Sitek Paweł, Poland
Luis Payá, Spain
Alexander Paz, Australia
Igor Pažanin, Croatia
Libor Pekař, Czech Republic
Francesco Pellicano, Italy
Marcello Pellicciari, Italy
Haipeng Peng, China
Bo Peng, China
Xindong Peng, China
Yuexing Peng, China
Mingshu Peng, China
Zhi-ke Peng, China
Zhengbiao Peng, Australia
Jian Peng, China
Xiang Peng, China
Marzio Pennisi, Italy
Maria Patrizia Pera, Italy
Matjaz Perc, Slovenia
A. M. Bastos Pereira, Portugal
Ricardo Perera, Spain
Wesley Peres, Brazil
F. Javier Pérez-Pinal, Mexico

Michele Perrella, Italy
Francesco Pesavento, Italy
Ivo Petras, Slovakia
Francesco Petrini, Italy
EUGENIA PETROPOULOU, Greece
Hoang Vu Phan, Republic of Korea
Lukasz Pieczonka, Poland
Dario Piga, Switzerland
Antonina Pirrotta, Italy
Marco Pizzarelli, Italy
Javier Plaza, Spain
Goutam Pohit, India
Kemal Polat, Turkey
Dragan Poljak, Croatia
Jorge Pomares, Spain
Hiram Ponce, Mexico
Sébastien Poncet, Canada
Volodymyr Ponomaryov, Mexico
Jean-Christophe Ponsart, France
Mauro Pontani, Italy
Cornelio Posadas-Castillo, Mexico
Francesc Pozo, Spain
Aditya Rio Prabowo, Indonesia
Anchasa Pramuanjaroenkij, Thailand
Christopher Pretty, New Zealand
Leonardo Primavera, Italy
B Rajanarayan Prusty, India
Luca Pugi, Italy
Krzysztof Puszynski, Poland
Goran D. Putnik, Portugal
Chuan Qin, China
Jianlong Qiu, China
Giuseppe Quaranta, Italy
Vitomir Racic, Italy
Ahmed G. Radwan, Egypt
Hamid Rahman, Pakistan
Carlo Rainieri, Italy
Kumbakonam Ramamani Rajagopal, USA
Venkatesan Rajinikanth, India
Ali Ramazani, USA
Higinio Ramos, Spain
Angel Manuel Ramos, Spain
Muhammad Afzal Rana, Pakistan
Amer Rasheed, 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
X. W. Ren, China
Carlo Renno, Italy
Fabrizio Renno, Italy
Shahram Rezapour, Iran
Ricardo Riaza, Spain
Francesco Riganti-Fulginei, Italy
Gerasimos Rigatos, Greece
Francesco Ripamonti, Italy
Marcelo Raúl Risk, Argentina
Jorge Rivera, Mexico
Eugenio Roanes-Lozano, Spain
Bruno G. M. Robert, France
Ana Maria A. C. Rocha, Portugal
Luigi Rodino, Italy
Francisco Rodríguez, Spain
Rosana Rodríguez López, Spain
Alessandra Romolo, Italy
Abdolreza Roshani, Italy
Francisco Rossomando, Argentina
Sudipta Roy, India
Jose de Jesus Rubio, Mexico
Weiguo Rui, China
Rubén Ruiz, Spain
Ivan D. Rukhlenko, Australia
Dr. Eswaramoorthi S., India
Chaman Lal Sabharwal, USA
Kishin Sadarangani, Spain
Andrés Sáez, Spain
Bekir Sahin, Turkey
Laxminarayan Sahoo, India
Michael Sakellariou, Greece
John S. Sakellariou, Greece
Salvatore Salamone, USA
Jose Vicente Salcedo, Spain
Alejandro Salcido, Mexico
Alejandro Salcido, Mexico
Salman saleem, Saudi Arabia
Ahmed Salem, Saudi Arabia
Nunzio Salerno, Italy
Rohit Salgotra, India
Miguel A. Salido, Spain

Zabidin Salleh, Malaysia
Roque J. Saltarén, Spain
Alessandro Salvini, Italy
Abdus Samad, India
Sovan Samanta, India
Nikolaos Samaras, Greece
Sylwester Samborski, Poland
pijush samui, India
Ramon Sancibrian, Spain
Giuseppe Sanfilippo, Italy
Omar-Jacobo Santos, Mexico
J Santos-Reyes, Mexico
José A. Sanz-Herrera, Spain
Evangelos J. Sapountzakis, Greece
Musavarah Sarwar, Pakistan
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
Joan Serra-Sagrista, Spain
Silvestar Šesnić, Croatia
Erhan Set, Turkey
Gerardo Severino, Italy
Ruben Sevilla, United Kingdom
Stefano Sfarra, Italy
Mohamed Shaat, United Arab Emirates
Mostafa S. Shadloo, France
Dr. Ismail Shah, Pakistan
Dr. Zahir Shah, Pakistan
Kamal Shah, Pakistan
Leonid Shaikhet, Israel
Vimal Shanmuganathan, India
Xingling Shao, China
Prayas Sharma, India
Xin Pu Shen, China
Bo Shen, Germany
hang shen, China
Hao Shen, China
Dimitri O. Shepelsky, Ukraine

Weichao SHI, United Kingdom
Jian Shi, China
Suzanne M. Shontz, USA
Babak Shotorban, USA
Zhan Shu, Canada
Angelo Sifaleras, Greece
Luca Silvestri, Italy
Nuno Simões, Portugal
Mehakpreet Singh, Ireland
Rajiv Singh, India
Harendra Singh, India
Thanin Sitthiwiratham, Thailand
Seralthan 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
Bosheng Song, China
Jussi Sopanen, Finland
Marco Spadini, Italy
Paolo Spagnolo, Italy
Bernardo Spagnolo, Italy
Ruben Specogna, Italy
Vasilios Spitas, Greece
Sri Sridharan, USA
Ivanka Stamova, USA
Rafał Stanisławski, Poland
Miladin Stefanović, Serbia
Florin Stoican, Romania
ALBERT ALEXANDER STONIER, India
Salvatore Strano, Italy
Yakov Strelniker, Israel
Kumarasamy Sudhakar, Malaysia
Zong-Yao Sun, China
Kangkang Sun, China
Qiuqin Sun, China
Shuaishuai Sun, Australia
Xiaodong Sun, China
Qiuye Sun, China
Suroso Suroso, Indonesia
Sergey A. Suslov, Australia

Nasser Hassen Sweilam, Egypt
Andrzej Swierniak, Poland
M Syed Ali, India
Andras Szekrenyes, Hungary
Kumar K. Tamma, USA
Yong (Aaron) Tan, United Kingdom
Marco Antonio Taneco-Hernández, Mexico
Hafez Tari, USA
Alessandro Tasora, Italy
Sergio Teggi, Italy
Adriana del Carmen Téllez-Anguiano, Mexico
Ana C. Teodoro, Portugal
Efsthathios E. Theotokoglou, Greece
Jing-Feng Tian, China
Alexander Timokha, Norway
Stefania Tomasiello, Italy
Gisella Tomasini, Italy
Isabella Torcicollo, Italy
Francesco Tornabene, Italy
Javier Martinez Torres, Spain
Mariano Torrisi, Italy
Thang nguyen Trung, Vietnam
Sang-Bing Tsai, China
George Tsiatas, Greece
Antonios Tsourdos, United Kingdom
Le Anh Tuan, Vietnam
Federica Tubino, Italy
Nerio Tullini, Italy
Emilio Turco, Italy
Ilhan Tuzcu, USA
Efstratios Tzirtzilakis, Greece
Filippo Ubertini, Italy
Marjan Uddin, Pakistan
Mohammad Uddin, Australia
Serdar Ulubeyli, Turkey
Mati ur Rahman, Pakistan
FRANCISCO UREÑA, Spain
Panayiotis Vafeas, Greece
Giuseppe Vairo, Italy
Jesus Valdez-Resendiz, Mexico
Eusebio Valero, Spain
Stefano Valvano, Italy
Marcello Vasta, Italy
Carlos-Renato Vázquez, Mexico
Miguel E. Vázquez-Méndez, Spain
Martin Velasco Villa, Mexico

Kalyana C. Veluvolu, Republic of Korea
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 Vitek, Czech Republic
Jan Vorel, Czech Republic
Michael Vynnycky, Sweden
Mohammad W. Alomari, Jordan
Fu-Kwun Wang, Taiwan
C. H. Wang, Taiwan
Yung-Chung Wang, Taiwan
Hao Wang, USA
Zhenbo Wang, USA
Zenghui Wang, South Africa
Yongqi Wang, Germany
Dagang Wang, China
Weiwei Wang, China
Yong Wang, China
Ji Wang, China
Bingchang Wang, China
Lei Wang, China
Qingling Wang, China
Xinyu Wang, China
Hui Wang, China
qiang wang, China
Kang-Jia Wang, China
Huaiyu Wang, China
J.G. Wang, China
Zhibo Wang, China
Shuo Wang, China
Guoqiang Wang, China
Roman Wan-Wendner, Austria
Fangqing Wen, China
P.H. Wen, United Kingdom
Waldemar T. Wójcik, Poland
Wai Lok Woo, United Kingdom
QiuHong Wu, China
Xianyi Wu, China
Chi Wu, Australia
Zhibin Wu, China
Yuqiang Wu, China
Changzhi Wu, China
Zhizheng Wu, China
Michalis Xenos, Greece
hao xiao, China
Xiao Ping Xie, China
Xue-Jun Xie, China
Hang Xu, China
Lei Xu, China
Qingzheng Xu, China
Zeshui Xu, China
Lingwei Xu, China
Qilong Xue, China
Yi Xue, China
Binghan Xue, China
Joseph J. Yame, France
Zhiguo Yan, China
Chuanliang Yan, China
Xinggang Yan, United Kingdom
Ray-Yeng Yang, Taiwan
Mijia Yang, USA
Jixiang Yang, China
Bo Yang, China
Hongtai Yang, China
Zaoli Yang, China
Weilin Yang, China
Zhihong Yao, China
Min Ye, China
Jun Ye, China
Luis J. Yebra, Spain
Peng-Yeng Yin, Taiwan
Muhammad Haroon Yousaf, Pakistan
Yuan Yuan, United Kingdom
Qin Yuming, China
Abdullahi Yusuf, Nigeria
Akbar Zada, Pakistan
Elena Zaitseva, Slovakia
Arkadiusz Zak, Poland
Muhammad Zakarya, Pakistan
Daniel Zaldivar, Mexico
Ernesto Zambrano-Serrano, Mexico
Francesco Zammori, Italy
Rafal Zdunek, Poland
Ahmad Zeeshan, Pakistan
Ibrahim Zeid, USA
Nianyin Zeng, China
Bo Zeng, China
Junyong Zhai, China
Wenyu Zhang, China







Lingfan Zhang, China
Xiaofei Zhang, China
Tongqian Zhang, China
Yong Zhang, China
Qian Zhang, China
Kai Zhang, China
Hao Zhang, China
Xuping Zhang, Denmark
Tianwei Zhang, China
Jian Zhang, China
Yinyan Zhang, China
Xianming Zhang, Australia
Haopeng Zhang, USA
Mingjie Zhang, Norway
Yifan Zhao, United Kingdom
Yongmin Zhong, Australia
Zebo Zhou, China
Zhe Zhou, China
Jian G. Zhou, United Kingdom
Debao Zhou, USA
Quanxin Zhu, China
Wu-Le Zhu, China
Gaetano Zizzo, Italy
Zhixiang Zou, China
Mingcheng Zuo, China

Contents




Nature-Inspired Intelligence Methods and Applications

Qingzheng Xu , Ana Maria A. C. Rocha , Erik Cuevas , and Iztok Fister Jr. 
Editorial (2 pages), Article ID 9825231, Volume 2022 (2022)




Multirobot Task Allocation in e-Commerce Robotic Mobile Fulfillment Systems

Ruiping Yuan , Juntao Li , Xiaolin Wang , and Liyan He 
Research Article (10 pages), Article ID 6308950, Volume 2021 (2021)



A Novel Crow Search Algorithm Based on Improved Flower Pollination

Qian Cheng , Huajuan Huang , and Minbo Chen 
Research Article (26 pages), Article ID 1048879, Volume 2021 (2021)

On-Demanding Information Acquisition in Multi-UAV-Assisted Sensor Network: A Satisfaction-Driven Perspective

Hua Yang , Jungang Yang , Wendong Zhao , and Cuntao Liu 
Research Article (14 pages), Article ID 2717733, Volume 2021 (2021)


Remote Sensing-Based Urban Green Space Detection Using Marine Predators Algorithm Optimized Machine Learning Approach

Nhat-Duc Hoang  and Xuan-Linh Tran 
Research Article (22 pages), Article ID 5586913, Volume 2021 (2021)

Virtual Screening of Drug Proteins Based on Imbalance Data Mining

Peng Li , Lili Yin, Bo Zhao, and Yuezhongyi Sun
Research Article (10 pages), Article ID 5585990, Volume 2021 (2021)

Multiobjective Brain Storm Optimization Community Detection Method Based on Novelty Search

Xiaoying Pan, Jia Wang , Miao Wei, and Hongye Li
Research Article (14 pages), Article ID 5535881, Volume 2021 (2021)

Editorial

Nature-Inspired Intelligence Methods and Applications

Qingzheng Xu ¹, **Ana Maria A. C. Rocha** ², **Erik Cuevas** ³ and **Iztok Fister Jr.** ⁴

¹National University of Defense Technology, Wuhan, China

²ALGORITMI Center, University of Minho, Braga, Portugal

³Universidad de Guadalajara, Guadalajara, Mexico

⁴University of Maribor, Maribor, Slovenia

Correspondence should be addressed to Qingzheng Xu; xuqingzheng@hotmail.com

Received 2 August 2022; Accepted 2 August 2022; Published 1 September 2022

Copyright © 2022 Qingzheng Xu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Research in nature-inspired intelligence methods and applications has increased exponentially in the past decade. Inspired by a natural phenomenon from biology, physics, or sociology, population-based nature-inspired algorithms aim to achieve satisfactory results for many difficult optimization problems effectively. Compared with deterministic optimization methods, they have many advantages, such as scalability, adaptability, collective robustness, and individual simplicity. However, they still face many challenges that require further research.

The target of this special issue was to provide a comprehensive and latest collection of research works on various aspects of population-based nature-inspired algorithms, as well as its potential application in various sciences and engineering domains. This special issue received 15 submissions in total. The authors were from 26 affiliations in 7 countries. Each submitted article was subject to assessment by at least two independent reviewers. After a fair and rigorous peer-review process, 6 of them are published in the special issue, with an acceptance rate of 40.0%.

The research paper submitted by Cheng et al., entitled “A novel crow search algorithm based on improved flower pollination”, proposed a crow search algorithm based on an improved flower pollination algorithm (IFCSA) to prevent stagnation and convergence to local minima. In order to balance the global search and local search capabilities, an inverse incomplete gamma function was first introduced to make the awareness probability decrease nonlinearly. In addition, a cross-pollination strategy with Cauchy mutation was also introduced, to avoid the blindness of individual location update. Experimental results on benchmark

problems show that this algorithm has better performance than the original crow search algorithm.

In the paper “Multiobjective brain storm optimization community detection method based on novelty search” by Pan et al., a multiobjective brain storm optimization based on novelty search (MOBSO-NS) was proposed to solve the premature convergence caused by the loss of diversity in complex network community detection. A novelty multi-population parallel search mechanism based on elite, ordinary, and novelty clusters was used effectively to avoid premature convergence. Secondly, a restarting strategy was introduced to help individuals escape from the local optimal point. Experimental results show that the algorithm can find the Pareto optimal network community structure set and excavate the network community with higher quality.

The paper entitled “Virtual screening of drug proteins based on imbalance data mining” by Li et al. dealt with the imbalanced data problem in traditional molecular docking-based virtual screening methods. As a preprocessing method, the GA-SMOTE algorithm can balance the imbalanced dataset by increasing the number of minority class samples. Furthermore, the idea of integrated learning was introduced to improve the prediction accuracy. More specifically, in this paper, the random forest was combined with the support vector machine (SVM). The effectiveness of the proposed technique was verified by experiments on three important datasets.

The task of determining the hyperparameters of a machine learning model can be modelled as an optimization problem. In the paper “Remote sensing-based urban green space detection using marine predators algorithm optimized machine learning approach” by Hoang et al., a marine

predators algorithm (MPA) was employed to optimize the SVM training phase by identifying an appropriate set of the SVM's hyperparameters. The hybrid method was developed and verified for urban green space detection.

The paper by Yuan et al. entitled "Multirobot task allocation in e-commerce robotic mobile fulfilment systems" studied the multirobot task allocation (MRTA) in an e-commerce robotic mobile fulfilment system. First, considering both the picking time balance and the load balance, a multirobot task allocation model was established to minimize the overall picking time. Then, a four-stage balanced heuristic auction algorithm was designed to solve the model efficiently.

In the paper "On-demanding information acquisition in a multi-UAV-assisted sensor network: A satisfaction-driven perspective" by Yang et al., a multi-UAV task assignment problem was developed as a complex optimization problem. The objective function is defined as both maximizing the priority-weighted satisfaction of users and minimizing the total energy consumption of UAVs. Then, a multi-population-based cooperation genetic algorithm (MPCGA) was proposed, and the numerical results demonstrated its effectiveness.

We are convinced that this special issue has led to important and inspiring contributions to the research of nature-inspired intelligent methods and applications.

Conflicts of Interest

The guest editors declare that there are no conflicts of interest regarding the publication of this special issue.

Qingzheng Xu

Ana Maria A. C. Rocha

Erik Cuevas





Iztok Fister Jr.

Acknowledgments

The guest editorial team would like to express gratitude to all the authors for their excellent contributions. The editors also wish to thank the anonymous reviewers for their many insightful comments and suggestions, which were very useful in improving the quality of the submitted papers.

Research Article

Multirobot Task Allocation in e-Commerce Robotic Mobile Fulfillment Systems

Ruiping Yuan ^{1,2}, Juntao Li ^{1,2}, Xiaolin Wang ¹, and Liyan He ¹

¹School of Information, Beijing Wuzi University, Beijing 101149, China

²Beijing Key Laboratory of Intelligent Logistics Systems, Beijing 101149, China

Correspondence should be addressed to Juntao Li; ljtletter@126.com

Received 13 August 2021; Revised 15 September 2021; Accepted 29 September 2021; Published 29 October 2021

Academic Editor: Qingzheng Xu

Copyright © 2021 Ruiping Yuan et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Robotic Mobile Fulfillment System (RMFS) is a new type of parts-to-picker order picking system and has become the development trend of e-commerce logistics distribution centers. There are usually a large number of tasks need to be allocated to many robots and the picking time for e-commerce orders is usually very tight, which puts forward higher requirements for the efficiency of multirobot task allocation (MRTA) in e-commerce RMFS. Current researches on MRTA in RMFS seldom consider task correlation and the balance among picking stations. In this paper, a task time cost model considering task correlation is built according to the characteristics of the picking process. Then, a multirobot task allocation model minimizing the overall picking time is established considering both the picking time balance of picking stations and the load balance of robots. Finally, a four-stage balanced heuristic auction algorithm is designed to solve the task allocation model and the tasks with execution sequence for each robot are obtained. By comparing with the traditional task time cost model and the algorithm without considering the balance among picking stations, it is found that the proposed model and algorithm can significantly shorten the overall picking time.

1. Introduction

The e-commerce logistics distribution center is an important part of the e-commerce supply chain, which greatly affects the operation efficiency of e-commerce. There are usually a large number of stock-keeping units (SKUs) in the e-commerce logistics distribution center, and e-commerce orders have the characteristics of small batch, high-frequency, strong randomness, and tight distribution time, which puts forward higher requirements for order picking efficiency [1]. The traditional manual picking mode is unable to meet the demand because of its high error rate and low picking efficiency. In recent years, a new picking system Robotic Mobile Fulfillment System (RMFS) has become the development trend of e-commerce logistics picking system because of its high efficiency, intelligence, and flexibility [2]. The deployment of the KIVA logistics robot by Amazon in 2012 triggered the application market of RMFS [3]. In RMFS, shelves are carried by robots to picking stations, where pickers pick goods (parts) from shelves, so it is a kind

of parts-to-picker picking system. Because of the significant difference between the new picking mode and the traditional manual picking mode, many decision-making problems in this new picking mode need to be studied in depth, such as storage assignment [4], order batching [5], multirobot task allocation [6], and path planning [7]. This paper mainly studies how to assign a batch of picking tasks to multiple robots, which belongs to multirobot task allocation (MRTA) problem.

Multirobot task allocation (MRTA) refers to the assigning of a series of tasks to multiple robots with certain constraints to achieve an objective, such as minimizing the total travel distance of all robots or the average cost of each task and so on [8]. The main methods for solving MRTA problems include combinatorial optimization, market-based approach, swarm intelligence approach, behavior-based approach, and emotional recruitment approach [9]. Among them, methods based on the market mechanism, such as the auction method, have attracted wide attention because of their high efficiency, good robustness, and easy expansion.

In the market mechanism, robots negotiate with each other through bidding and ultimately complete the task allocation [10]. In a dynamic task allocation environment, using an auction algorithm will contribute to the fairness and real-time of task allocation and reduce the design complexity of the system [11].

Zlot et al. [12] applied a market mechanism algorithm to multirobot dynamic task allocation for the first time, which can solve small-scale dynamic task allocation problems. Elango et al. [13] used the K-means clustering algorithm and auction mechanism to solve the dual-objective task allocation model, which considers both total travel distance and robot efficiency. Lozenguez et al. [14] proposed a sequential synchronous auction protocol to coordinate the task allocation of robots. Heap and Pagnucco [15] designed a repetitive sequential single clustering auction algorithm for multirobot dynamic task assignment. For task allocation in an intelligent warehousing system, Zhou et al. [16] proposed a balanced heuristic auction algorithm balancing a load of robots to improve the efficiency of task allocation in an intelligent warehousing system. An important part of the auction algorithm is the computation of task cost, which can be measured by completion time or path length. Liu and Kroll [17] regarded the time a robot takes to fulfill a task as the task cost for MRTA. Dou et al. [18] took path length as task cost using reinforcement learning. Lamballais et al. [19] established a queueing model to measure the utility of logistics robots and workers.

Although there are many literature on MRTA, few studies are about the MRTA problems in e-commerce RMFS. In an e-commerce order picking environment, there are a large number of picking tasks and mobile robots. So, the task allocation problem is a complex NP-hard optimization problem. Furthermore, the picking time for e-commerce orders is usually very tight, which requires higher efficiency of task allocation. Therefore, it is necessary to propose an efficient task allocation method based on the application characteristics. Former research studies on MRTA mainly consider task completion time or the total travel length of robots but seldom consider the workload balance among picking stations and robots. Because of the parallel operation mode of multiple picking stations, the picking time of the whole system is determined by the picking station with the longest picking time. In the process of task allocation, the balance among picking stations should be fully considered. Uneven idleness among picking stations, e.g., robots wait in line at some picking stations while other picking stations are idle, certainly will reduce the efficiency of the system. Therefore, balancing the picking time among picking stations plays an important role in improving the picking efficiency. In addition, the correlation among tasks was not fully taken into account in the previous task cost model. For example, if two tasks of one picking station are on the same shelf, by assigning the two tasks to the same robot and arranging their execution sequence adjacent to each other, the two tasks can be completed through one shelf visit and the cost for completing the tasks can be greatly reduced.

The innovations of this paper are as follows: (1) based on the real operation mode that a robot can serve multiple picking stations at one time, the correlation between tasks is refined and a new task time cost model is proposed according to different types of task correlation. (2) Because of the parallel operation mode of multiple picking stations, the picking time of the whole system is determined by the picking station with the longest picking time. So the balance among picking stations as well as the load balance of robots is considered to improve picking efficiency. (3) A four-stage balanced heuristic auction algorithm is designed to solve the task allocation model, which achieves the goal of balancing picking time among picking stations by controlling the sequence of task assignments.

The remainder of this paper is organized as follows. In Section 2, the task assignment problem of logistics robots in e-commerce RMFS is described in detail, and the parameters for model formulation are given. In Section 3, a new task cost calculation method considering the correlation between tasks is described, and the multiple logistics robot task allocation model considering the balance of picking stations is established. In Section 4, a four-stage balanced heuristic auction algorithm is designed to solve the task allocation model. In Section 5, simulation experiments are conducted, and the results are analyzed to verify the proposed model and algorithm. Section 6 concludes the paper and presents the limitations and prospects of the research.

2. Problem Description

2.1. The Operation Process of e-Commerce RMFS. E-commerce orders have the characteristics of many varieties, small batch, and high frequency. In order to improve order picking efficiency, multiple orders arrived in a certain period of time are usually combined into one batch for picking, which is called wave-picking [20]. That is, the continuously arriving orders are placed in an order pool, and then, a certain number of orders are selected from the order pool as a wave of orders for picking. After that, orders are allocated to picking stations, and the items to be picked in the orders of each picking station are merged to generate a picking list, which consists of many picking tasks. Each item in the picking lists corresponds to a picking task. Finally, these picking tasks are assigned to robots according to some rules, and these robots cooperate to complete these picking tasks.

An e-commerce RMFS is depicted in Figure 1, which is similar to the typical KIVA Systems [21, 22]. The picking system consists of movable shelves, picking stations, mobile robots, conveyor belts, etc. The storage area is composed of neatly distributed movable shelves. Different kinds of goods can be placed on the same shelf. Each of the picking stations on the left side is equipped with a picker to pick items from shelves and a buffer area, where shelves carried by logistics robots can queue and wait for picking. Next to the picking stations is a conveyor belt, which is used to transport the picked items.

The process of goods picking is as follows: a robot runs from the current location to the shelf where the required goods are located (Figure 1 ①); then the logistics robot

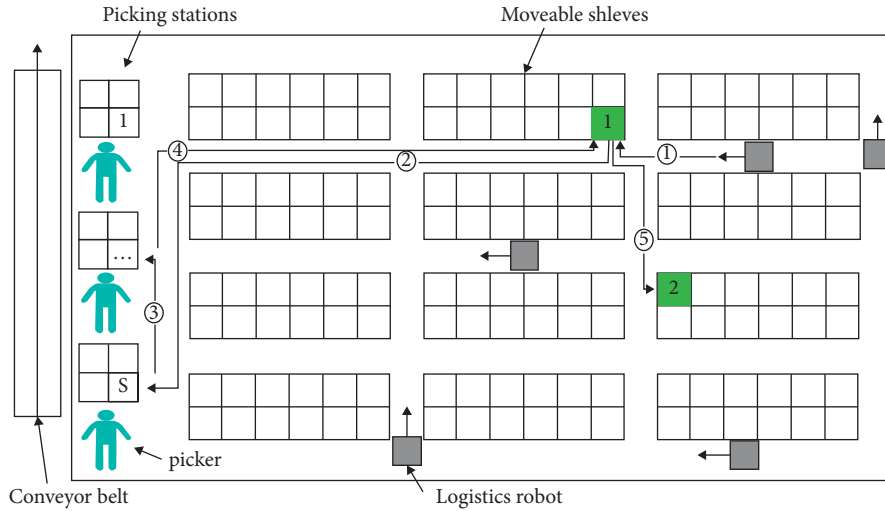


FIGURE 1: The typical layout of an e-commerce RMFS.

carries the shelf to the corresponding picking station (Figure 1 ②). After the picker picks the required goods from the shelf, the logistics robot either carries the shelf to the next picking station (Figure 1 ③) or returns the shelf back to its original position in the storage area (Figure 1 ④). Then, the robot goes on to the next task (Figure 1 ⑤). All robots work together to complete the picking tasks. When all the orders assigned to the picking stations are fulfilled, this wave of picking is finished.

2.2. Parameter Definition of MRTA in RMFS. Through the operation process analysis, we find that a robot can perform multiple tasks at one time (may be multiple goods located on the same shelf), and one task can only be performed by one robot. The MRTA problem in RMFS can be classified as multitask robots and single-robot tasks (MT-SR) problem [23].

Since a robot can serve one or more picking stations and perform one or more tasks at one shelf visit, the time cost of each task should be calculated in different ways. In this paper, the time cost of robots to perform different tasks is distinguished through refining the task allocation process, especially considering the situation that robots can transport a shelf to serve multiple picking stations at one time. Due to the parallel operating mode of multiple picking stations, the picking station with the longest picking time will determine the total picking time of the orders. So, the picking time among picking stations is balanced by controlling the task allocation sequence in this paper.

Before the task allocation model formulation, the following assumptions are needed:

- (1) The picking time for each item is the same, which is a constant.
- (2) Several different goods can be placed on the same shelf and one kind of goods can only be placed on one shelf. Therefore, the location of each goods is known.

- (3) Logistics robots are isomorphic and travel at the same speed, without considering the interaction of logistics robots.

Due to the road layout of the RMFS (see Figure 1) and the kinematic constraints of robots, Manhattan distance is adopted and the grid map method is used to model the environment. The parameters and variables used in the model formulation are defined as follows:

$A = \{a_1, a_2, \dots, a_m\}$ is a set of picking tasks, and m is the total number of the picking tasks of this wave.

$R = \{r_1, r_2, \dots, r_n\}$ is a set of mobile robots, and n is the total number of the robots.

$S = \{s_1, s_2, \dots, s_h\}$ is a set of picking stations, and h is the total number of picking stations.

$O = \{O_1, O_2, \dots, O_n\}$ is a collection of the task assignment schemes for all robots, where $O_j = \{a_{j1} \rightarrow a_{j2} \rightarrow \dots \rightarrow a_{j|O_j|}\}$ represents the task assignment scheme for the robot r_j , which is an orderly set of tasks assigned to r_j . a_{jl} is the task performed by r_j in order l , and $|O_j|$ represents the total number of tasks assigned to r_j .

s_{a_i} represents the picking station where task a_i was assigned, which is already known before allocating tasks to robots.

\bar{d}_{a_i} represents the Manhattan distance between the shelf of task a_i and its picking station.

$d_{a_i a_{i'}}$ represents the Manhattan distance between the shelves of task a_i and task $a_{i'}$.

$d_{s_k s_{k'}}$ represents the Manhattan distance between picking stations s_k and $s_{k'}$.

$d_{r_j a_i}$ represents the Manhattan distance between the initial location of the robot r_j and the shelf of the task a_i .

v' is a constant, which represents the moving speed of logistics robots.

t' is a constant, which represents the picking time for one item.

l'_{jk} is the order of the last task of picking station s_k fulfilled by robot r_j .

$x_{ijl} = \begin{cases} 1, & \text{if } a_{jl} = a_i, \\ 0, & \text{else,} \end{cases}$, which means that only if task a_i

is the l th task allocated to robot r_j , x_{ijl} equals 1.

t_{ijl} is the time cost, which means the time used of fulfilling the task a_i by robot r_j in the l th order.

3. Model Formulation

3.1. The Time Cost of Tasks considering Task Correlation. From the previous analysis, it is known that when tasks are fulfilled by different robots in a different order, the time costs of the tasks are different. If task a_i is fulfilled by robot r_j and the order of execution is l , that is $a_{jl} = a_i$, the time cost of task a_i is represented by t_{ijl} . When $l = 1$, a_i is the first task fulfilled by the robot r_j and has no preceding task. When $l \geq 2$, t_{ijl} is relevant with the preceding task $a_{j(l-1)}$. For the convenience of representation, let $\alpha = a_{j(l-1)}$. The time cost t_{ijl} of completing task a_i is expressed as follows:

$$t_{ijl} = \begin{cases} \frac{d_{r_j a_i} + 2\bar{d}_{a_i}}{v'} + t', & l = 1, \\ \frac{d_{\alpha a_i} + 2\bar{d}_{a_i}}{v'} + t', & l \geq 2 \text{ and } B_{\alpha a_i} = 0, \\ \frac{d_{s_\alpha s_{a_i}} + \bar{d}_{a_i} - \bar{d}_\alpha}{v'} + t', & l \geq 2 \text{ and } B_{\alpha a_i} = 1. \end{cases} \quad (1)$$

$B_{\alpha a_i}$ is a 0-1 variable. Its value is 1 when task a_i is fulfilled by robot r_j and located on the same shelf as its preceding task α . Otherwise, its value is 0. The time cost of completing one task includes the travel time of the logistics robot and the picking time of the picker. The picking time is a constant, and the travel time of the logistics robot is related to the travel distance. When $l = 1$, a_i is the first task of robot r_j , and the travel distance of r_j includes the distance r_j moving from its initial location to the shelf where task a_i is located ($d_{r_j a_i}$), carrying the shelf to the picking station and returning the shelf to its original location ($2\bar{d}_{a_i}$). So, the total distance that r_j travels is $d_{r_j a_i} + 2\bar{d}_{a_i}$. When $l \geq 2$ and $B_{\alpha a_i} = 0$, task a_i and its preceding task α are not located on the same shelf, and the travel distance of r_j consists of three parts: the distance robot r_j traveling from the shelf of the preceding task to the shelf of task a_i ($d_{\alpha a_i}$, because r_j should send back the preceding task α to its original location, so when r_j begins to perform task a_i , it starts from there), and the distance r_j carrying the shelf to the picking station and returning the shelf to its original location ($2\bar{d}_{a_i}$). So, the total distance r_j that travels is $d_{\alpha a_i} + 2\bar{d}_{a_i}$. When $l \geq 2$ and $B_{\alpha a_i} = 1$, task a_i and its preceding task α are located on the same shelf. Robot r_j does not need to send the shelf of the preceding task to its original location but to send the shelf to the picking station s_{a_i} from the picking station s_α , and the travel distance is $d_{s_\alpha s_{a_i}}$.

After picking a_i , r_j needs to send the shelf to its original location and the travel distance is \bar{d}_{a_i} . So, the total distance r_j that travels is $d_{s_\alpha s_{a_i}} + \bar{d}_{a_i} - \bar{d}_\alpha$.

It can be seen from equation (1) that when $B_{\alpha a_i} = 1$, the time cost of the task a_i will be greatly reduced. In this case, task a_i and task α are called correlated tasks. Assuming that $\alpha = a_{j(l-1)}$ and $\beta = a_{jl}$ are two tasks fulfilled by the robot r_j in adjacent sequence, the correlation between these two tasks can be divided into the following categories:

- (1) Strongly correlated tasks: when $B_{\alpha\beta} = 1$ and $s_\alpha = s_\beta$, it means that task α and task β are located on the same shelf and belong to the same picking station. In this case, the two tasks are called strongly correlated tasks. According to equation (1), the time cost of task β only consists of picking time t' , which is the minimum time cost of task β .
- (2) Weakly correlated tasks: when $B_{\alpha\beta} = 1$ and $s_\alpha \neq s_\beta$, it means that task α and task β are located on the same shelf but belong to different picking stations. In this case, the two tasks are called weakly correlated tasks. According to equation (1), the time cost of task β is only the travel time between two picking stations and picking time, which is also greatly reduced.
- (3) Uncorrelated tasks: when $B_{\alpha\beta} = 0$, it means that task α and task β are located on different shelves. In this case, the two tasks are called uncorrelated tasks.

As above, when the two tasks fulfilled sequentially by the same robot are correlated, the time cost will be greatly reduced. Therefore, when establishing the MRTA model, the correlation between tasks should be fully considered.

3.2. The Time Cost of Robots and Picking Stations. The total time cost T_j of robot r_j taking to fulfill assigned tasks is expressed as equation (2), which means the total time used by robot r_j to fulfill all the tasks assigned to it.

$$T_j = \sum_{i=1}^m \sum_{l=1}^{|o_j|} (x_{ijl} \times t_{ijl}). \quad (2)$$

The total picking time T'_k of picking station s_k is shown in the following equation:

$$T'_k = \max_j \left(\sum_{i=1}^m \sum_{l=1}^{l'_{jk}} t_{ijl} \times x_{ijl} \right). \quad (3)$$

In equation (3), l'_{jk} represents the last task of picking station s_k assigned to robot r_j . T'_k represents the longest time the robots take to complete the tasks of picking station s_k . Because the tasks of a picking station are fulfilled by the cooperation of multiple robots, the total picking time of a picking station is determined by the logistics robot that uses the longest time to complete the tasks of the picking station.

3.3. Multirobot Task Allocation Model. Considering the balance of picking time among picking stations, the first objective function f_1 is to find the picking station with the

shortest picking time, which is the picking station the next task to be assigned.

$$f_1 = \min_k T'_k. \quad (4)$$

After determining the tasks to be assigned next, which robots these tasks should be assigned to are further determined. The objective of task allocation is to minimize the total picking time of all tasks in this wave. Because of the parallel operation mode of logistics robots, the total picking time of this wave depends on the robot with the longest picking time. An integer linear programming model is established to minimize the picking time of the logistics robot with the longest picking time.

$$f_2 = \min \left(\max_j T_j \right), \quad (5)$$

s.t.

$$\sum_{i=1}^n \sum_{l=1}^{|o_i|} x_{ijl} = 1, \quad (6)$$

$$\sum_{j=1}^n |o_j| = m. \quad (7)$$

Equation (6) means that a task can only be fulfilled by one robot, and the execution sequence is unique. Equation (7) means that the sum of tasks assigned to all robots is equal to the total number of tasks.

4. Algorithm Design

According to the characteristics of MRTA problem in e-commerce RMFS, this paper designs a four-stage balanced heuristic algorithm using parallel single-task auction [24] to solve the task allocation model. Parallel single-task auction is faster and more robust than other auction algorithms [25]; so, it is more suitable for task allocation of large-scale order picking robots. The algorithm is divided into four stages: (1) set the initial task allocation rules to decide the first task for each robot; (2) set the next task allocation rules using the correlation between tasks and considering the situation when one shelf can serve multiple picking stations at one time; (3) use the sequential auction algorithm to determine the picking station where the next to-be-assigned task is located, which reduces the range of the next task and balances the picking time of picking stations; then, the parallel single-task auction algorithm is used to determine the next task to be assigned and the robot to complete the task; (4) perform a dynamic adjustment on the calculation results to handle congestion and delay of robots during the process of fulfilling tasks. The specific steps of the algorithm are explained in detail in Sections 4.1–4.4 and the flowchart is shown in Figure 2.

4.1. Step 1: Initial Task Allocation for Each Robot.

- (1) Define a set Unallocated_{s_k} for each picking station s_k , which is used to store tasks that have not been

assigned on picking station s_k and each set is updated once after a task of the set has been assigned.

- (2) Define a set O_j for each robot r_j to store tasks that have been assigned to logistics robot r_j and their specific execution sequence. The initial state of the set is empty.
- (3) For robot r_j , traverse tasks are not assigned on all the picking stations to find the task with the shortest distance from the initial location of robot r_j and put the task into set O_j as the first task a_{j1} of robot r_j . At the same time, update the set Unallocated_{s_k} .
- (4) When there is one and only one task in each set O_j , the initial task allocation of all robots is completed. Then, proceed to Step 2.

4.2. Step 2: Task Allocation of the l th ($l \geq 2$) Task of Robots considering Task Correlation.

- (1) For robot r_j , denote the last task in O_j at present as α_j .
- (2) Find all the strongly correlated tasks with α_j and add them into set O_j . Traverse the tasks in the set $\text{Unallocated}_{s_{\alpha_j}}$. If there are strongly correlated tasks with α_j , add them in turn to set O_j , and update $\text{Unallocated}_{s_{\alpha_j}}$; then, turn to (3); otherwise, turn to (3) directly.
- (3) Find all the weakly correlated tasks with α_j and add them into set O_j . Traverse all the to-be-assigned tasks in other stations except s_{α_j} . If there are weakly correlated tasks with α_j , add them to set O_j in proper order according to the distance between s_{α_j} and the picking stations where the tasks are located, and update the unallocated task set of the related picking stations.
- (4) If there are still tasks unallocated in this round, go to Step 3; otherwise, go to Step 4.

4.3. Step 3: Task Allocation Based on Sequential and Parallel Single-Task Auction Algorithm.

- (1) Use a sequential auction algorithm to decide the task on which picking station is the next task to be assigned. According to equation (3), the picking station with the shortest picking time at present is found, denoted as s' , from which the next task to be assigned is selected. The purpose is to balance the picking time among the picking stations, avoid uneven workload, and also reduce the selection range of the next to-be-assigned task. Then, turn to (2).
- (2) For the set $\text{Unallocated}_{s'}$ found in (1), the parallel auction algorithm is used to calculate the next to-be-assigned task and the robot fulfilling the task.
 - (a) Each of the robots calculates the time cost of fulfilling each task in set $\text{Unallocated}_{s'}$ according to equation (1), based on its tasks situation and the position of its last task.

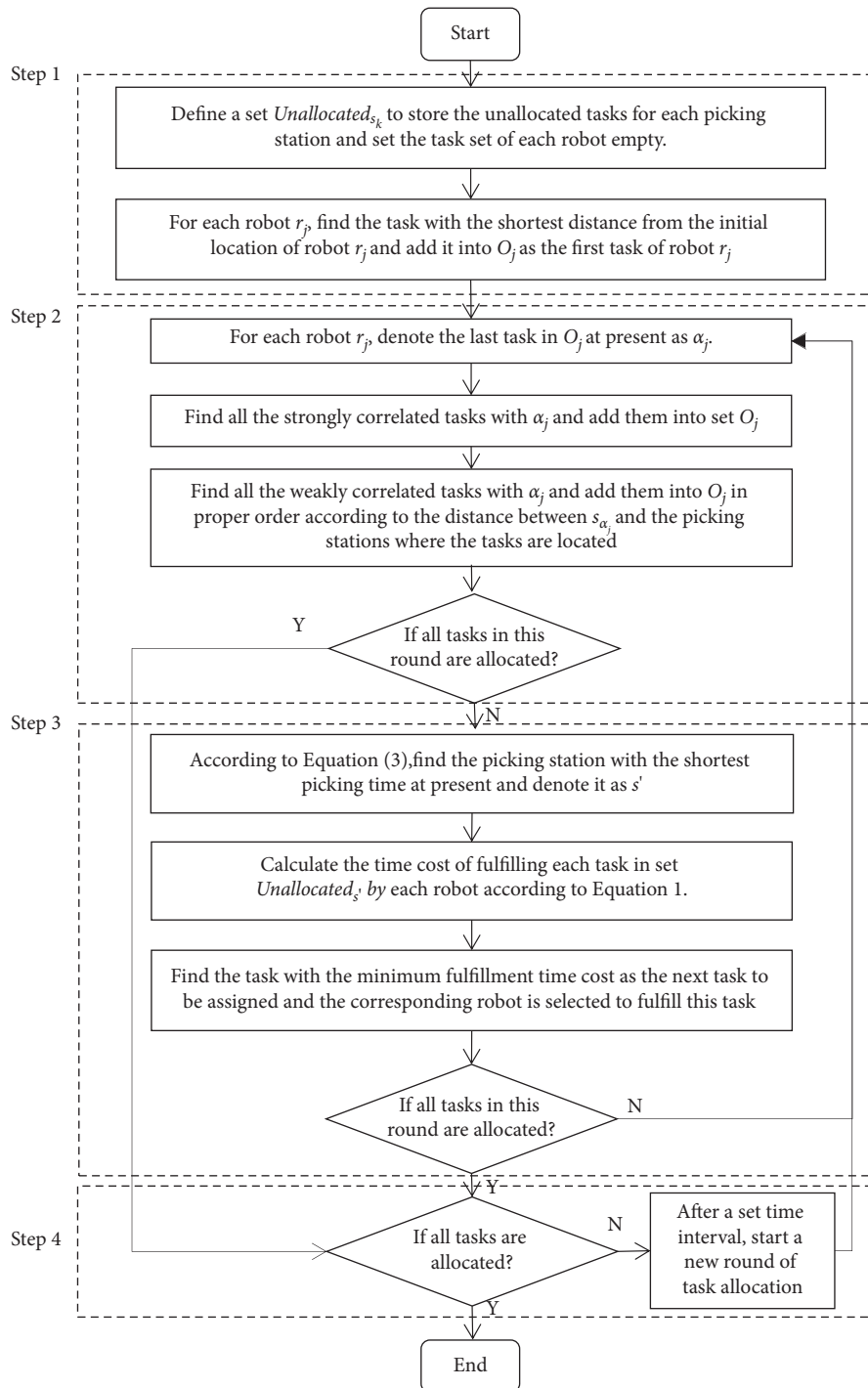


FIGURE 2: Flowchart of the proposed algorithm.

- (b) Find the task with the minimum fulfillment time cost, which is the next task to be assigned, and the corresponding robot is selected to fulfill this task.
- (c) If there are still tasks unallocated, go to Step 2; otherwise, go to Step 4.

4.4. Step 4: Dynamic Adjustment and Update. In the real operation process, task time errors may arise owing to congestion, communication delay, or other faults of logistics robots. However, the delay is not considered in the construction of the model. Consequently, if the auction of all tasks is conducted simultaneously, it may cause serious

```

Input: unallocated task set  $Unallocated_{s_k}$ , Time interval  $T$ 
Output: task allocation result  $O_j$  for each robot
(1) # Initial task allocation
(2) for ( $j = 1; j \leq n; j^{++}$ ) do
(3)   find the task with the shortest distance with  $r_j$  and put it in to set  $O_j$ 
(4) end
(5) #Task allocation considering task correlation
(6) for ( $j = 1; j \leq n; j^{++}$ ) do
(7)    $\alpha_j \leftarrow$  the last task in  $O_j$ 
(8)   find the tasks strongly related to  $\alpha_j$ , add them into  $O_j$ 
(9)   find the tasks weakly related to  $\alpha_j$ , add them into  $O_j$  in proper order
(10) end
(11) #task allocation based on auction algorithm
(12) for ( $k = 1; k \leq h; k^{++}$ ) do
(13)   calculate the picking time  $T'_k$  according to equation (3)
(14) end
(15)  $s' \leftarrow$  the picking station with the smallest picking time
(16) for ( $j = 1; j \leq n; j^{++}$ ) do
(17)   calculate the time cost of fulfilling each task in  $Unallocated_{s'}$  by  $r_j$  according to equation (1)
(18) end
(19) find the task with the minimum fulfillment time cost and allocate it to the corresponding robot

```

ALGORITHM 1: The main part of the balanced heuristic algorithm.

cumulative errors. So, in the paper, the auction is divided into many rounds. The time node and the number of tasks assigned in each round are set based on the total number of tasks. Each round of auction is performed according to Step 2 and Step 3. Finally, the allocation of tasks to robots and the sequence of tasks executed by each robot are obtained.

The pseudocode of the main part of the algorithm is expressed in Algorithm 1.

5. Simulation Analysis

The effectiveness of the logistics robot task allocation model and algorithm proposed in this paper is verified using the data of an e-commerce company. In the experiment, there are 3 picking stations and 300 shelves placed in a warehouse with an area of 800 m². The layout of the warehouse refers to the layout in Figure 1. There are more than 2,000 SKUs on sales in the company. In order to avoid system congestion and improve picking efficiency, a decentralized storage strategy is adopted. That is, goods with high turnover rate are stored on shelves in different areas (for specific storage allocation methods, see [26]). Picking tasks are generated from the historical order data of the company. Each item in an order corresponds to a picking task. 15 mobile robots are employed to fulfill these tasks. Other parameters used in the experiment are given in Table 1.

The location of the tasks and their picking stations are known. The location of robots can be known at any time, and the distance between any two positions is calculated using the Manhattan distance because of the kinematics constraints of robots. The time interval between each round of task allocation was set to 15 minutes. MATLAB 7.11 was used to calculate the task allocation results. The proposed model and algorithm are also compared with the traditional time cost model and the algorithm without considering equilibrium.

5.1. Task Allocation Scheme and Task Fulfillment Time.

The task allocation scheme is the task allocation result of robots. Based on the proposed task time cost model and the designed auction algorithm, we first obtained the solution for the allocation of 200 tasks, which are numbered from 1 to 200. The task allocation scheme, which includes the allocation of tasks to robots and the task execution sequence, and the time cost for each robot to perform the task are given in Table 2. The fulfillment time of all tasks is the longest picking time of all robots, which is 860 seconds in this case.

It can be seen from Table 2 that using the proposed task time cost model and considering the time balance of picking stations, we can get the reasonable assignment of tasks to robots and the execution sequence of tasks. Furthermore, the proposed algorithm can achieve the goal of load balance among logistics robots, which is conducive to the parallel operation mode of multiple robots.

5.2. Comparative Analysis between Different Task Time Cost Models.

Then, task allocation with different numbers of tasks (100, 200, 300, 400, 500) was used to compare the performance of the task time cost model proposed in this paper and the traditional task time cost model in [13]. The proposed time cost model in this paper considers task correlation, and the time cost of tasks with different types of correlation is calculated differently. The traditional task time cost model does not consider task correlation and the cost of a task is the time a robot spent performing the task alone, that is, the cost of every task is calculated according to equation (1) when $l = 1$. The total picking time of all tasks using these two models under different sizes of tasks is compared in Figure 3.

It can be seen from Figure 3 that the proposed task time cost model reduces the total picking time compared with the traditional task time cost model. Besides, with the increase of

TABLE 1: Parameters setting.

Parameter	Description	Value
n	Number of logistics robots	15
m	Number of tasks	[100, 500]
h	Number of picking stations	3
v'	Moving velocity of robots (m/s)	2
t'	Picking time of one item (s)	4

TABLE 2: Task allocation scheme and time cost for each logistics robot.

Robot	Task allocation scheme	Time cost (s)
1	32→177→121→193→20→88→114→86→176→144→131→38→106→190	746.5
2	62→25→93→186→81→162→165→153→111→149→169→150→157	858.0
3	136→75→115→5→14→125→188→134→69→79→130→73→40	797.5
4	120→41→180→108→84→76→109→178→160→175→61→36→107→60	809.0
5	66→39→71→118→16→68→124→123→55→64→37→57	822.5
6	12→30→151→187→17→96→65→173→159→191→139→135	809.5
7	142→99→164→33→45→95→113→101→80→58→140→198→138→9→167→194→48	752.0
8	19→70→148→146→27→13→197→179→133→155→152→158	843.5
9	63→145→163→90→102→18→185→94→119→47→117	747.5
10	28→100→182→170→98→77→34→132→143→168→174→200→195→49	860.0
11	46→52→22→104→89→192→184→199→26→54→29→44→50→1	793.5
12	35→8→4→24→171→53→82→129→15→67→10→172→21	851.0
13	78→181→97→116→161→85→56→31→74→91→72→105→42→196	765.0
14	183→87→43→112→3→147→189→154→2→6→92→127→11	745.0
15	59→103→83→156→126→128→110→51→122→23→7→137→166→141	787.0

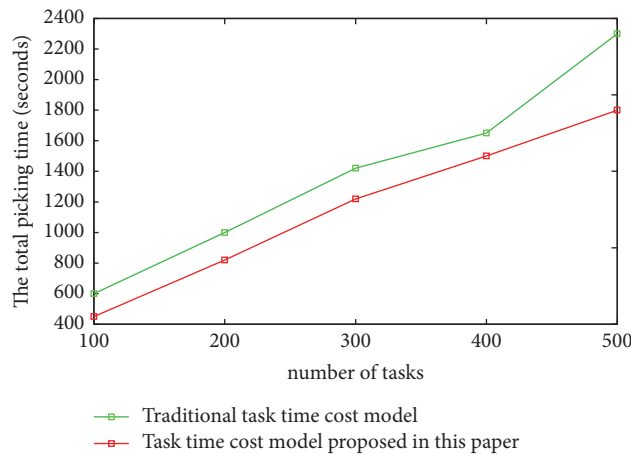


FIGURE 3: Comparison of the total piking time under two different time cost models.

TABLE 3: Influence of the balance among picking stations on task fulfillment time.

Number of tasks	Balance among picking stations considered?	Picking station 1 Time (seconds)	Picking station 2 Time (seconds)	Picking station 3 Time (seconds)
300	Yes	1271.5	1175.5	1269.5
	No	979.31	1532.01	1329.53
400	Yes	1488.5	1379.0	1560.5
	No	1732.7	1438.5	1867.2
500	Yes	1801.8	1610.5	1830.5
	No	2010.2	1953.28	2303.5

the number of tasks, the advantage of the proposed task time cost model is becoming more and more obvious. That is because, with the increase of the number of tasks, the correlation between tasks increases, and thus, considering task correlation can save more time.

5.3. Influence of the Balance among Picking Stations. Because of the parallel operation mode of multiple picking stations, the total fulfillment time of all tasks is determined by the longest picking time of all picking stations. Therefore, balancing the picking time among picking stations is an important part of the design of the task allocation algorithm. Different numbers of tasks (300, 400, 500) were experimented on for comparative analysis between our proposed algorithm and the algorithm without considering the balance of the picking stations. The results are given in Table 3, in which the time in bold is the fulfillment time of all tasks under that scenario.

It can be seen from Table 3 that the picking time of each picking station is relatively balanced and the completion time of all tasks is shorter when considering the picking time balance among picking stations. In contrast, without considering the balance of picking stations, the picking time of each picking station fluctuates greatly and it takes a longer time to complete all the tasks. As the balance among picking stations is not taken into account, the burden of each picking station is uneven, which may lead to long queues of some picking stations and idleness of other picking stations. In that case, the picking time difference between picking stations is relatively large, and the fulfillment time of all tasks is extended.

6. Conclusions and Prospects

This paper studied the multirobot task allocation problem in e-commerce RMFS. Combined with the operation process of the picking system, a task time cost model considering the correlation between tasks is proposed. The time balance among picking stations is added to multirobot task allocation model, and a four-stage balanced heuristic auction algorithm is designed to solve the model efficiently. Simulation results showed the proposed model and algorithm can significantly improve the efficiency of task allocation and reduce the fulfillment time of orders compared with the methods which do not consider the correlation between tasks and the balance among picking stations.

The proposed task allocation method in this paper only considers the standard operation process, which is applicable to the daily smooth operation of small- and medium-sized e-commerce enterprises. For task allocation in large-scale and changeable application scenarios, dynamic uncertain factors, such as order cancellation, order insertion, and traffic congestion, should be considered to improve efficiency. As reinforcement learning is suitable for dynamic and changeable environments and deep learning can handle the problem with high complexity and large space state, the MRTA method based on deep reinforcement learning may be a promising direction to solve the task allocation problem in large-scale robotic mobile fulfillment systems.

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 article.

Acknowledgments

This paper was funded by the National Natural Science Foundation of China (72101033 and 71831001), Beijing Key Laboratory of Intelligent Logistics Systems (BZ0211), Canal Plan-Youth Top-Notch Talent Project of Beijing Tongzhou District (YHQN2017014), and Ningxia Science and Technology Key Research and Development Project (2018BEG03003).

References

- [1] N. Boysen, R. De Koster, and F. Weidinger, "Warehousing in the E-commerce era: a survey," *European Journal of Operational Research*, vol. 277, no. 2, pp. 396–411, 2018.
- [2] D. Roy, S. Nigam, R. De Koster, I. Adan, and J. Resing, "Robot-storage zone assignment strategies in mobile fulfillment systems," *Transportation Research Part E: Logistics and Transportation Review*, vol. 122, pp. 119–142, 2019.
- [3] M. Wulfraat, "Is Kiva systems a good fit for your distribution center? an unbiased distribution consultant evaluation," 2012, http://www.mwvpl.com/html/kiva_systems.html.
- [4] X. Xu and Z. Ma, "Robotic mobile fulfillment systems: state-of-the-art and prospects," *Acta Automatica Sinica*, vol. 46, no. 9, pp. 1–25, 2020.
- [5] T. Lamballais, D. Roy, and M. B. M. D. Koster, "Estimating performance in a robotic mobile fulfillment system," *European Journal of Operational Research*, vol. 256, no. 3, pp. 976–990, 2016.
- [6] N. Boysen, D. Briskorn, and S. Emde, "Parts-to-picker based order processing in a rack-moving mobile robots environment," *European Journal of Operational Research*, vol. 262, no. 2, pp. 550–562, 2017.
- [7] Y. Yang, J. Li, and L. Peng, "Multi-robot path planning based on a deep reinforcement learning DQN algorithm," *CAAI Transactions on Intelligence Technology*, vol. 5, no. 3, pp. 177–183, 2020.
- [8] A. Khamis, A. Hussein, and A. Elmogy, "Multi-robot task allocation: a review of the state-of-the-art," in *Cooperative Robots and Sensor Networks 2015*, pp. 31–51, Springer, Berlin, Germany, 2015.
- [9] Y. Zhang and S. H. Liu, "Survey of multi-robot task allocation," *CAAI Transactions on Intelligent Systems*, vol. 3, no. 2, pp. 115–120, 2008.
- [10] F. Tang and L. E. Parker, "A complete methodology for generating multi-robot task solutions using ASyMTRE-D and market-based task allocation," in *Proceedings of the International Conference on Robotics and Automation*, IEEE, Roma, Italy, April 2007.
- [11] J. J. Zheng, J. H. Dong, and Z. Y. Guan, "etc. "Swarm simulation of combination auction model based on PSO," *System Engineering-Theory & Practice*, vol. 36, no. 12, pp. 3142–3151, 2016.

- [12] R. Zlot, A. Stentz, M. B. Dias, and S. Thayer, "Multi-robot exploration controlled by a market economy," in *Proceedings of the 2002 IEEE International Conference on Robotics and Automation*, pp. 3016–3023, Washington, DC, USA, May 2002.
- [13] M. Elango, S. Nachiappan, and M. K. Tiwari, "Balancing task allocation in multi-robot systems using K-means clustering and auction based mechanisms," *Expert Systems with Applications*, vol. 38, no. 6, pp. 6486–6491, 2011.
- [14] G. Lozenguez, L. Adouane, A. Beynier, A. I. Mouaddib, and P. Martinet, "Punctual versus continuous auction coordination for multi-robot and multi-task topological navigation," *Autonomous Robots*, vol. 40, no. 4, pp. 1–15, 2015.
- [15] B. Heap and M. Pagnucco, "Repeated sequential single-cluster auctions with dynamic tasks for multi-robot task allocation with pickup and delivery," in *Multiagent System Technologies*, pp. 87–100, Springer, Berlin, Germany, 2013.
- [16] L. Zhou, Y. Shi, J. Wang, and P. Yang, "A balanced heuristic mechanism for multirobot task allocation of intelligent warehouses," *Mathematical Problems in Engineering*, vol. 2014, Article ID 380480, 10 pages, 2014.
- [17] C. Liu and A. Kroll, "Memetic algorithms for optimal task allocation in multi-robot systems for inspection problems with cooperative tasks," *Soft Computing*, vol. 19, no. 3, pp. 567–584, 2015.
- [18] J. Dou, C. Chen, and P. Yang, "Genetic scheduling and reinforcement learning in multirobot systems for intelligent warehouses," *Mathematical Problems in Engineering*, vol. 2015, Article ID 597956, 10 pages, 2015.
- [19] T. Lamballais, D. Roy, and M. B. M. De Koster, "Estimating performance in a robotic mobile fulfillment system," *European Journal of Operational Research*, vol. 256, no. 3, pp. 976–990, 2017.
- [20] Y. D. Li, "Model and algorithm for cartonization and slotting optimization simultaneously in wave-picking zone-based system," *Systems Engineering—Theory & Practice*, vol. 33, no. 5, pp. 1269–1276, 2013.
- [21] P. R. Wurman, M. Mountz, and M. Mountz, "Coordinating hundreds of cooperative, autonomous vehicles in warehouses," in *Proceedings of the National Conference on Innovative Applications of Artificial Intelligence*, pp. 1752–1759, AAAI Press, Chicago, IL, USA, July 2008.
- [22] J. J. Enright and P. R. Wurman, "Optimization and coordinated autonomy in mobile fulfillment systems," in *Proceedings of the AAAI Conference on Automated Action Planning for Autonomous Mobile Robots*, pp. 33–38, AAAI Press, San Francisco, CA, USA, August 2011.
- [23] B. P. Gerkey and M. J. Mataric, "A formal analysis and taxonomy of task allocation in multi-robot systems," *The International Journal of Robotics Research*, vol. 23, no. 9, pp. 939–954, 2004.
- [24] M. Nanjanath and M. Gini, "Repeated auctions for robust task execution by a robot team," *Robotics and Autonomous Systems*, vol. 58, no. 7, pp. 900–909, 2010.
- [25] L. Luo, N. Chakraborty, and K. Sycara, "Distributed algorithms for multirobot task assignment with task deadline constraints," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 3, pp. 876–888, 2015.
- [26] R. Yuan, H. Wang, J. Li, and K. Lui, "The slotting optimization model and algorithm in robotic mobile fulfillment systems," *System Science and Mathematics*, vol. 40, no. 6, pp. 1050–1060, 2020.

Research Article

A Novel Crow Search Algorithm Based on Improved Flower Pollination

Qian Cheng ¹, Huajuan Huang ², and Minbo Chen ¹

¹College of Electronic Information, Guangxi University for Nationalities, Nanning 530000, China

²College of Artificial Intelligence, Guangxi University for Nationalities, Nanning 530000, China

Correspondence should be addressed to Huajuan Huang; hhj-025@163.com

Received 11 August 2021; Revised 18 September 2021; Accepted 22 September 2021; Published 26 October 2021

Academic Editor: Qingzheng XU

Copyright © 2021 Qian Cheng et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Crow search algorithm (CSA) is a new type of swarm intelligence optimization algorithm proposed by simulating the crows' intelligent behavior of hiding and retrieving food. The algorithm has the characteristics of simple structure, few control parameters, and easy implementation. Like most optimization algorithms, the crow search algorithm also has the disadvantage of slow convergence and easy fall into local optimum. Therefore, a crow search algorithm based on improved flower pollination algorithm (IFCSA) is proposed to solve these problems. First, the search ability of the algorithm is balanced by the reasonable change of awareness probability, and then the convergence speed of the algorithm is improved. Second, when the leader finds himself followed, the cross-pollination strategy with Cauchy mutation is introduced to avoid the blindness of individual location update, thus improving the accuracy of the algorithm. Experiments on twenty benchmark problems and speed reducer design were conducted to compare the performance of IFCSA with that of other algorithms. The results show that IFCSA has better performance in function optimization and speed reducer design problem.

1. Introduction

In modern societies where global resources are increasingly scarce, optimization has become one of the most important and hottest research topics [1]. It is in the core process of various engineering fields such as engineering, science, energy, and computer. With the increasing complexity of scientific and engineering problems, traditional mathematical methods sometimes cannot solve them well. Therefore, some scholars have proposed metaheuristic algorithms, such as particle swarm optimization (PSO) [2], bat algorithm (BA) [3], butterfly optimization algorithm (BOA) [4], flower pollination algorithm (FPA) [5], pigeon inspired optimization (PIO) [6], whale optimization algorithm (WOA) [7], gray wolf optimizer (GWO) [8], and teaching-learning-based optimization (TLBO) [9]. Compared with traditional algorithms, these intelligent algorithms can make up for the defects of the traditional algorithm in the problem of great complexity. However, they still have the problems of low solution accuracy and slow convergence speed.

Crow search algorithm (CSA) [10] is a new swarm intelligence optimization algorithm proposed by Askarzadeh in 2016. The algorithm has the advantages of simple structure, fewer control parameters, and easy implementation. Now it has been widely used to solve various practical optimization problems in chemical engineering and QSAR [11], image processing [12], feature selection [13], neural network and support vector machine [14], aircraft maintenance inspection [15], wireless sensor network [16], and other major engineering fields. However, like most optimization algorithms, crow search algorithm itself also has the defects of slow convergence speed and easy fall into local optimum.

In view of the shortcomings of crow search algorithm, many scholars have put forward their own improvement schemes, which can be roughly divided into two categories. One is to analyze and improve the parameters of the standard crow search algorithm, and the other is to integrate it with other intelligent algorithms to make up for the shortcomings of crow search algorithm. For the first kind of

improvement, Wu et al. proposed a crow search algorithm incorporated in Levy flight (LFCSA) [17] and applied it to update the finite element model. The chaotic sequence was used to initialize the population, so that the particles were evenly distributed in the solution space and the diversity of the population was increased. Therefore, Liu et al. proposed the chaotic binary crow search algorithm (CBCSA) [18] for the discrete space, using it to solve the problem of {0–1} knapsack. By introducing adaptive step size, Mohammadi and Abdi proposed self-adaptive step size crow search algorithm (MCSA) [19] and applied it to nonconvex economic load scheduling. For the second kind of improvement, Xiao et al. [20] combined CSA with sine cosine algorithm to solve pressure vessel design problem. Arora et al. [21] combined the crow search algorithm with the gray Wolf algorithm and used it to solve the problem of feature selection.

All the above improved algorithms have improved the performance of the algorithm to some extent, but some improvements were only optimized for a certain strategy in the crow update mechanism or simply mixed optimization with other optimization algorithms. For example, LFCSA uses Levy flight strategy, CBCSA takes advantage of the particularity of chaos mapping, and MCSA introduces self-adaptive flight step. Although these improved strategies can make the algorithm jump out of local optimum better, they cannot make up for the slow convergence speed. Other improvements are simply hybrid optimizations with other algorithms. This kind of improvement scheme ignores the limitations of the fusion algorithm itself, which also makes the optimization ability of the improved algorithm have some defects. In response to these shortcomings, this article mainly improves the standard crow search algorithm from three aspects: increasing population diversity, self-adaptive awareness probability, and improved cross-pollination mechanism of flower pollination algorithm. The population is initialized by tent chaotic sequence so that the particles are evenly distributed in the search space, and the increase of population diversity enables the algorithm to better jump out of the local optimum and accelerate the convergence speed. The next strategies are self-adaptive awareness probability and improved cross-pollination mechanism. It is beneficial to balance the global search ability and local search ability of the crow search algorithm and avoid the blindness of updating the random search position of crows, thus improving the solution accuracy and convergence speed of the algorithm.

In Section 2 of the paper, we will review the crow search algorithm and the flower pollination algorithm. In Section 3, the improved strategy will be described to produce an improved crow search algorithm. In Section 4, the proposed algorithm will be tested by using 20 well-known benchmark functions and applied to a practical engineering problem. Finally, the conclusion is given in Section 5.

2. Overview of Crow Search Algorithm and Flower Pollination Algorithm

2.1. Crow Search Algorithm. The crow is a very clever bird, which can remember the face of human beings and warn its kind when encountering danger. One of the most obvious characteristics of the cleverness of crows is their ability to hide food and remember the location of the hidden food. At the same time, they will follow each other to get a better source of food, but when the crow finds itself followed by other crows, it will try to change the hidden place of its food to avoid food theft. Based on the living habits of crows, the crow search algorithm has the following principles:

- (1) Crows are social animals
- (2) Crows can remember the location of the hidden food
- (3) Crows will follow each other and steal others' food
- (4) Crows do their best to protect their food from being stolen

Based on the four principles, the basic process of CSA is described as follows:

Step 1: initializing the parameters of CSA. These include population size (n), maximum iteration number (Maxsize), flight step size (fl), and awareness probability (AP).

Step 2: initializing the individual crows and memory matrix. n crows are generated in the search space of d – dimension, and each crow $x_i = (X_{i,1}, X_{i,2}, \dots, X_{i,d})$ represents a feasible solution of a problem. Since the initial population has no experience, it is assumed that the initial memory matrix is the initial position.

Step 3: evaluating the quality of each crow according to the fitness function.

Step 4: generating a new location for each crow in the d – dimensional search space. Assuming that crow i randomly follows a crow (for example, crow j), in order to find the place of the hidden food of crow j , the position update of crow i can be divided into the following two situations:

Case 1: crow j does not find that crow i is following it. In this case, the position update formula of crow i is

$$x^{i,iter+1} = x^{i,iter} + r_i + fl^{i,iter} \times (m^{i,iter} - x^{i,iter}). \quad (1)$$

Case 2: crow j finds that crow i is following it, and crow j will take crow i to a random position.

To sum up, the position update formula of crow i is

$$x^{i,iter+1} = \begin{cases} x^{i,iter} + r_i \times fl^{i,iter} \times (m^{i,iter} - x^{i,iter}), & r_j \geq AP, \\ a \text{ random position}, & \text{otherwise.} \end{cases} \quad (2)$$

Here, r_i, r_j are random numbers that obey the uniform distribution of $[0, 1]$. AP represents the perception probability. When the AP is smaller, the probability of occurrence of Case 1 is greater, and the algorithm tends to search locally. When the AP is larger, the probability of finding in Case 2 is greater, and the algorithm tends to search globally. $fl^{i,iter}$ is the flight step length of crow i . When $fl^{i,iter} < 1$, the next position of crow i is between $x^{i,iter}$ and $m^{j,iter}$, as shown in Figure 1. When $fl^{i,iter} > 1$, the next position of crow i is outside the line between $x^{i,iter}$ and $m^{j,iter}$, as shown in Figure 2. Therefore, fl will affect the search ability of the algorithm. If the value is too large, it tends to search globally, and the algorithm has poor convergence. If the value is too small, it is easy to fall into the local optimum.

Step 5: checking whether the new position of each crow is feasible. If possible, change the crow's position. Otherwise, it is not updated.

Step 6: calculating the fitness value of the new position of each crow.

Step 7: updating the memory matrix of each crow.

$$m^{i,iter+1} = \begin{cases} x^{i,iter+1}, & f(x^{i,iter+1}) \text{ is better than } f(m^{i,iter}), \\ x^{i,iter}, & \text{otherwise.} \end{cases} \quad (3)$$

Step 8: repeating Steps 4–7 until the termination condition is reached.

2.2. Flower Pollination Algorithm. Flower pollination algorithm is a new metaheuristic swarm intelligence optimization algorithm proposed by Yang, a scholar from Cambridge University, UK, in 2012. The algorithm simulates the two processes of cross-pollination and self-pollination of flowering plants, which corresponds to the global search mechanism and local search mechanism in the algorithm. Because the algorithm has few parameters, simple structure, and easy implementation, it has widely attracted the interest of other scholars.

In order to simplify the problem and make the algorithm more efficient, considering that there is only one solution to the optimization problem, Yang assumes that each flowering plant can only produce one flower and each flower can only produce one pollen gamete. Flower pollination process can be summarized as the following four rules:

- (1) Biological cross-pollination is considered a global pollination process in which pollen carriers carry pollen on Levy flights
- (2) Abiotic self-pollination is regarded as a local pollination process
- (3) The reproduction probability is the constancy of flowers, and the value of reproduction probability is proportional to the similarity of two flowers

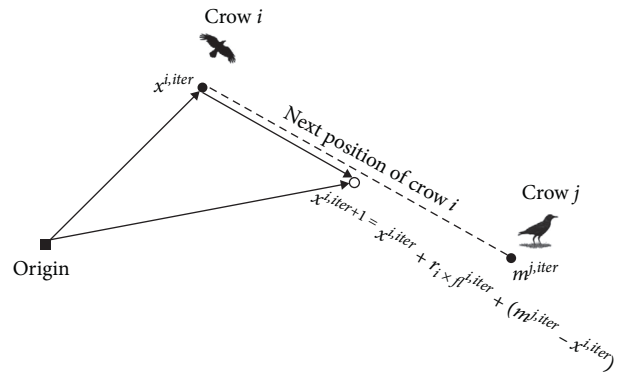


FIGURE 1: Schematic diagram for updating the position of crow i when $fl < 1$.

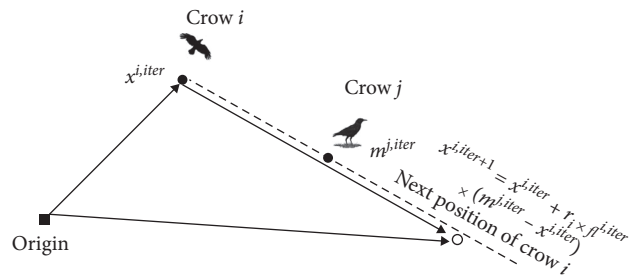


FIGURE 2: Schematic diagram for updating the position of crow i when $fl > 1$.

- (4) The transition probability $p \in [0, 1]$ is used to control the transition between local pollination and global pollination

Through the above rules, the following mathematical model is established.

Definition 1. In the process of global pollination, the formula of pollen position update is

$$x^{i,iter+1} = x^{i,iter} + L \times (x^{i,iter} - gbest). \quad (4)$$

Here, $x^{i,iter+1}$ and $x^{i,iter}$, respectively, represent the solution of the iter + 1 generation and the iter generation; $gbest$ is the global optimal solution in an iteration process; and L is the step size, which obeys the Levy distribution. The calculation formula of L is as follows:

$$L \sim \frac{\lambda \Gamma(\lambda) \sin(\pi\lambda/2)}{\pi} \cdot \frac{1}{S^{1+\lambda}} (S \gg S_0 \gg 0). \quad (5)$$

Here, $\Gamma(\lambda)$ is the standard gamma function; $\lambda = 1.5$.

Definition 2. The position update formula for the partial pollination stage is as follows:

$$x^{i,iter+1} = x^{i,iter} + \varepsilon \times (x^{j,iter} - x^{k,iter}). \quad (6)$$

Here, $x^{j,iter}$ and $x^{k,iter}$ randomly select solutions different from $x^{i,iter}$ in the population, and ε is the probability of

reproduction, which is a random number subject to the uniform distribution of $[0, 1]$.

Definition 3. The transition between global pollination and local pollination is controlled by the value of transition probability $p \in [0, 1]$. A large number of simulation experiments show that the algorithm can obtain the best optimization performance when $p = 0.8$.

3. Crow Search Algorithm Based on Improved Flower Pollination Algorithm (IFCSA)

In order to improve the convergence speed and accuracy of the algorithm, two strategies are used, namely, self-adaptive awareness probability and improved cross-pollination mechanism of flower pollination algorithm. The following is a detailed description of these three strategies.

3.1. Self-Adaptive Awareness Probability. The awareness probability has a greater impact on the performance of the standard crow search algorithm. When the perception probability AP is larger, the individuals in the population are more inclined to search globally, but it is not conducive to improving the convergence accuracy. When the perception probability AP is smaller, the individuals in the population are more inclined to search locally and easily fall into the local optimum. However, when AP is a fixed value, the global search and local search capabilities cannot be balanced. In response to the problem, the inverse incomplete Γ function is introduced to make the awareness probability drop nonlinearly, so as to balance the global search and local search capabilities. The formula is as follows:

$$AP = 1 \div \left(\left(AP_1 \times \left(\frac{AP_2 - AP_1}{\lambda} \right) \times \Gamma \left(\lambda, \left(1 - \frac{\text{iter}}{\text{Maxiter}} \right) \right) \right) \times 100 \right). \quad (7)$$

Here, AP_2 and AP_1 are, respectively, the upper and lower limits of AP; λ is a random variable; and Maxiter is the maximum number of iterations.

Figure 3 shows the decreasing curve of nonlinear awareness probability AP when $AP_1 = 0.05$, $AP_2 = 0.25$, and $\lambda = 0.01$. It can be seen from Figure 3 that the AP is large at the beginning of iteration, which makes the algorithm focus on global search. With the iterative process, the AP value decreases gradually, which makes the algorithm tend to search locally, the population concentrate quickly, and the convergence process improve. Nonlinear decreasing awareness probability can better balance global search and local search, thus improving the performance of the algorithm.

3.2. Based on Improved Cross-Pollination. From the analysis of (2), it can be seen that when the leader finds that he is being followed, the position of the individual is randomly generated. Although this location update strategy can prevent the algorithm from falling into the local optimum to a certain extent, it also reduces the convergence speed and

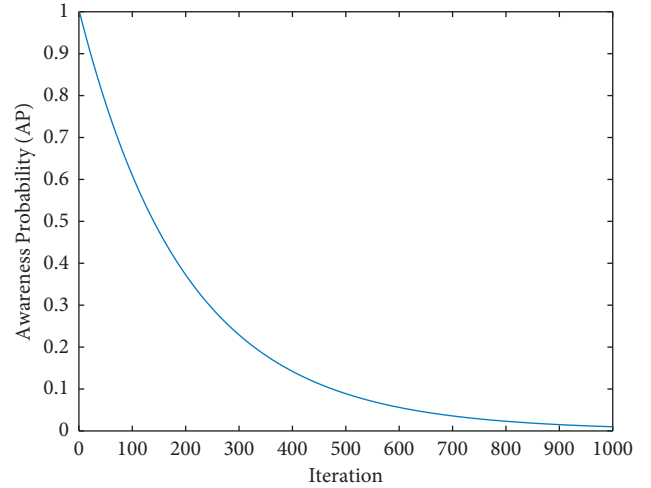


FIGURE 3: Decreasing curve of nonlinear awareness probability.

accuracy of the algorithm. For this reason, this article solves this problem by introducing improved cross-pollination.

Firstly, it is assumed that all crows obtain the global optimal position based on their own experience and memory as thieves. Secondly, the idea of cross-pollination is used to effectively guide the crow individuals to fly close to the optimal individual to reach the optimal value. However, the cross-pollination strategy itself has some limitations. In the later stage of iteration, it will lead to the decrease of population diversity, which will result in falling into local optimum and difficulty in jumping out. Introducing the cross-pollination strategy of Cauchy mutation proposed by Zhang and Gao [22], the formula is as follows:

$$x^{i,\text{iter}+1} = g\text{best} + x^{i,\text{iter}} \times C(0, 1). \quad (8)$$

Here, $g\text{best}$ is the optimal value and $C(0, 1)$ is Cauchy distribution.

To sum up, the crow position update formula is as follows:

$$x^{i,\text{iter}+1} = \begin{cases} x^{i,\text{iter}} + r_i \times f^{l,\text{iter}} \times (m^{i,\text{iter}} - x^{i,\text{iter}}), & r_j \geq AP, \\ g\text{best} + x^{i,\text{iter}} \times C(0, 1), & \text{otherwise.} \end{cases} \quad (9)$$

3.3. Basic Flow of IFCSA. The pseudocode of the algorithm (IFCSA), which is based on the improved flower pollination algorithm, is shown in Figure 4.

4. Experimental Simulation and Result Analysis

4.1. Experimental Parameter Settings. The IFCSA is validated by comparing it with five famous optimization algorithms, namely, BOA [4], SSA [23], GWO [8], CSA [10], and MISCOSA [24]. We used 20 well-known benchmark test functions for validation. The set of benchmark test functions is explained in Tables 1~3. We used the most common parameter settings that exist in the literature for the seven

```

Initialize the initial position
for iter = 1 : Maxiter
  for i = 1 : n
    Calculate the AP
    if  $r_j = AP$ 
       $x^{i,iter+1} = x^{i,iter} + r_i \times f^{i,iter} \times (m^{i,iter} - x^{i,iter})$ 
    else
       $x^{i,iter+1} = gbest + x^{i,iter} \times C(0,1)$ 
    endif
  endfor
  Check boundary
  Evaluate the new position of crows
  Update the memory matrix of each crow
endfor

```

FIGURE 4: Pseudocode of the IFCSA.

TABLE 1: High-dimensional unimodal benchmark functions.

Benchmark function	Dim	Range	f_{\min}
$f_1 = \sum_{i=1}^n x_i^2$	30	[-100, 100]	0
$f_2 = \sum_{i=1}^n x_i + \prod_{i=1}^n x_i $	30	[-10, 10]	0
$f_3 = \max\{ x_i , 1 \leq i \leq n\}$	30	[-100, 100]	0
$f_4 = \sum_{i=1}^n (\sum_{j=1}^i x_j)^2$	30	[-100, 100]	0
$f_5 = \sum_{i=1}^n i x_i^4 + \text{random}[0, 1)$	30	[-1.28, 1.28]	0
$f_6 = 10^6 x_1^2 + \sum_{i=2}^n x_i^2$	30	[-100, 100]	0
$f_7 = \sum_{i=1}^n 10^{6(i-1)/(n-1)} x_i^2$	30	[-100, 100]	0

TABLE 2: High-dimensional multimodal benchmark functions.

Benchmark function	Dim	Range	f_{\min}
$f_8 = \sum_{i=1}^n [x_i^2 - 10 \cos(2\pi x_i) + 10]$	30	[-5.12, 5.12]	0
$f_9 = -20 \exp(-0.2 \sqrt{1/n \sum_{i=1}^n x_i^2}) - \exp(1/n \sum_{i=1}^n \cos(2\pi x_i)) + 20 + e$	30	[-32, 32]	0
$f_{10} = 1/4000 \sum_{i=1}^n x_i^2 - \prod_{i=1}^n \cos(x_i/\sqrt{i}) + 1$	30	[-600, 600]	0
$f_{11} = \sum_{i=1}^n x_i \sin(x_i) + 0.1 x_i $	30	[-10, 10]	0
$f_{12} = \sum_{i=1}^{n/4} [(x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + (x_{4i-2} - 2x_{4i-1})^4 + 10(x_{4i-3} - x_{4i})^4]$	30	[-4, 5]	0
$f_{13} = \sin(\pi w_1) + \sum_{i=1}^{n-1} (w_i - 1)^2 [1 + 10 \sin^2(\pi w_i)] + (w_n - 1)^2 [1 + \sin^2(2\pi w_n)]$, where $w_i = 1 + x_i + 1/4$, for all $i = 1, 2, 1, \dots, n$	30	[-10, 10]	0
$f_{14} = 0.1 \{ \sin^2(32\pi x_1) + \sum_{i=1}^n (x_i - 1)^2 [1 + \sin^2(3\pi x_i + 1)] + (x_n - 1)^2 [1 + \sin^2(2\pi x_n)] \}$ $+ \sum_{i=1}^n u(x_i, 5, 100, 4)$	30	[-50, 50]	0
$u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a \\ 0, & -a < x_i < a \\ k(-x_i - a)^m, & x_i < -a \end{cases}$			

TABLE 3: Fixed-dimensional multimodal benchmark functions.

Benchmark function	Dim	Range	f_{\min}
$f_{15} = 0.5 + \sin^2(\sqrt{x_1^2 + x_2^2 + 0.5}) / (1.0 + 0.0001(x_1^2 + x_2^2))^2$	2	[-100, 100]	0
$f_{16} = (1.5 - x_1 + x_1 x_2)^2 + (2.25 - x_1 + x_1 x_2^2)^2 + (2.625 - x_1 + x_1 x_2^3)^2$	2	[-4.5, 4.5]	0
$f_{17} = \sum_{i=1}^{11} [a_i - x_1 (b_i^2 + b_i x_2) / b_i^2 + b_i x_3 + x_4]^2$	4	[-5, 5]	0.00030
$f_{18} = 2x_1^2 - 1.05x_1^4 + x_1^6/6 + x_1 x_2 + x_2^2$	2	[-5, 5]	0
$f_{19} = (x_2 - \frac{5.1}{4\pi^2} x_1^2 + \frac{5}{\pi} x_1 - 6)^2 + 10(1 - 1/8\pi) \cos x_1 + 10$	2	[-5, 5]	0.398
$f_{20} = [1 + (x_1 + x_2 + 1)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1 x_2 + 3x_2^2)] \times [30 + (2x_1 - 3x_2)^2 (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1 x_2 + 27x_2^2)]$	2	[-2, 2]	3

algorithms used in validation. The details about parameter settings are explained in Table 4. For the sake of fairness, we used a fixed population size for all algorithms: $n = 50$ individuals; all algorithms are executed in 30 independent runs. All algorithms are implemented in MATLAB (version R2020a) and executed on HP computer (Windows 10, Intel Core i5-6300HQ, 2.3 GHz, 8 GB RAM). The number of iterations in each run for each algorithm equals 1000. The experimental data obtained after running are shown in Tables 5~7, where Best, Worst, Mean, and Std represent the optimal value, worst value, average value, and Std value obtained by the algorithm independently running 30 times. The algorithm performance is ranked according to the accuracy of the optimal value. The best results are formatted in bold type.

4.2. Improved Strategy Effectiveness Analysis

4.2.1. Effectiveness Analysis of Self-Adaptive Awareness Probability. The awareness probability is one of the decision variables of the standard crow search algorithm, which has a greater impact on the performance of the algorithm. In order to verify whether the adaptive decreasing perceived probability can improve the convergence speed of the algorithm. Six groups of benchmark test functions in Table 1~3 are selected for comparative experiment. Among them, f_1 and f_2 are high-dimensional unimodal functions, f_8 and f_{10} are high-dimensional multimodal functions, and f_{14} and f_{16} are fixed multimodal functions.

As can be seen from Figures 5~8, for f_1, f_2, f_8, f_{10} , and f_{14} , IFCSA with adaptive change awareness probability has better convergence effect under the same number of iterations. It can be seen from Figure 10 that, for function f_{16} , although the convergence effect of IFCSA with adaptively changing awareness probability is worse than IFCSA without adaptively changing perception probability, it can find the theoretical optimal value, which is acceptable.

4.2.2. Effectiveness Analysis of the Cross-Pollination Strategy with Cauchy Mutation. This sections aims to verify whether the improved strategy can avoid the blindness of individual location update and improve the accuracy and convergence ability of the algorithm. At the same time, it aims to further clarify that under the condition that the leader finds himself followed, the performance of the algorithm under the cross-pollination mechanism with Cauchy mutation is better than that under the standard cross-pollination mechanism. The standard crow search algorithm, the crow search algorithm that introduces the standard cross-pollination strategy, and the crow search algorithm that introduces the alienation pollination strategy with Cauchy mutation are used for comparison experiments on 20 benchmark functions. They were run independently 30 times, and the average value was taken as the final criterion.

From the simulation results in Table 8, it can be seen that, for most benchmark functions, the optimization performance of the algorithm that introduces the standard cross-pollination strategy is worse than that of the standard crow

search algorithm. In the benchmark function f_{16} , the optimization performance of the algorithm that introduces the standard cross-pollination strategy is the best, but the performance of the algorithm that introduces the cross-pollination strategy with Cauchy mutation is also good, which is acceptable. However, in the 20 benchmark functions, the standard crow search algorithm introduces the cross-pollination strategy with Cauchy mutation, the performance of the algorithm is greatly improved, and the effect is better than the introduction of the standard cross-pollination strategy, except for the benchmark function f_{16} . In short, the crow search algorithm that introduces the cross-pollination strategy with Cauchy mutation has better performance in function optimization.

4.3. Comparative Experiment of Different Intelligent Algorithms

4.3.1. High-Dimensional Unimodal Function Test Results. It can be seen from the experimental results in Table 5 that the optimization performance of IFCSA is better than that of the other algorithms for most high-dimensional unimodal functions. In the test function of f_5 , although the performance of IFCSA is inferior to that of MISCSEA, the accuracy obtained by them is almost the same. In the other six test functions of $f_1, f_2, f_3, f_4, f_6, f_7$, compared with other algorithms, the optimization accuracy of IFCSA is improved to a certain extent. In addition, the variance of IFCSA is lower than that of the other algorithms, which indicates that IFCSA has good stability. In a word, IFCSA can solve high-dimensional unimodal problems well, which demonstrates its effectiveness and feasibility in solving high-dimensional unimodal problems.

Figures 11~24 are the convergence curve graphs and variance graphs of the algorithms of BOA [4], SSA [23], GWO [8], CSA [10], MISCSEA [24], and IFCSA for different test functions. All convergence curves are drawn based on the optimal values. Figures 13 and 21 show that GWO converges faster than IFCSA in the middle of the iteration, but GWO will fall into the local optimum, and IFCSA can jump out of the local optimum to get a better solution. It can be seen from Figures 11, 15, 17, and 23 that the convergence speed and optimization accuracy of IFCSA are better than those of other algorithms. As can be seen from Figures 12, 14, 16, 18, 20, 22, and 24, the performance of IFCSA algorithm is very stable. This shows that IFCSA can better solve high-dimensional unimodal problems compared to other intelligent algorithms and some improved algorithms.

4.3.2. High-Dimensional Multimodal Function Test Results. The experimental data in Table 6 shows that the optimal value and average value of IFCSA for functions $f_8 \sim f_{13}$ rank first in the comparison algorithm, which shows that IFCSA can solve most high-dimensional multimodal functions well. In the test function of f_{14} , the optimal value obtained by IFCSA is slightly smaller than those of GWO and SSA, ranking third in the comparison algorithm. Experimental results show that IFCSA can effectively solve high-dimensional multimodal functions and has strong global search capabilities.

TABLE 4: Parameter settings.

Algorithm	Parameter settings
BOA	$p = 0.8, c = 0.01, \alpha = 0.1$
FPA	$p = 0.8$
CSA	$AP = 0.1, fl = 2$
MISCSA	$AP = 0.1, fl = 2, \alpha_1 = 0.4, \alpha_2 = 0.7, \delta_1 = 0.4, \delta_2 = 0.001, s = 0.8$
IFCSA	$AP = 0.1, fl = 2$

TABLE 5: Results of high-dimensional unimodal benchmark functions.

Function	Index	BOA	SSA	GWO	CSA	MISCSA	IFCSA
f_1	Best	$1.53E-14$	$5.66E-09$	$1.40E-68$	$3.41E-03$	$7.59E-33$	$2.26E-102$
	Worst	$1.85E-14$	$1.40E-08$	$1.26E-72$	$3.73E-02$	$7.83E-30$	$1.03E-89$
	Mean	$1.69E-14$	$8.84E-09$	$9.11E-70$	$1.36E-02$	$7.98E-31$	$6.03E-91$
	Std	$9.60E-16$	$1.74E-09$	$2.82E-69$	$7.20E-03$	$1.56E-30$	$2.03E-90$
	Rank	4	5	2	6	3	1
f_2	Best	$2.02E-12$	$1.56E-04$	$3.43E-42$	$3.80E-01$	$3.85E-17$	$1.69E-50$
	Worst	$1.17E-11$	$3.05E+00$	$4.46E-40$	$2.75E+00$	$2.02E-15$	$4.66E-43$
	Mean	$9.19E-12$	$6.77E-01$	$6.50E-41$	$1.47E+00$	$4.55E-16$	$1.56E-44$
	Std	$2.97E-12$	$7.52E-01$	$9.17E-41$	$6.29E-01$	$4.20E-16$	$8.51E-44$
	Rank	4	5	2	6	3	1
f_3	Best	$1.01E-11$	$8.78E-01$	$1.46E-18$	$1.22E+00$	$5.80E-17$	$1.57E-50$
	Worst	$1.27E-11$	$9.46E+00$	$6.98E-17$	$5.48E+00$	$2.28E-15$	$4.79E-44$
	Mean	$1.15E-11$	$4.50E+00$	$1.25E-17$	$3.09E+00$	$5.24E-16$	$2.87E-45$
	Std	$6.68E-13$	$2.38E+00$	$1.57E-17$	$1.02E+00$	$5.20E-16$	$1.06E-44$
	Rank	4	5	2	6	3	1
f_4	Best	$1.48E-14$	$5.93E+00$	$4.57E-26$	$4.07E+02$	$1.06E-30$	$4.02E-100$
	Worst	$1.89E-14$	$2.51E+02$	$4.53E-17$	$1.53E+03$	$1.33E-27$	$1.38E-88$
	Mean	$1.70E-14$	$6.46E+01$	$1.55E-18$	$8.79E+02$	$2.52E-28$	$6.89E-90$
	Std	$9.12E-16$	$5.98E+01$	$8.27E-18$	$3.26E+02$	$3.93E-28$	$2.59E-89$
	Rank	4	5	3	6	2	1
f_5	Best	$8.80E-05$	$2.27E-02$	$9.33E-05$	$7.07E-03$	$6.38E-06$	$9.98E-06$
	Worst	$1.18E-03$	$1.16E-01$	$1.35E-03$	$3.99E-02$	$3.53E-04$	$3.39E-04$
	Mean	$5.96E-04$	$5.79E-02$	$5.17E-04$	$2.09E-02$	$5.35E-05$	$9.12E-05$
	Std	$3.01E-04$	$2.28E-02$	$3.35E-04$	$7.87E-03$	$6.36E-05$	$7.17E-05$
	Rank	3	6	4	5	1	2
f_6	Best	$1.34E-14$	$3.45E+03$	$7.91E-72$	$6.67E+02$	$5.90E-30$	$2.17E-97$
	Worst	$1.81E-14$	$3.09E+04$	$1.18E-68$	$2.62E+03$	$6.76E-27$	$7.54E-88$
	Mean	$1.61E-14$	$8.82E+03$	$1.08E-69$	$1.45E+03$	$9.79E-28$	$2.71E-89$
	Std	$1.13E-15$	$5.69E+03$	$2.77E-69$	$5.15E+02$	$1.53E-27$	$1.38E-88$
	Rank	4	6	2	5	3	1
f_7	Best	$1.49E-14$	$4.97E+04$	$1.44E-70$	$7.57E+03$	$6.03E-29$	$1.63E-95$
	Worst	$1.90E-14$	$3.65E+05$	$2.08E-67$	$5.13E+04$	$1.23E-25$	$1.34E-87$
	Mean	$1.74E-14$	$1.57E+05$	$2.30E-68$	$2.19E+04$	$8.07E-27$	$1.26E-88$
	Std	$1.08E-15$	$8.29E+04$	$4.33E-68$	$9.20E+03$	$2.40E-26$	$2.94E-88$
	Rank	4	6	2	5	3	1

Figures 25–31 are the convergence curves of all algorithms for solving high-dimensional multimodal functions independently run 30 times. Figures 25–30 show that IFCSA can find the optimal value faster and better for some high-dimensional multimodal functions. Furthermore, in the functions f_8 and f_{10} , IFCSA can find the theoretical optimal value. Figures 32–38 are the variance graphs of $f_8 \sim f_{14}$, which shows that IFCSA has good stability. The experimental results show that, compared with some classic intelligent algorithms and some improved algorithms, IFCSA can better solve high-dimensional multimodal functions.

4.3.3. Fixed-Dimensional Multimodal Function Test Results. Table 7 shows the experimental results of different algorithms for fixed multimodal functions. From the experimental results in Table 7, it can be seen that the best values of IFCSA in functions $f_{15}, f_{16}, f_{17}, f_{19}$, and f_{20} are all ranked first, and they are all theoretical best values. It can be also seen from the standard deviation in Table 7 that the performance of IFCSA is relatively stable. Although the IFCSA does not obtain a better optimal value than GWO in the function f_{18} , the optimal value obtained by the IFCSA is of the order of 100^{-178} , which is acceptable.

TABLE 6: Results of high-dimensional multimodal benchmark functions.

Function	Index	BOA	SSA	GWO	CSA	MISCSA	IFCSA
f_8	Best	0.00E+00	2.49E+01	0.00E+00	1.09E+01	0.00E+00	0.00E+00
	Worst	2.10E+02	7.46E+01	4.51E+00	4.08E+01	0.00E+00	0.00E+00
	Mean	1.24E+01	4.60E+01	1.50E-01	2.27E+01	0.00E+00	0.00E+00
	Std	4.76E+01	1.27E+01	8.24E-01	8.63E+00	0.00E+00	0.00E+00
	Rank	1	6	1	5	1	1
f_9	Best	6.05E-12	1.78E+00	7.99E-15	1.91E+00	8.88E-16	8.88E-16
	Worst	1.32E-11	3.63E+00	1.51E-14	4.70E+00	8.88E-16	8.88E-16
	Mean	1.12E-11	1.78E+00	1.31E-14	3.30E+00	8.88E-16	8.88E-16
	Std	1.39E-12	8.54E-01	2.59E-15	6.50E-01	0.00E+00	0.00E+00
	Rank	4	5	3	6	1	1
f_{10}	Best	0.00E+00	2.27E-08	0.00E+00	3.11E-02	0.00E+00	0.00E+00
	Worst	3.66E-15	4.91E-02	1.35E-02	2.58E-01	0.00E+00	0.00E+00
	Mean	9.25E-16	1.01E-02	1.10E-03	1.05E-01	0.00E+00	0.00E+00
	Std	1.05E-15	1.17E-02	3.43E-03	4.18E-02	0.00E+00	0.00E+00
	Rank	1	6	1	5	1	1
f_{11}	Best	1.52E-15	5.72E-02	1.57E-41	6.60E-03	4.12E-18	1.88E-52
	Worst	1.24E-12	4.92E+00	5.50E-04	5.17E-01	5.08E-17	1.08E-46
	Mean	4.70E-14	1.73E+00	2.21E-05	1.04E-01	1.91E-16	7.20E-48
	Std	2.25E-13	1.15E+00	1.00E-04	1.45E-01	3.83E-17	2.24E-47
	Rank	4	5	2	6	3	1
f_{12}	Best	9.11E-15	9.43E-02	1.45E-07	2.82E-01	2.49E-34	1.79E-103
	Worst	1.65E-14	1.17E+00	1.17E-05	1.94E+00	1.21E-30	5.87E-92
	Mean	1.39E-14	4.51E-01	2.80E-06	6.99E-01	1.14E-31	2.86E-93
	Std	1.70E-15	2.98E-01	2.77E-06	3.74E-01	2.24E-31	1.12E-92
	Rank	3	5	4	6	2	1
f_{13}	Best	1.51E+00	2.69E-01	7.22E-01	2.70E-01	6.31E-01	1.37E-04
	Worst	2.52E+00	1.27E+01	1.27E+00	3.21E+00	2.34E+00	8.21E-01
	Mean	2.00E+00	6.61E+00	9.89E-01	1.06E+00	1.47E+00	5.99E-02
	Std	2.47E-01	3.05E+00	1.52E-01	7.62E-01	3.94E-01	1.68E-01
	Rank	6	2	5	3	4	1
f_{14}	Best	1.55E+00	2.91E-10	1.43E-05	4.90E+00	1.40E+00	1.60E-03
	Worst	3.00E+00	1.10E-01	6.23E-01	3.95E+01	4.22E+00	2.90E+00
	Mean	2.57E+00	7.93E-03	3.16E-01	2.03E+01	2.92E+00	7.19E-01
	Std	3.50E-01	2.07E-02	1.52E-01	9.93E+00	7.59E-01	6.88E-01
	Rank	5	1	2	6	4	3

Figures 39–44 are the convergence curves of the 6 algorithms for different fixed multimodal functions, and Figures 45–50 are the variance graphs of the 6 algorithms. From Figures 39, 41, 43, and 44, it can be seen that IFCSA can converge to the global optimum faster than the other five algorithms. Figure 40 shows that the optimization performance of IFCSA is weaker than MISCSA, while the overall performance of IFCSA is better than that of the other algorithms. It can be seen from Figure 42 that the optimization performance of IFCSA is better than that of the other algorithms except GWO. In short, IFCSA has certain advantages in optimizing fixed multimodal functions.

4.4. Statistical Validation. According to the paper by Derrac et al. [25], it is not rigorous to only use the average value, standard deviation, optimal value, and worst value obtained after the algorithm is independently run 30 times as the evaluation index of the algorithm performance. Therefore, Wilcoxon rank sum test is carried out on the 20 benchmark test functions in Tables 1~3 for the six algorithms in this paper. Moreover, the p values obtained after rank sum check

of the six algorithms are recorded in Table 9. If the algorithm with the best performance is IFCSA, a pair comparison is performed between IFCSA and BOA, IFCSA and SSA, etc. Since the best algorithm cannot be compared with itself, it is marked as “NA,” indicating that it is not applicable, which also means that the corresponding algorithm does not have corresponding data to be compared with itself in the rank sum verification process. When the p value is less than 0.05, there is a big difference between the two comparison algorithms. Otherwise, it indicates that there is some similarity between the two comparison algorithms, and the value of p is marked in bold type.

4.5. Engineering Optimization Problem. As can be seen from the previous section, IFCSA has better performance in function optimization than other intelligent algorithms and some improved algorithms. In order to further verify whether IFCSA is effective in practical engineering applications, it is applied to the speed reducer design problem. The problem is one of the most fully researched problems in the optimization test. It represents a simple gearbox design,

TABLE 7: Results of fixed-dimensional multimodal benchmark functions.

Function	Index	BOA	SSA	GWO	CSA	MISCSA	IFCSA
f_{15}	Best	$3.33E-16$	$8.66E-15$	$0.00E+00$	$0.00E+00$	$0.00E+00$	$0.00E+00$
	Worst	$1.02E-02$	$9.72E-03$	$9.72E-03$	$9.72E-03$	$0.00E+00$	$0.00E+00$
	Mean	$8.15E-03$	$1.94E-03$	$2.27E-03$	$7.00E-04$	$0.00E+00$	$0.00E+00$
	Std	$3.71E-03$	$3.95E-03$	$4.18E-03$	$2.46E-03$	$0.00E+00$	$0.00E+00$
	Rank	5	6	1	1	1	1
f_{16}	Best	$1.55E-05$	$4.57E-18$	$1.29E-09$	$3.68E-27$	$0.00E+00$	$0.00E+00$
	Worst	$3.98E-01$	$5.52E-15$	$2.37E-07$	$7.56E-24$	$1.40E-28$	$0.00E+00$
	Mean	$8.38E-02$	$1.03E-15$	$3.28E-08$	$7.73E-25$	$5.56E-30$	$0.00E+00$
	Std	$1.26E-01$	$1.34E-15$	$4.51E-08$	$1.54E-24$	$2.56E-29$	$0.00E+00$
	Rank	6	4	5	3	1	1
f_{17}	Best	0.00031	0.00031	0.00030	0.00030	0.00031	0.00030
	Worst	0.00038	0.00124	0.02036	0.00122	0.00133	0.00122
	Mean	0.00033	0.00081	0.00381	0.00034	0.00043	0.00037
	Std	0.00002	0.00026	0.00754	0.00017	0.00020	0.00023
	Rank	4	4	1	1	4	1
f_{18}	Best	$2.29E-19$	$7.08E-18$	$0.00E+00$	$7.24E-28$	$1.22E-43$	$2.41E-177$
	Worst	$2.61E-17$	$2.86E-15$	$0.00E+00$	$1.04E-24$	$7.79E-41$	$4.06E-166$
	Mean	$6.01E-18$	$7.11E-16$	$0.00E+00$	$1.88E-25$	$1.19E-41$	$1.45E-167$
	Std	$7.05E-18$	$7.43E-16$	$0.00E+00$	$2.45E-25$	$1.70E-41$	$0.00E+00$
	Rank	5	6	1	4	3	2
f_{19}	Best	0.398	0.398	0.398	0.398	0.398	0.398
	Worst	1.212	0.398	0.398	0.398	0.398	0.398
	Mean	0.441	0.398	0.398	0.398	0.398	0.398
	Std	$1.532E-01$	$2.410E-15$	$1.252E-07$	$0.000E+00$	$0.000E+00$	$0.000E+00$
	Rank	1	1	1	1	1	1
f_{20}	Best	3.0002	3.0000	3.0000	3.0000	3.0000	3.0000
	Worst	3.2338	3.0000	3.0000	3.0000	3.0000	3.0000
	Mean	3.0297	3.0000	3.0000	3.0000	3.0000	3.0000
	Std	$5.10E-02$	$7.55E-14$	$3.91E-06$	$1.47E-15$	$1.65E-15$	$1.37E-15$
	Rank	6	1	1	1	1	1

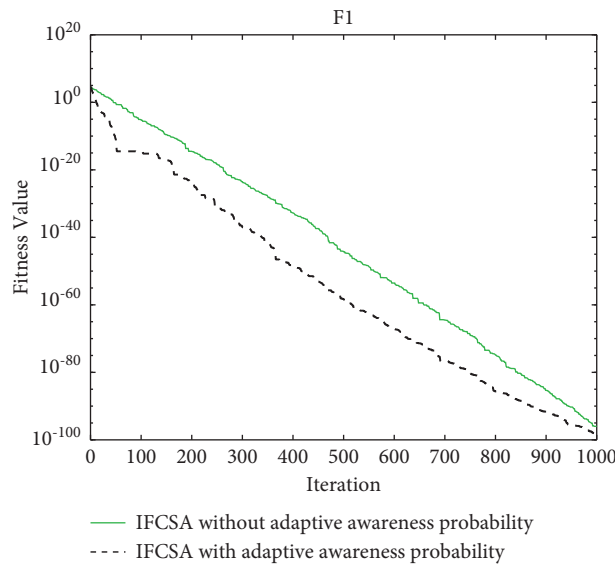


FIGURE 5: Comparison chart of f_1 convergence curve.

which can be used between the aircraft engine and the propeller to make the components rotate at the most effective speed.

The goal of speed reducer design is to minimize the weight of the gear under bending stress, the lateral deflection of the shaft, and the constraint of the shaft. As shown in

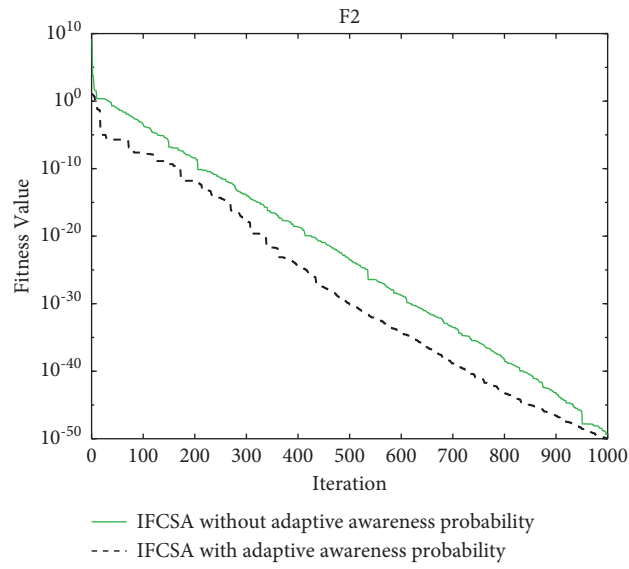


FIGURE 6: Comparison chart of f_2 convergence curve.

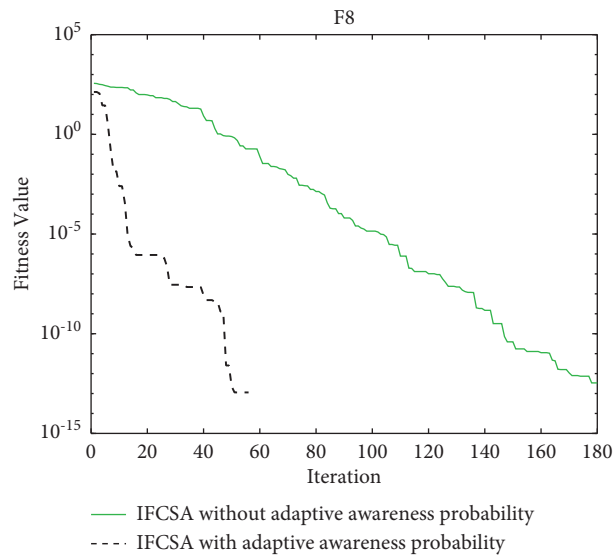


FIGURE 7: Comparison chart of f_8 convergence curve.

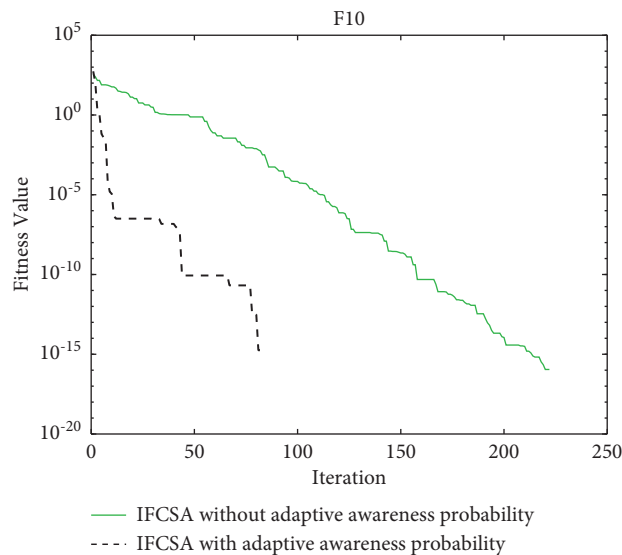


FIGURE 8: Comparison chart of f_{10} convergence curve.

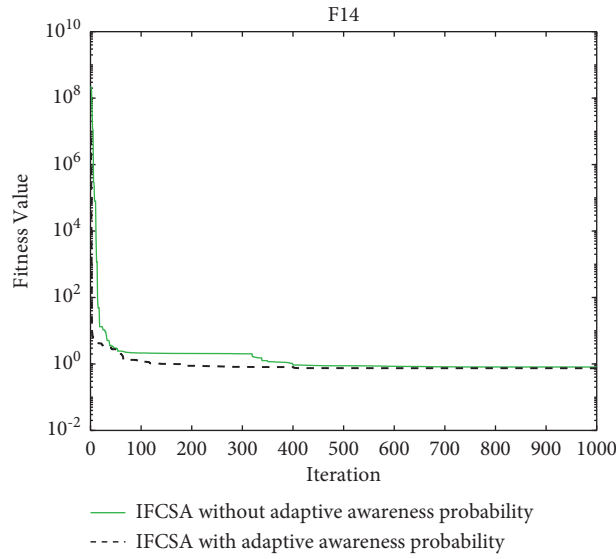


FIGURE 9: Comparison chart of f_{14} convergence curve.

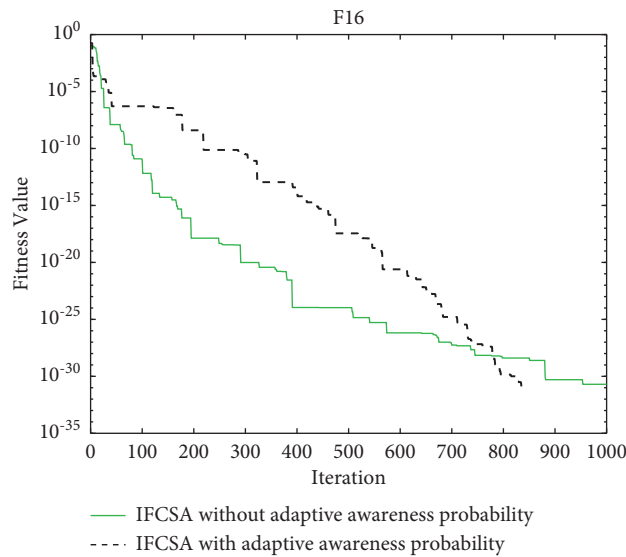


FIGURE 10: Comparison chart of f_{16} convergence curve.

TABLE 8: Results of experiments with different strategies.

Function	CSA	CSA + FPA (without tent chaos to the initial position)	CSA + FPA + Cauchy mutation (without tent chaos to the initial position)
f_1	$1.43E - 02$	$1.01E + 03$	$2.90E - 94$
f_2	$1.40E + 00$	$9.04E + 00$	$5.31E - 48$
f_3	$2.62E + 00$	$1.37E + 01$	$1.04E - 47$
f_4	$8.61E + 02$	$3.18E + 03$	$9.65E - 95$
f_5	$2.17E - 02$	$1.35E - 01$	$3.43E - 04$
f_6	$1.47E + 03$	$2.41E + 03$	$2.30E - 92$
f_7	$2.54E + 04$	$3.03E + 05$	$1.39E - 90$
f_8	$2.18E + 01$	$7.79E + 01$	$0.00E + 00$
f_9	$3.29E + 00$	$8.99E + 00$	$8.88E - 16$
f_{10}	$9.90E - 02$	$9.84E + 00$	$0.00E + 00$
f_{11}	$2.15E - 01$	$4.60E + 00$	$1.37E - 49$
f_{12}	$6.01E - 01$	$4.55E + 01$	$1.21E - 95$

TABLE 8: Continued.

Function	CSA	CSA + FPA (without tent chaos to the initial position)	CSA + FPA + Cauchy mutation (without tent chaos to the initial position)
f_{13}	$7.77E - 01$	$3.21E + 00$	$1.26E - 01$
f_{14}	$2.04E + 01$	$1.14E + 04$	$1.30E + 00$
f_{15}	$9.72E - 04$	$3.89E - 03$	$0.00E + 00$
f_{16}	$4.01E - 25$	$0.00E + 00$	$1.52E - 30$
f_{17}	0.00037	0.00034	0.00030
f_{18}	$2.35E - 25$	$2.78E - 163$	$2.90E - 182$
f_{19}	0.398	0.398	0.398
f_{20}	$3.00E + 00$	$3.00E + 00$	$3.00E + 00$

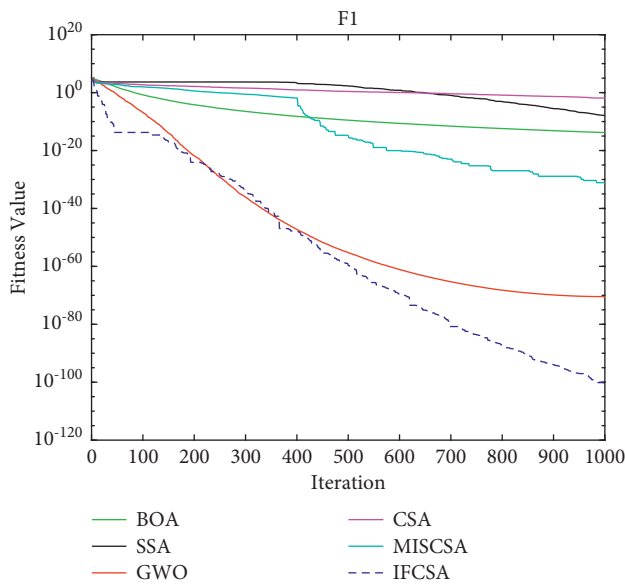


FIGURE 11: Convergence curve of f_1 .

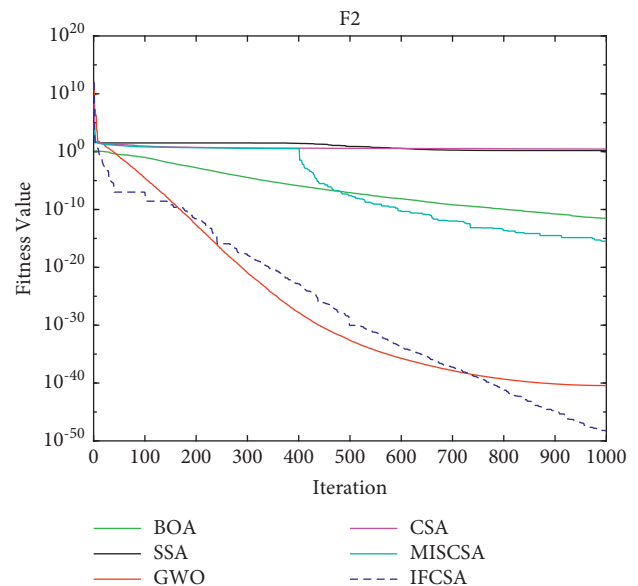


FIGURE 13: Convergence curve of f_2 .

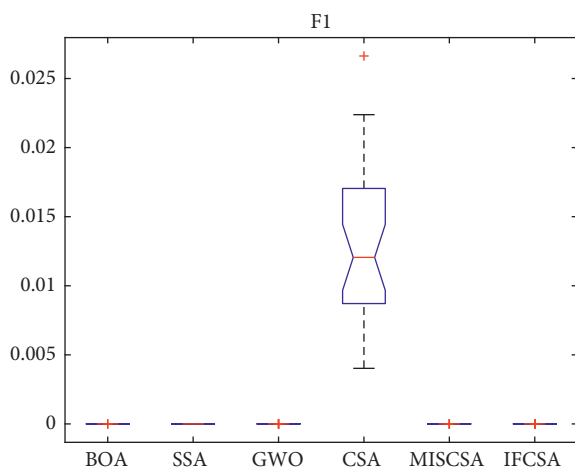


FIGURE 12: Variance diagram of f_1 .

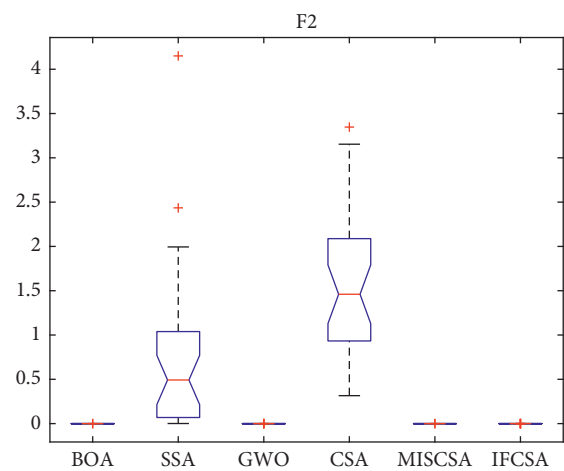


FIGURE 14: Variance diagram of f_2 .

Figure 51, the optimization problem includes seven decision variables, namely, surface width (b or x_1), module of teeth (m or x_2), number of teeth on pinion (z or x_3), length of shaft 1

between bearings (l_1 or x_4), length of shaft 2 between bearings (l_2 or x_5), diameter of shaft 1 (d_1 or x_6), and diameter of shaft 2 (d_2 or x_7).

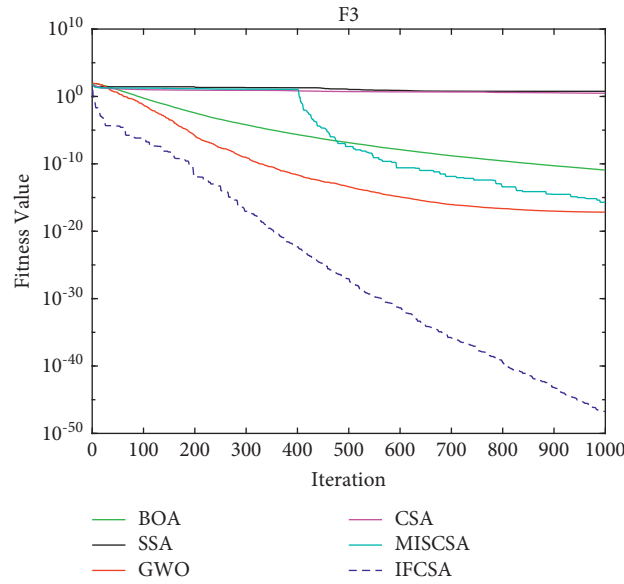


FIGURE 15: Convergence curve of f_3 .

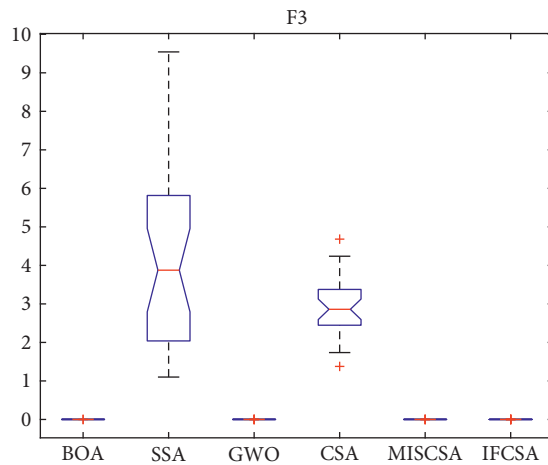


FIGURE 16: Variance diagram of f_3 .

The mathematical model of the problem is as follows:

$$\text{Min. } f(x) = 0.7854x_1x_2^2(3.3333x_3^2 + 14.9334x_3 - 43.0934) - 1.508x_1(x_6^2 + x_7^2) + 7.477(x_6^3 + x_7^3) + 0.7854(x_4x_6^2 + x_5x_7^2),$$

$$\text{S.t. } g_1(x) = \frac{27}{x_1x_2^2x_3} - 1 \leq 0,$$

$$g_2(x) = \frac{397.5}{x_1x_2^2x_3^2} - 1 \leq 0,$$

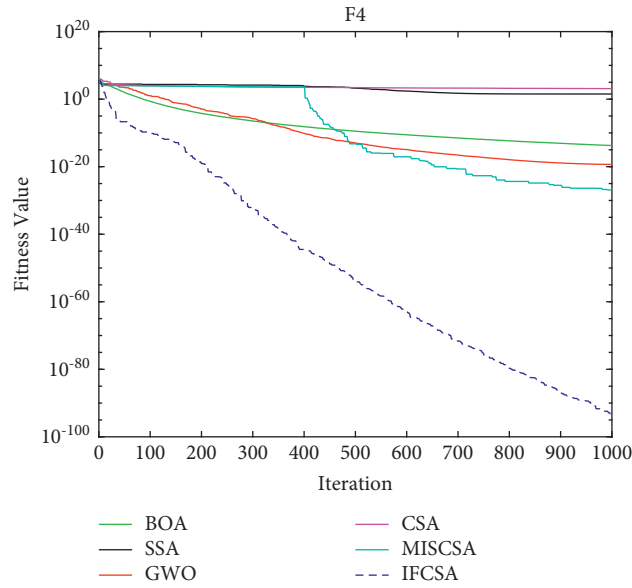


FIGURE 17: Convergence curve of f_4 .

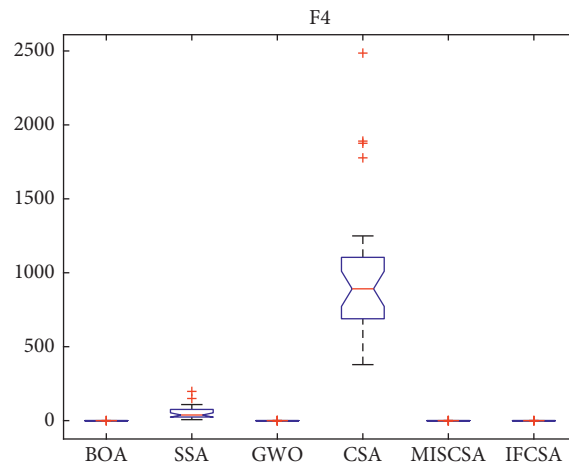


FIGURE 18: Variance diagram of f_4 .

$$g_3(x) = \frac{1.93x_4^3}{x_1x_6^4x_3} - 1 \leq 0,$$

$$g_4(x) = \frac{1.93x_5^3}{x_1x_7^4x_3} - 1 \leq 0,$$

$$g_5(x) = \frac{\sqrt{(745x_4/x_2x_3)^2 + 1.69 \times 10^6}}{110x_6^3} - 1 \leq 0,$$

$$g_6(x) = \frac{\sqrt{(745x_5/x_2x_3)^2 + 157.5 \times 10^6}}{85x_7^3} - 1 \leq 0,$$

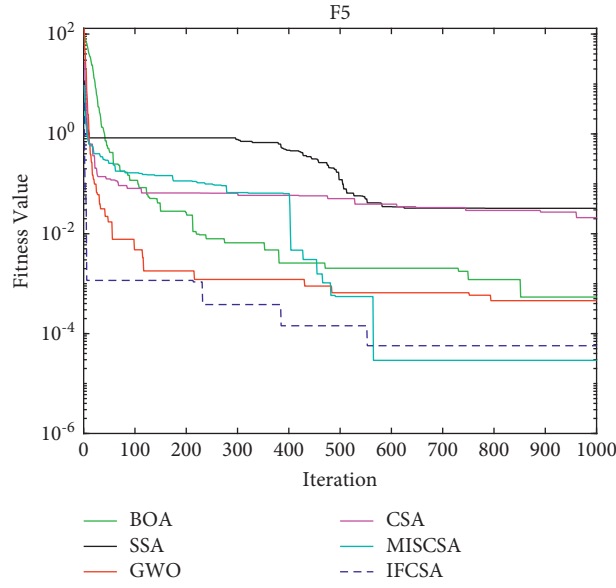


FIGURE 19: Convergence curve of f_5 .

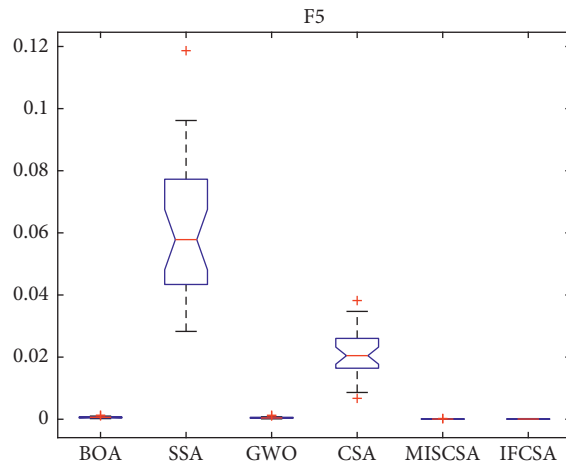


FIGURE 20: Variance diagram of f_5 .

$$g_7(x) = \frac{x_2 x_3}{40} - 1 \leq 0,$$

$$g_8(x) = \frac{5x_2}{x_1} - 1 \leq 0,$$

$$g_9(x) = \frac{x_1}{12x_2} - 1 \leq 0,$$

$$g_{10}(x) = \frac{1.5x_6 + 1.9}{x_4} - 1 \leq 0,$$

$$g_{11}(x) = \frac{1.1x_7 + 1.9}{x_5} - 1 \leq 0,$$

$$2.6 \leq x_1 \leq 3.6, 0.7 \leq x_2 \leq 0.8, 17 \leq x_3 \leq 28, 7.3 \leq x_4 \leq 8.3, 7.8 \leq x_5 \leq 8.3, 2.9 \leq x_6 \leq 3.9, 5 \leq x_7 \leq 5.5.$$

(10)

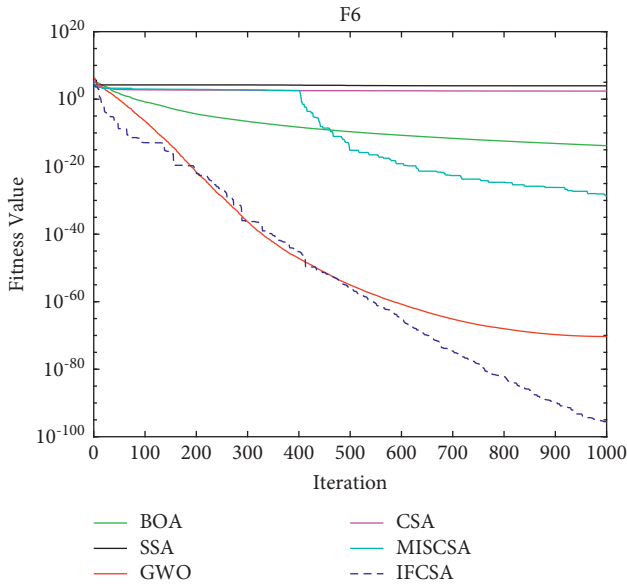


FIGURE 21: Convergence curve of f_6 .

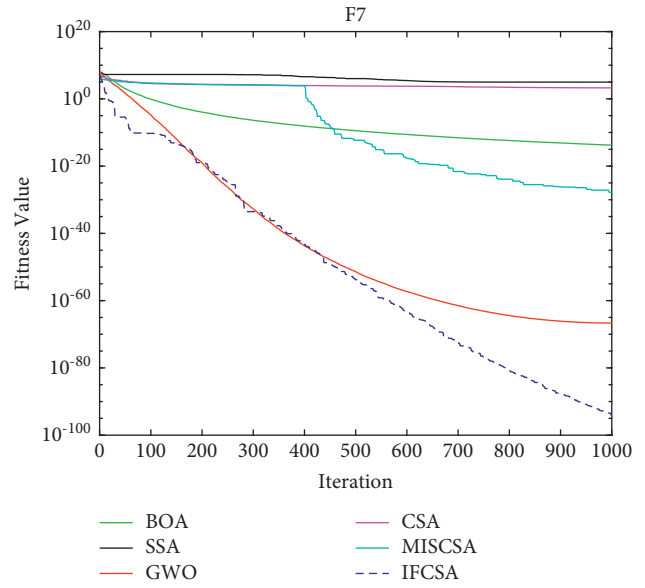


FIGURE 23: Convergence curve of f_7 .

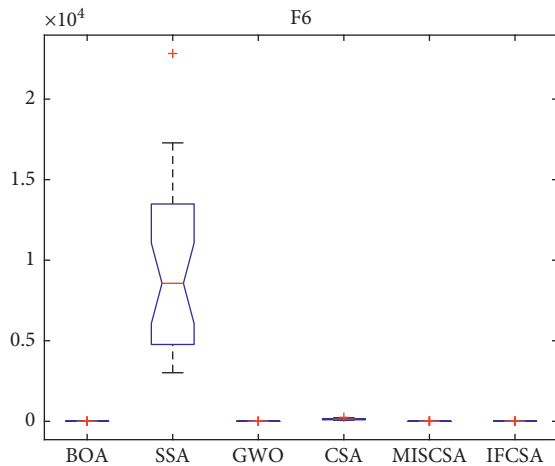


FIGURE 22: Variance diagram of f_6 .

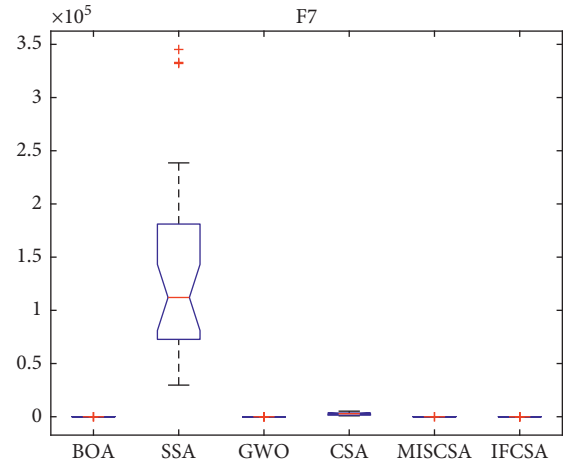


FIGURE 24: Variance diagram of f_7 .

Table 10 shows the optimal values and values of decision variables obtained after 30 independent runs of different algorithms for deceleration design problems. The results are compared with those of CSO [26], Gandomi et al. [27], ABC [28], Akhtar et al. [29], and Montes et al.

[30]. According to Table 10, IFCSA finds the optimal cost of 2896.26, which is the least expensive among the comparison algorithms. The solution to find the optimal value is as follows: $x_1 = 3.5, x_2 = 0.7, x_3 = 17, x_4 = 7.3, x_5 = 7.8, x_6 = 2.9, x_7 = 5.286683$. The results show that IFCSA has good performance in dealing with the optimization of speed reducer design problem.

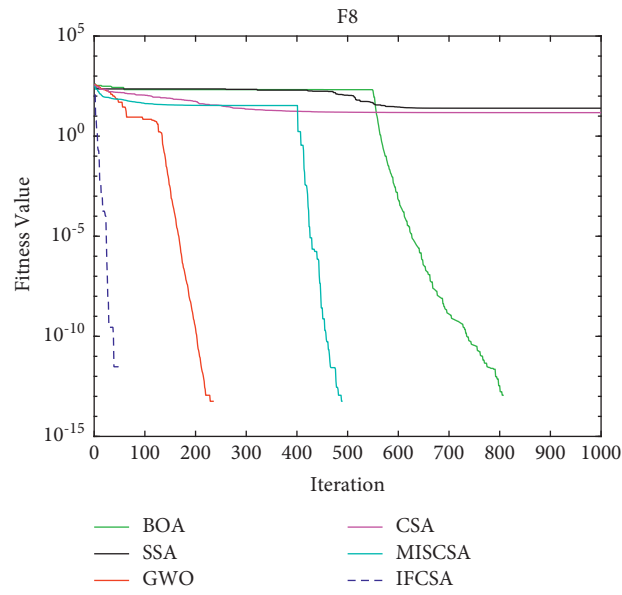


FIGURE 25: Convergence curve of f_8 .

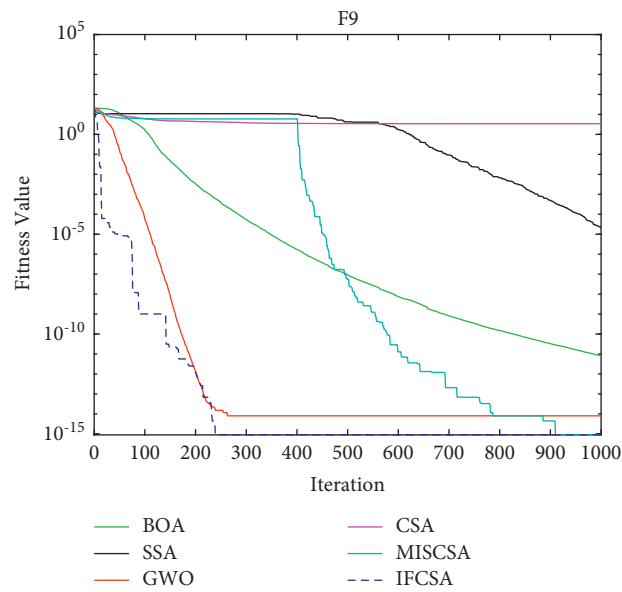


FIGURE 26: Convergence curve of f_9 .

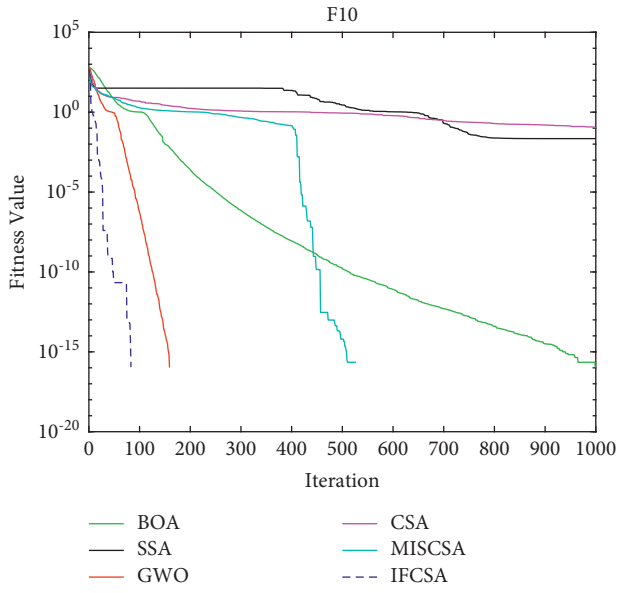


FIGURE 27: Convergence curve of f_{10} .

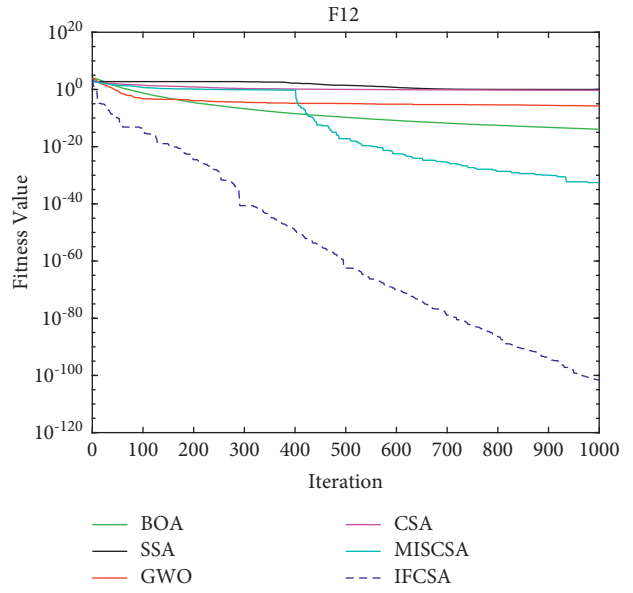


FIGURE 29: Convergence curve of f_{12} .

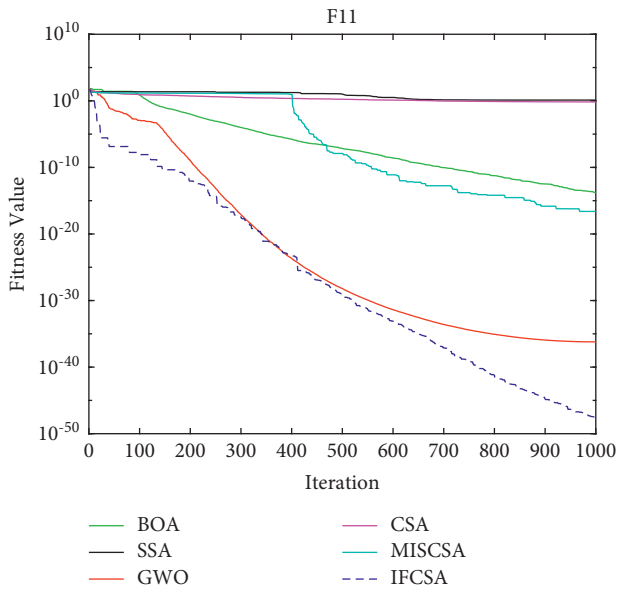


FIGURE 28: Convergence curve of f_{11} .

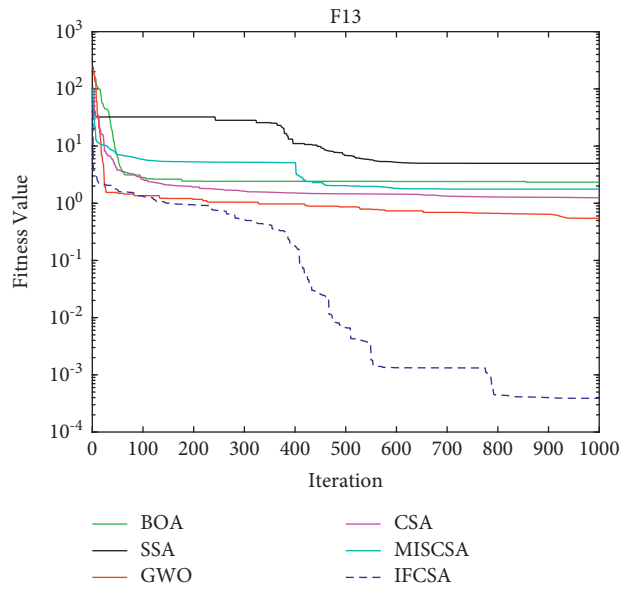


FIGURE 30: Convergence curve of f_{13} .

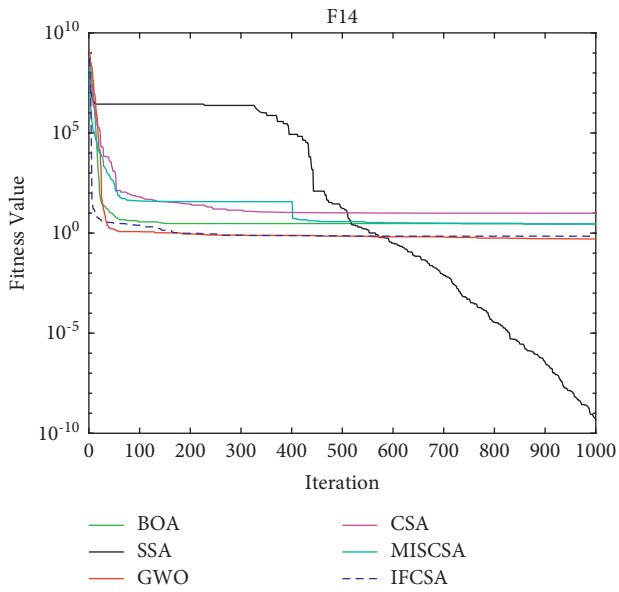


FIGURE 31: Convergence curve of f_{14} .

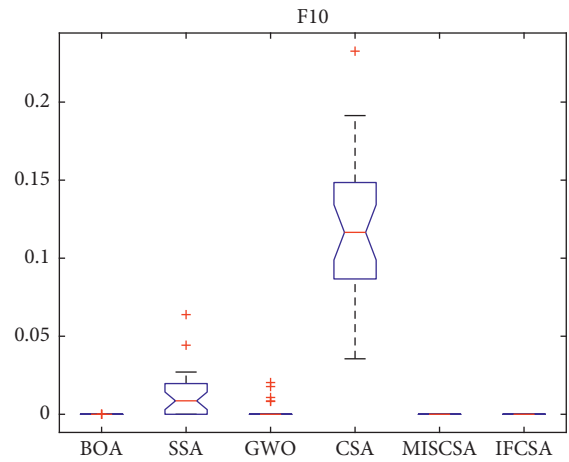


FIGURE 34: Variance diagram of f_{10} .

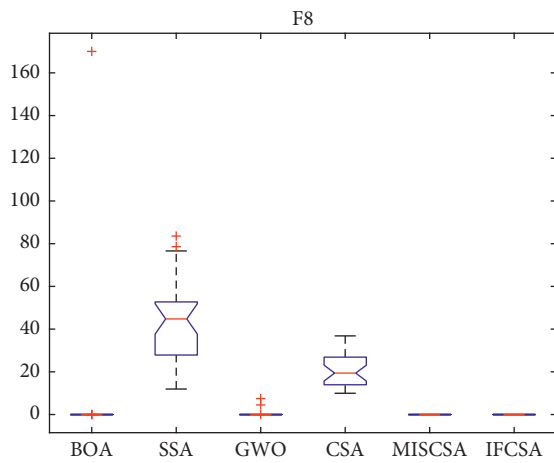


FIGURE 32: Variance diagram of f_8 .

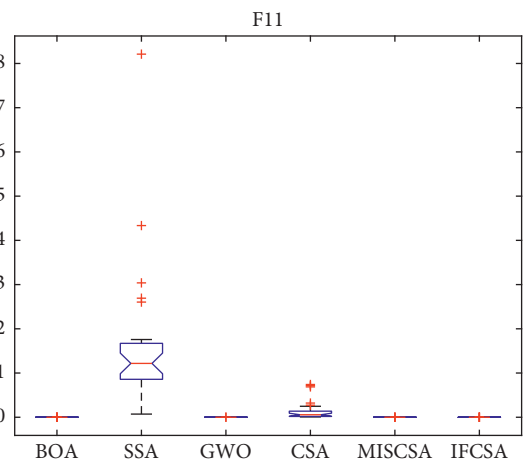


FIGURE 35: Variance diagram of f_{11} .

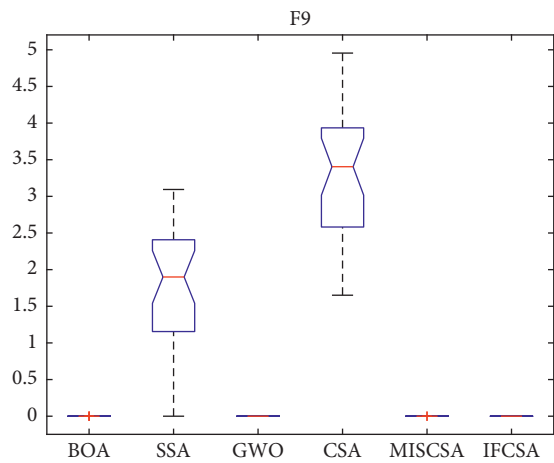


FIGURE 33: Variance diagram of f_9 .

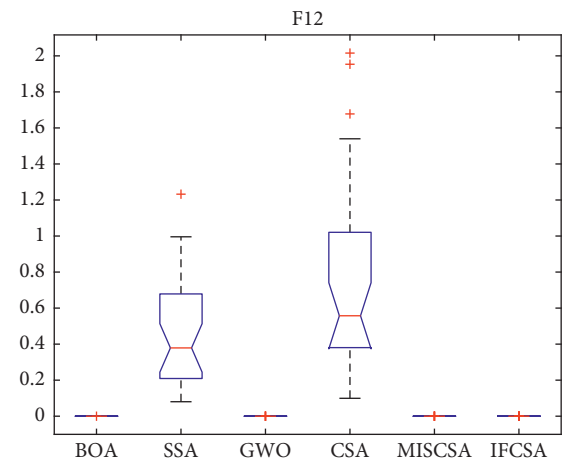


FIGURE 36: Variance diagram of f_{12} .

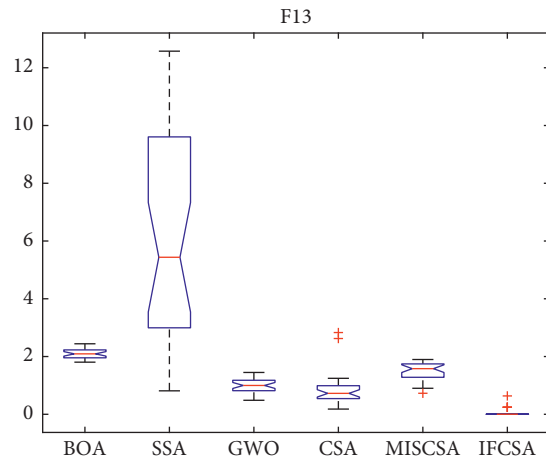


FIGURE 37: Variance diagram of f_{13} .

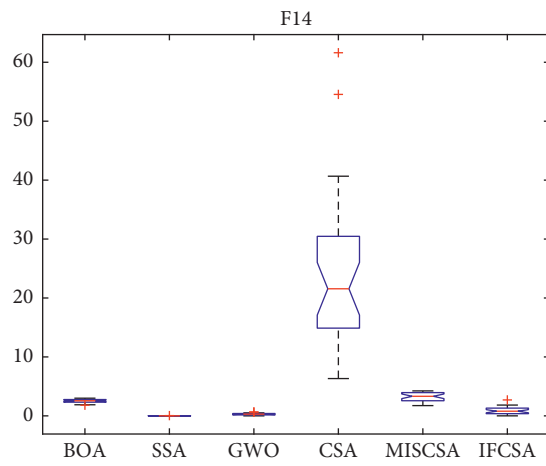


FIGURE 38: Variance diagram of f_{14} .

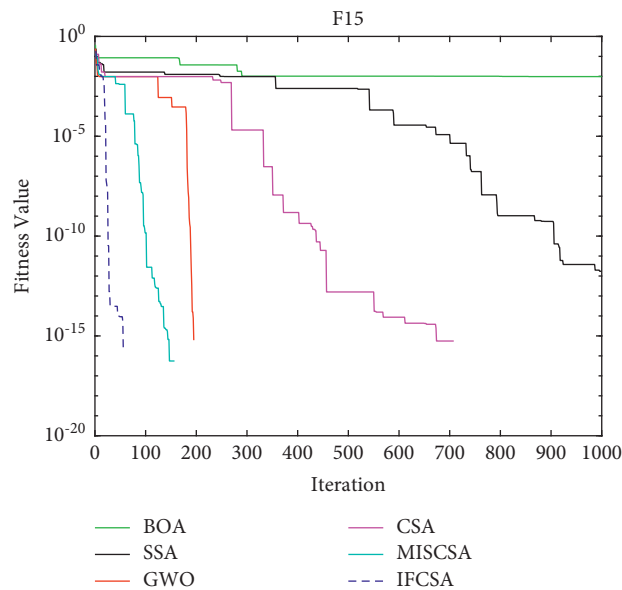


FIGURE 39: Convergence curve of f_{15} .

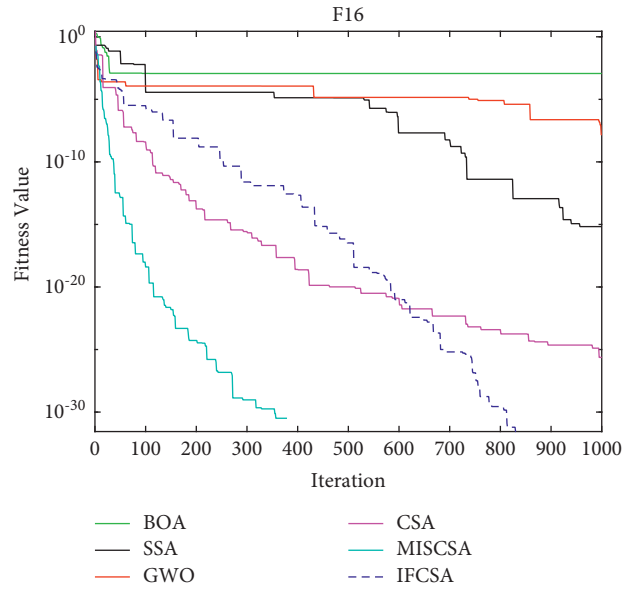


FIGURE 40: Convergence curve of f_{16} .

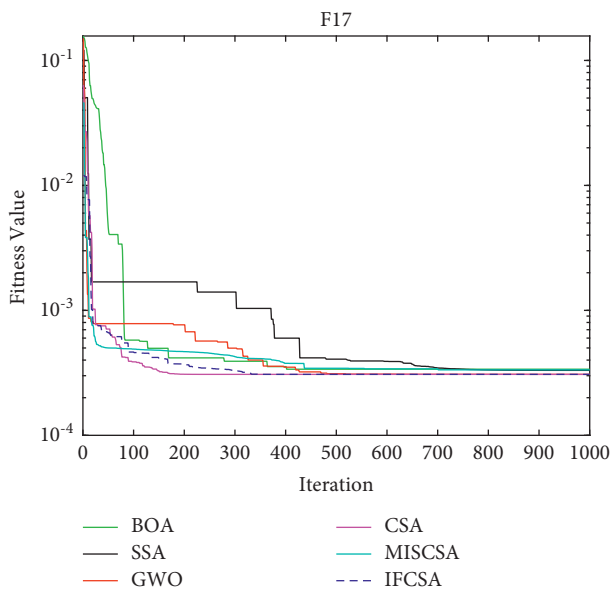


FIGURE 41: Convergence curve of f_{17} .

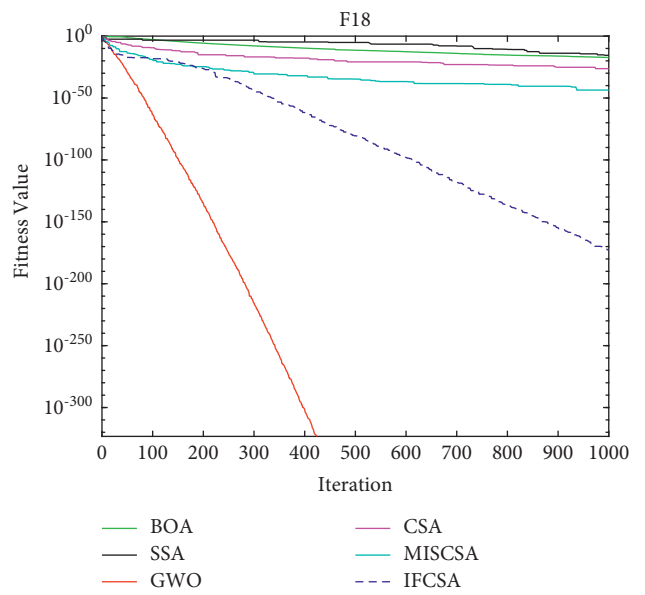


FIGURE 42: Convergence curve of f_{18} .

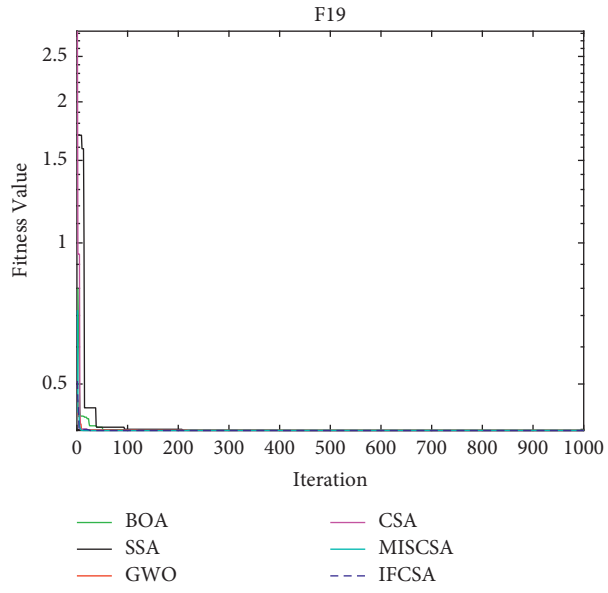


FIGURE 43: Convergence curve of f_{19} .

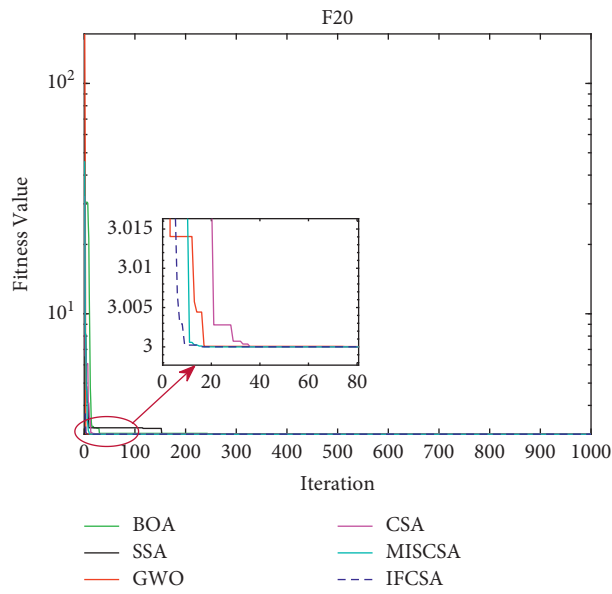


FIGURE 44: Convergence curve of f_{20} .

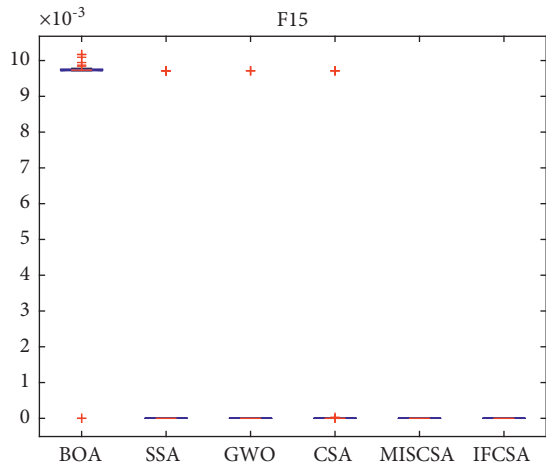


FIGURE 45: Variance diagram of f_{15} .

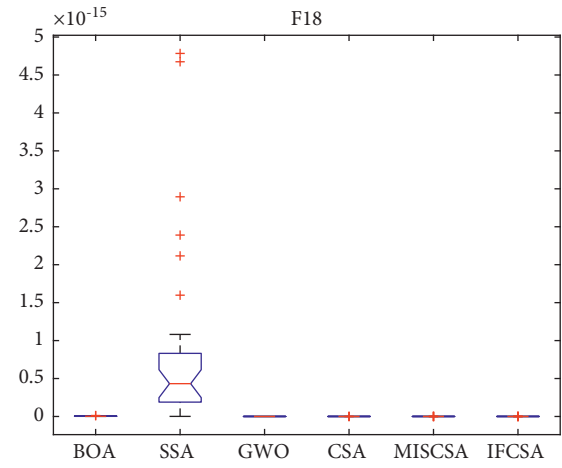


FIGURE 48: Convergence curve of f_{18} .

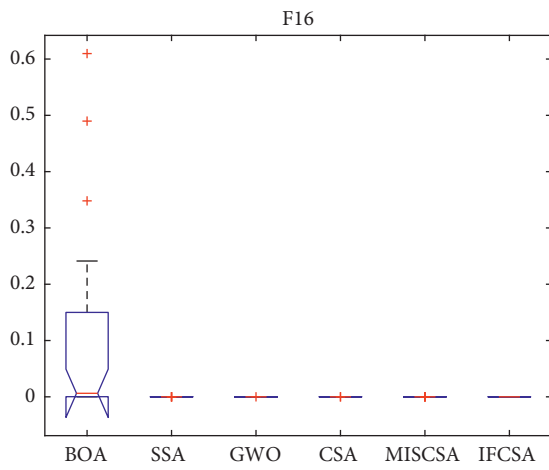


FIGURE 46: Convergence curve of f_{16} .

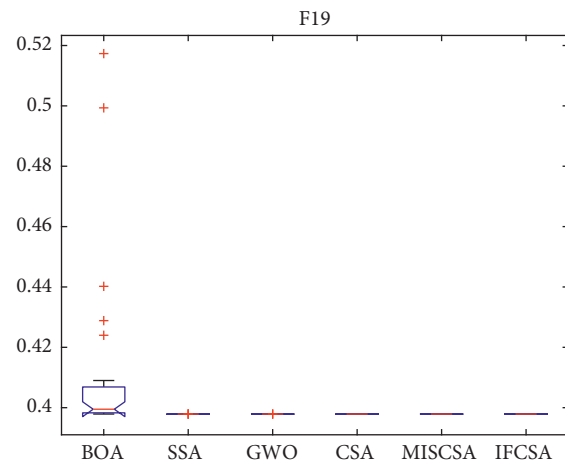


FIGURE 49: Convergence curve of f_{19} .

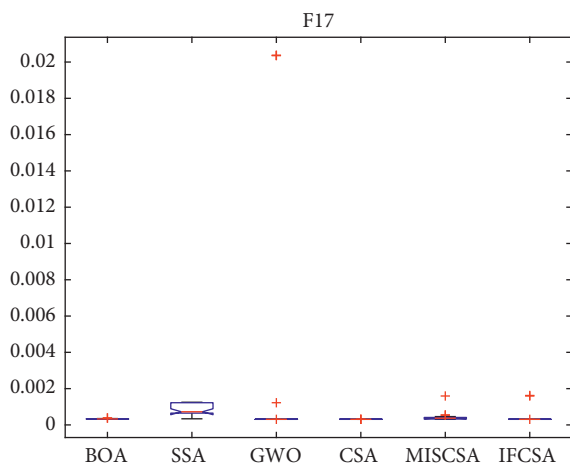


FIGURE 47: Convergence curve of f_{17} .

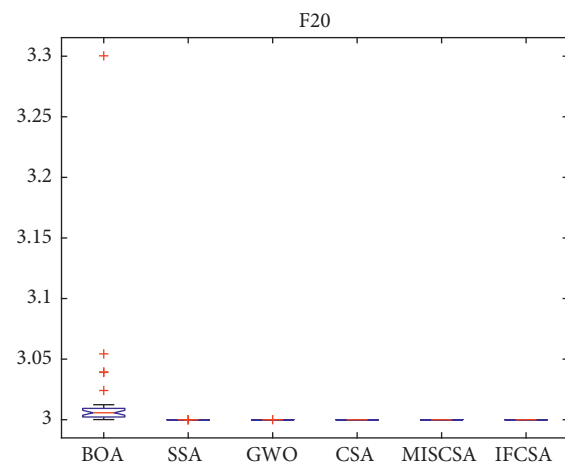


FIGURE 50: Convergence curve of f_{20} .

TABLE 9: Wilcoxon rank sum test and p value.

Function	IFCSA/BOA	IFCSA/SSA	IFCSA/GWO	IFCSA/CSA	IFCSA/MISCFA
f_1	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$
f_2	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$
f_3	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$
f_4	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$
f_5	$1.33E-10$	$3.02E-11$	$3.82E-10$	$3.02E-11$	$5.57E-03$
f_6	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$
f_7	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$
f_8	$2.16E-02$	$1.21E-12$	$8.15E-02$	$1.21E-12$	NA
f_9	$1.21E-12$	$1.21E-12$	$5.47E-13$	$1.21E-12$	NA
f_{10}	$1.93E-09$	$1.21E-12$	$8.15E-02$	$1.21E-12$	NA
f_{11}	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$
f_{12}	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$	$3.02E-11$
f_{13}	$3.02E-11$	$3.69E-11$	$6.07E-11$	$1.61E-10$	$1.61E-10$
f_{14}	$5.57E-10$	$2.44E-09$	$7.62E-03$	$3.02E-11$	$2.61E-10$
f_{15}	$1.21E-12$	$1.21E-12$	$5.58E-03$	$1.37E-03$	NA
f_{16}	$1.21E-12$	$1.21E-12$	$1.21E-12$	$1.21E-12$	$5.83E-09$
f_{17}	$8.15E-09$	$2.34E-09$	$1.35E-09$	$6.76E-09$	$6.84E-09$
f_{18}	$3.02E-11$	$3.02E-11$	$1.21E-12$	$1.21E-12$	$1.21E-12$
f_{19}	$1.21E-12$	$3.09E-04$	$1.21E-12$	NA	NA
f_{20}	$1.10E-11$	$1.10E-11$	$1.10E-11$	$1.35E-06$	$3.55E-06$

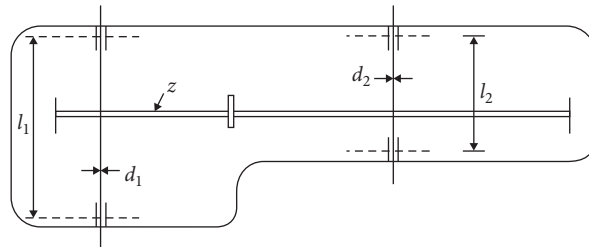


FIGURE 51: Schematic diagram of speed reducer design.

TABLE 10: Test results of different algorithms for speed reducer design.

	IFCSA	CSO [26]	Gandomi et al. [27]	ABC [28]	Akhtar et al. [29]	Montes et al. [30]
Best	2896.26	2996.60	3000.98	2997.06	3008.08	3025.01
x_1	3.500000	3.500000	3.501500	3.499999	3.506122	3.506163
x_2	0.700000	0.700000	0.700000	0.700000	0.700006	0.700831
x_3	17.00000	17.00000	17.00000	17.00000	17.00000	17.00000
x_4	7.300000	7.308000	7.605000	7.300000	7.549126	7.460181
x_5	7.800000	7.802000	7.818100	7.800000	7.859330	7.962143
x_6	2.900000	3.350000	3.352000	3.350215	3.365576	3.362900
x_7	5.286683	5.287000	5.287500	5.287800	5.289773	5.309000

5. Conclusions

By studying the principle and updating the formula of the standard crow search algorithm, IFCSA is proposed to solve the problem that the algorithm slowly converges and easily falls into local optimum in the later iteration. In this paper, so as to improve the convergence ability of the algorithm, the inverse incomplete gamma function is introduced to make the perceptual probability decrease nonlinearly. Aiming at the blindness of crows' random search for location updating, a cross-pollination strategy with Cauchy mutation was introduced to make crows tend to take the best individual direction, thus obtaining the best value. The experimental results in this paper also show

that the optimization performance of IFCSA is better than that of the original algorithm and other intelligent algorithms.

In future work, IFCSA will be used to solve more complex optimization problems, such as multiresource constrained project sequencing, image processing, and UAV path planning. IFCSA will be also used to solve more engineering examples, which is to provide reference value for engineering applications.

Data Availability

The data, models, or code generated or used during the study are available at <https://github.com/happyfate/IFCSA>.

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 (no. 61662005); Guangxi Natural Science Foundation (no. 2018GXNSFAA294068); Basic Ability Improvement Project for Young and Middle-Aged Teachers in Colleges and Universities in Guangxi (no. 2019KY0195); and Research Project of Guangxi University for Nationalities (no. 2019KJYB006).

References

- [1] Y. Tadepalli, M. Kollati, S. Kuraparthi, P. Kora, A. K. Budati, and L. Kala Pampana, "Content-based image retrieval using Gaussian-hermite moments and firefly and grey wolf optimization," *CAAI Transactions on Intelligence Technology*, vol. 6, no. 2, pp. 135–146, 2021.
- [2] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference On Neural Networks 1995*, vol. 4, pp. 1942–1948, Perth, WA, Australia, December 2002.
- [3] X. S. Yang and X. He, "Bat algorithm: literature review and applications," *International Journal of Bio-Inspired Computation*, vol. 5, no. 3, pp. 141–149, 2013.
- [4] S. Arora and S. Singh, "Butterfly optimization algorithm: a novel approach for global optimization," *Soft Computing - A Fusion of Foundations, Methodologies and Applications Archive*, vol. 23, no. 3, pp. 715–734, 2019.
- [5] X.-S. Yang, "Flower pollination algorithm for global optimization," in *Proceedings of the 11th International Conference on Unconventional Computation and Natural Computation UCNC'12*, pp. 240–249, Orléans, France, September 2012.
- [6] H. Duan and P. Qiao, "Pigeon-inspired optimization: a new swarm intelligence optimizer for air robot path planning," *International Journal of Intelligent Computing and Cybernetics*, vol. 7, no. 1, pp. 24–37, 2014.
- [7] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in Engineering Software*, vol. 95, no. 95, pp. 51–67, 2016.
- [8] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.
- [9] R. V. Rao and V. Patel, "An elitist teaching-learning-based optimization algorithm for solving complex constrained optimization problems," *International Journal of Industrial Engineering Computations*, vol. 3, no. 4, pp. 535–560, 2012.
- [10] A. Askarzadeh, "A novel metaheuristic method for solving constrained engineering optimization problems: crow search algorithm," *Computers & Structures*, vol. 169, pp. 1–12, 2016.
- [11] G. Y. Abdallah and Z. Y. Algarni, "A QSAR classification model of skin sensitization potential based on improving binary crow search algorithm," *Electronic Journal of Applied Statistical Analysis*, vol. 13, no. 1, pp. 86–95, 2020.
- [12] D. Oliva, S. Hinojosa, E. Cuevas, G. Pajares, O. Avalos, and J. Gálvez, "Cross entropy based thresholding for magnetic resonance brain images using crow search algorithm," *Expert Systems with Applications*, vol. 79, pp. 164–180, 2017.
- [13] R. C. T. D. Souza, D. S. C. Leandro, A. D. M. Camila, and P. Juliano, "A V-shaped binary crow search algorithm for feature selection," in *Proceedings of the 2018 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8, Rio de Janeiro, Brazil, July 2018.
- [14] R. S. Chithra and P. Jagatheeswari, "Fractional crow search-based support vector neural network for patient classification and severity analysis of tuberculosis," *IET Image Processing*, vol. 13, no. 1, pp. 108–117, 2019.
- [15] N. Siswanto, A. N. Adianto, H. A. Prawira, and A. Rusdiansyah, "A crow search algorithm for aircraft maintenance check problem and continuous airworthiness maintenance program," *Jurnal Sistem Dan Manajemen Industri*, vol. 3, no. 2, pp. 115–123, 2019.
- [16] P. K. Kodoth and G. Edachana, "An energy efficient data gathering scheme for wireless sensor networks using hybrid crow search algorithm," *IET Communications*, vol. 15, no. 7, pp. 906–916, 2021.
- [17] H. Wu, P. Wu, K. Xu, and F. Li, "Finite element model updating using crow search algorithm with Levy flight," *International Journal for Numerical Methods in Engineering*, vol. 121, no. 13, pp. 2916–2928, 2020.
- [18] X. Liu, Y. He, C. Wu, and L. Li, "Chaotic binary crow algorithm for 0-1 knapsack problem," *Computer Engineering and Applications*, vol. 54, no. 10, pp. 173–179, 2018.
- [19] F. Mohammadi and H. Abdi, "A modified crow search algorithm (MCSA) for solving economic load dispatch problem," *Applied Soft Computing*, vol. 71, pp. 51–65, 2018.
- [20] Z. Xiao, S. Liu, F. Han, and J. Yu, "Crow search algorithm based on directing of sine cosine algorithm," *Computer Engineering and Applications*, vol. 55, no. 21, pp. 52–59, 2019.
- [21] S. Arora, H. Singh, M. Sharma, S. Sharma, and P. Anand, "A new hybrid algorithm based on grey Wolf optimization and crow search algorithm for unconstrained function optimization and feature selection," *IEEE Access*, vol. 7, pp. 26343–26361, 2019.
- [22] S. Zhang and D. Gao, "Flower pollination algorithm based on dynamic adjustment and collaborative search," *Computer Engineering and Applications*, vol. 55, no. 24, pp. 46–53, 2019.
- [23] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, and S. M. Mirjalili, "Salp swarm algorithm," *Advances in Engineering Software*, vol. 114, pp. 163–191, 2017.
- [24] Z. Xin, D. Zhang, Z. Chen, H. Zhang, and W. Yan, "Shared crow algorithm using multi-segment perturbation," *Computer Engineering and Applications*, vol. 56, no. 2, pp. 55–61, 2020.
- [25] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 3–18, 2011.
- [26] X. Meng, Y. Liu, X. Gao, and H. Zhang, "A new bio-inspired algorithm: chicken swarm optimization," in *Proceedings of the International Conference in Swarm Intelligence*, pp. 86–94, Hefei, China, October 2014.
- [27] A. H. Gandomi, X.-S. Yang, and A. H. Alavi, "Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems," *Engineering With Computers*, vol. 29, no. 1, pp. 17–35, 2013.
- [28] B. Akay and D. Karaboga, "Artificial bee colony algorithm for large scale problems and engineering design optimization," *Journal of Intelligent Manufacturing*, vol. 23, no. 4, pp. 1001–1014, 2012.
- [29] S. Akhtar, K. Tai, and T. Ray, "A SOCIO-behavioural simulation model for engineering design optimization," *Engineering Optimization*, vol. 34, no. 4, pp. 341–354, 2002.

- [30] E. Mezura-Montes, C. A. Coello Coello, and R. Landa-Becerra, "Engineering optimization using simple evolutionary algorithm," in *Proceedings of the 15th IEEE International Conference on Tools with Artificial Intelligence*, pp. 149–156, Sacramento, CA, USA, November 2003.

Research Article

On-Demanding Information Acquisition in Multi-UAV-Assisted Sensor Network: A Satisfaction-Driven Perspective

Hua Yang ¹, Jungang Yang ¹, Wendong Zhao ² and Cuntao Liu ²

¹School of Information and Communication, National University of Defense Technology, Xi'an 710000, China

²Communications Engineering College, Army Engineering University of PLA, Nanjing 210000, China

Correspondence should be addressed to Jungang Yang; yangjg_ty18@tom.com

Received 27 May 2021; Revised 30 August 2021; Accepted 14 September 2021; Published 13 October 2021

Academic Editor: Alessandro Formisano

Copyright © 2021 Hua Yang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

When multiple heterogeneous unmanned aerial vehicles (UAVs) provide service for multiple users in sensor networks, users' diverse priorities and corresponding priority-related satisfaction are rarely concerned in traditional task assignment algorithms. A priority-driven user satisfaction model is proposed, in which a piecewise function considering soft time window and users' different priority levels is designed to describe the relationship between user priority and user satisfaction. On this basis, the multi-UAV task assignment problem is formulated as a combinatorial optimization problem with multiple constraints, where the objective is maximizing the priority-weighted satisfaction of users while minimizing the total energy consumption of UAVs. A multipopulation-based cooperation genetic algorithm (MPCGA) by adapting the idea of "exploration-exploitation" into traditional genetic algorithms (GAs) is proposed, which can solve the task assignment problem in polynomial time. Simulation results show that compared with the algorithm without considering users' priority-based satisfaction, users' weighted satisfaction can be improved by about 47% based on our algorithm in situations where users' information acquisition is tight time-window constraints. In comparison, UAVs' energy consumption only increased by about 6%. Besides, compared with traditional GA, our proposed algorithm can also improve users' weighted satisfaction by about 5% with almost the same energy consumption of UAVs.

1. Introduction

Nowadays, unmanned aerial vehicles (UAVs) are gaining increasing popularity in various fields [1], such as situation awareness, intelligence reconnaissance, data collection, and relaying. Compared with traditional data collection means, UAVs possess significant advantages such as stronger mobility and flexibility, the ability to realize high-speed data transmission, and no risk of casualties, which makes them more suitable for data collection and transmission within sensor networks in complex environments [2, 3]. As the task execution capability of a single UAV is restricted by its limited flying capacity, battery capacity, reconnaissance capability, etc., it is imperative to use multiple UAVs to carry out cooperative data collection tasks. During this process, efficient task allocation [4–7] is one of the critical factors to improve the task execution efficiency of multiple UAVs

effectively. Besides, since task allocation problems and path planning problems in UAV-based data collection are highly coupled, they are usually considered together in practice.

Generally, a data collection task can be abstracted as a process of visiting multiple ground sensors (GSs). When considering multi-UAV-based data collection, a group of GSs and corresponding visiting sequence should be assigned to each UAV through a reasonable task assignment strategy. UAVs need to visit their assigned GSs in sequence, collecting data from the GSs and send it back, directly or indirectly, to users who need it. As a typical COP with multiple constraints, the multi-UAV-based data collection task assignment problem is NP-hard [8] and usually cannot be solved directly to get the optimal solution. Till now, a large amount of studies have focused on this problem and several customized COP method-based problem-solving models, such as cooperative multiple task assignment problem (CMTAP)

model, multiple vehicle routing problem (MVRP) model, multiple traveling salesman problem (MTSP) model, and mixed-integer linear programming (MILP) model, have been proposed [9–12]. While several kinds of problem-solving methods, such as dynamic programming [13], stochastic and deterministic optimization approaches (cross-entropy method and branch and bound algorithm) [14, 15], heuristic [16, 17], swarm intelligence [18–20], reinforcement learning [21, 22], and game theory [23, 24], have been widely used in realizing an effective task assignment.

Since UAV's onboard battery capacity, flight ability, and payload capacity are limited, related constraints such as energy consumption, flight speed, and amount of data that need to be collected are usually considered during the process of task assignment and path planning. Correspondingly, various optimization objectives in solving task assignment and path planning problems have been constructed from different perspectives. For example, studies [25–28] devoted to studying the minimization problem of the total energy consumption of UAVs participating in the reconnaissance tasks. The authors of [29] studied the fairness problem of UAVs' energy consumption. The authors of [30, 31] focused on the minimization problem of task completion time. While in these studies, problems of users' diverse priority-related satisfaction towards the execution effect of tasks are rarely considered.

However, in these studies, the influence of users' satisfaction toward the information obtained is rarely taken into account. In practical, as the purposes of users' acquiring information are different, the information from the same GS may also have different value/importance from users' point of view. Correspondingly, even if users receive the information they need at the same time, different users are likely to have different levels of satisfaction. For example, suppose both users u_1 and u_2 want to acquire information about target a . For u_1 , the information about a is vital, and the earlier the information is obtained, the higher the satisfaction of u_1 will be. While for u_2 , the information about a is not important, and as long as the information can be obtained before a deadline, the satisfaction of u_2 will be basically the same. In this situation, taking users' satisfaction into account is meaningful, as data collection services are supposed to be user-centric.

Generally, user satisfaction (US) is usually embodied in the process of users enjoying products/services [32], which is ubiquitous in nature and society. Kotler defines it as a psychological feeling derived from the comparison of the actual product/service experienced by users and their expectation [33], while Cardozo extends user satisfaction to the field of marketing, believing that it will influence customers' subsequent purchase behavior [34]. At present, there is no consensus on the understanding of user satisfaction degree in academia, and the main ideas can be divided into three categories: (1) users' expectation determines user satisfaction; (2) the quality of users' experience dominates user satisfaction; (3) the degree to which users' subjective expectation is consistent with their experience should be used

to describe user satisfaction. In this paper, we adopt the idea of describing user satisfaction degree as the distance between users' experience and psychological expectation of service quality, which can better reflect the connotation of user satisfaction in our opinions.

Currently, the above ideas of describing user satisfaction have been adopted in wireless communication [35], network management [36] as well as in multi-UAV task assignment [37–39]. As described in [32], user satisfaction (US) is an abstract concept and may be measured differently in different scenarios. When considering user satisfaction degree in the above studies, a satisfaction function related to information transmission rate, information acquisition time, or energy efficiency was defined based on their different optimization objectives. For example, the authors focused on a competitive environment, where different users are trying to meet their different QoS requirements in terms of data rate in a selfish manner [35], and the user satisfaction is then considered a QoS-related concept. On which basis, the game theory is adopted to realize a "satisfaction equilibrium" among users to balance meeting users' expectations and saving energy consumption. In [36], the authors focused on the network management problem, while the metric of user satisfaction is defined as a function of the network response time for serving the decision-making requests, which is used to help realizing an effective load-balancing of the decision-making requests. However, the diverse priorities of users were rarely considered in these studies. Generally, it is a simple but important aim to ensure that the demands of high-priority users should be met first in practice, i.e., users with higher priorities can get as higher satisfaction as possible. To do this, a user-priority-based satisfaction maximization problem related to users' demanding should be considered.

In this paper, we consider a scenario where multiple UAVs provide data collection services for multiple users with different priorities and optimize the task assignment problem by adopting a priority-based user satisfaction-driven strategy. The main contributions are as follows:

Considering users' diverse priorities, a priority-driven user satisfaction model is built to measure users' differentiated satisfaction towards the information obtained. Specifically, a piecewise function considering soft time window and users' priority levels is designed to describe the relationship between user priority and user satisfaction.

A satisfaction-driven multi-UAV cooperative task assignment problem is formulated as a COP, where the problem of maximizing priority-weighted user satisfaction and minimizing UAVs' total energy consumption is considered comprehensively, and weight factors are adapted to realize a trade-off between them.

To solve the task assignment problem efficiently, a multipopulation-based cooperation genetic algorithm (MPCGA) by introducing the idea of "exploration-exploitation" into traditional GAs is proposed. Numerical

results demonstrate the effectiveness of MPCGA in realizing an efficient user satisfaction-driven task assignment while minimizing total energy consumption.

2. System Model and Problem Formulation

2.1. System Model. As shown in Figure 1, we consider a multi-UAV-based data collection scenario, including one base station (BS), one relay UAV (UAV_r), N data collection UAVs (UAV_s), denoted by $\mathcal{N} = \{1, 2, \dots, N\}$, M users, denoted by $\mathcal{M} = \{1, 2, \dots, M\}$, and K GSs, denoted by $\mathcal{K} = \{1, 2, \dots, K\}$. Among them, rotary-wing UAVs that can hover above GSs when executing data collection are used. BS is responsible for receiving information requirements from users, assigning tasks to UAV_s, and distributing the collected information obtained by UAV_s to the corresponding users. UAV_s is responsible for visiting all the GSs in \mathcal{K} and send collected information back to BS, where the task of n ($n \in \mathcal{N}$) is to detect a subset of \mathcal{K} , which K_n denotes. In K_n , the order of the targets indicates the corresponding order that n visits them. Among them, the i th target in K_n is denoted by k_n^i and the serial number of k_n^i in \mathcal{K} is denoted by $f(k_n^i)$. UAV_r is responsible for data relay, while its position is supposed to be determined by BS and remain unchanged during the process of data collection. We assume that there is no direct communication link between each UAV_s and BS, while the data from a UAV_s to BS should be relayed through UAV_r, which can maintain connection with BS and UAV_s during the process of data collection.

In this paper, our goal is to find an optimal task allocation strategy that maximizes users' satisfaction with the information obtained while minimizing the total energy consumption of the data collection UAVs. The constraints, data collection and transmission process, energy consumption model, user satisfaction model, and optimization objective are described as follows.

2.1.1. Constraints of Target Visiting. During the data collection process, all UAVs are assumed to take off from BS simultaneously (time 0) and return to BS after finishing their tasks, i.e., conducting data collection on the respective task targets in turn and sending the collected data collection on the respective task targets data back to BS. Besides, it is assumed that a UAV can only visit a target, and the UAV can only visit it once. On this basis, the constraints of target visiting can be described by equation (1), where $|K_n|$ represents the number of elements in K_n .

$$\begin{cases} \mathcal{K} = \cup_{n=1}^N K_n, & n \in \mathcal{N}, \\ K_{n1} \cap K_{n2} = \emptyset, & n1, n2 \in \mathcal{N} \wedge n1 \neq n2, \\ \sum |K_n| = K & n \in \mathcal{N}. \end{cases} \quad (1)$$

2.1.2. Constraints of Flight Time. The location of the BS is denoted as $(0, 0, H_0)$, where H_0 is the height of BS. The location of target k is denoted by $l_k^e(t) = (x_k^e, y_k^e, H_k^e)$, and

the location of UAV_r is denoted by $l_r(t) = (x_r, y_r, H_r)$. Besides, the location of n is denoted by $l_n^u(t) = (x_n^u(t), y_n^u(t), H_n)$, where the horizontal coordinate of the UAVs' initial and final locations is $(0, 0)$. Ignoring the process of take off and landing, UAV_s n is supposed to fly at a constant height H_n during the whole data collection process, and the flight heights of UAVs are supposed to be different from each other to realize collision avoidance. Besides, it is supposed that n hovers above k ($k \in K_n$) when collecting data from k , while flying in a straight line with a constant speed V_0 in other cases. Denote the time when n arrives BS after finishing its data collection task as T_n^F , and then T_n^F can be calculated as follows:

$$T_n^F = T_n^f + T_n^h = \frac{\|\mathcal{L}_n\|}{V_0} + \sum_{i=1}^{|K_n|} \frac{I_n^i}{C_{n,i}}, \quad n \in \mathcal{N}, \quad (2)$$

where T_n^f and T_n^h represents the flight time and hover time of n , respectively. \mathcal{L}_n represents the flight trajectory of n , and $\|\mathcal{L}_n\|$ represents the Euclidean norm of \mathcal{L}_n . I_n^i represents the total amount of data that should be retrieved from k_n^i , and it is codetermined by the requirements of all users who need to acquire the information about k_n^i . Here, we assume that the difference in users' information acquirement demands about k_n^i is only reflected in the amount of the data collected, and I_n^i is selected as the maximum amount of data required by the users, which can be described as $I_n^i = \max\{x_{m,n}^i \cdot I_{m,n}^i\}$. Among them, $m \in \mathcal{M}$, $I_{m,n}^i$ represents the amount of data, collected from k_n^i , required by m , and $x_{m,n}^i$ is an indicator variable, which is used to indicate whether m needs the information about target k_n^i . If m needs the information about target k_n^i , and then $x_{m,n}^i = 1$; otherwise, $x_{m,n}^i = 0$. Besides, $C_{n,i}$ represents the data collection ability of n regard to k_n^i , i.e., the amount of data that can be transmitted from k_n^i to n per second. In this paper, we assume that the wireless channels between GSs and UAVs are dominated by line-of-sight (LoS) links, and the power gain of the channel between k_n^i and n is represented by $h_{n,i} = \beta_0 / (H_n - H_{k_n^i}^e)^2$, where β_0 represents the power gain at the reference distance $d_0 = 1$ m. Then, $C_{n,i}$ can be calculated as follows.

$$C_{n,i} = B \log_2 \left(\frac{1 + p_0 h_{n,i}}{\sigma^2} \right), \quad (3)$$

where p_0 represents the transmit power of $f(k_n^i)$, σ^2 is the additive white Gaussian noise (AWGN) power at the receiver, and B represents the available channel bandwidth.

Denote $T_{n,\max}$ as the maximum flight duration of UAV_s n ; then the constraints of flight time for UAVs can be expressed as follows.

$$T_n^F \leq T_{n,\max}, \quad n \in \mathcal{N}. \quad (4)$$

Considering that UAV_r is the last one to depart from BS and the first one to return to BS after finishing the relay of data collected from the last target, we assume that the endurance of UAV_r is sufficient enough during the data collection process as long as equation (4) can be satisfied.

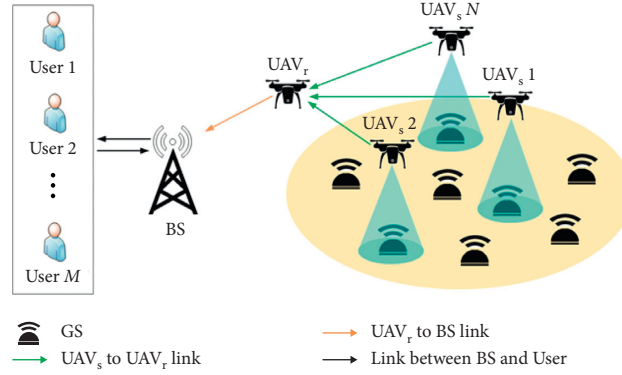


FIGURE 1: System model.

2.1.3. Data Collection and Transmission. Suppose the start and finish time of collecting data from k_n^i is $t_n^{i,0}$ and $t_n^{i,1}$, respectively, then $t_n^{i,0}$ and $t_n^{i,1}$ can be calculated as follows.

$$\begin{cases} t_n^{i,0} = t_n^{i-1,1} + \frac{\|\mathcal{L}_n^{i-1,i}\|}{V_0}, \\ t_n^{i,1} = t_n^{i,0} + \frac{\text{Inf}_n^i}{C_{n,i}}, \end{cases} \quad (5)$$

where $i \in [1, |K_n| + 1]$. Here, we use target sequence number 0 and $|K_n| + 1$ to represent BS to simplify the analysis process, e.g., $t_n^{0,1}$ represents the time when n takes off from BS and $t_n^{|K_n|+1,0}$ represents the time when n lands at BS. $\|\mathcal{L}_n^{i-1,i}\|$ is the length of flight trajectory segment with k_n^{i-1} and k_n^i as endpoints. Then, the time that BS receives the information about $f(k_n^i)$, denoted by $t_{B,n}^i$, can be calculated as follows.

$$t_{B,n}^i = t_n^{i,1} + \frac{I_n^i}{C_{n,r}} + \frac{I_n^i}{C_{r,B}}, \quad (6)$$

where $C_{n,r}$ represents the data transmission rate from n to UAV_r and $C_{r,B}$ represents the data transmission rate from UAV_r to the BS. We suppose $C_{n,r}$ and $C_{r,B}$ to be constant during the whole data collection process, which can be guaranteed by reasonable channel bandwidth allocation and channel access strategy. In this paper, an orthogonal frequency division multiple access (OFDMA) strategy similar as described in [40] is adopted, where the total available bandwidth is divided into multiple subcarriers with equal bandwidth. When multiple data collection UAVs need to transmit data to the relay UAV simultaneously, different subcarriers can be allocated to different UAVs, and the communication interference among UAVs is therefore ignored.

In this paper, the communication time between BS and users is not considered. That is, we take the time when BS receives the information about $f(k_n^i)$ as the time when the users, who need the information about $f(k_n^i)$, obtain the required information. Denote the time when m ($m \in \mathcal{M}$) get the information about $f(k_n^i)$ as $t_{m,n}^i$. Then, it can be described as follows.

$$t_{m,n}^i = \begin{cases} t_{B,n}^i, & x_{m,n}^i = 1, \\ 0, & x_{m,n}^i = 0. \end{cases} \quad (7)$$

2.1.4. Energy Consumption Model. Generally, the energy consumption of UAVs during flight can be divided into two parts: motion energy consumption and communication energy consumption, while ignoring the energy consumption during the take off and landing stage. Since the position of UAV_r and BS remain unchanged during the data collection process, the communication energy consumption between them, as well as the motion energy consumption of UAV_r , can be viewed as a constant term that is not affected by the task assignment strategies. In this section, we mainly consider the influence of task assignment strategies on the energy consumption of data collection UAVs.

(1) *Motion Energy Consumption.* During the data collection process, UAVs' movement state mainly includes two kinds: flying and hovering. The corresponding motion energy consumption is viewed as flight energy consumption and hover energy consumption, which mainly depends on the propulsion power of UAV in the two states. Here, the power consumption (watt) model derived in [27] is adopted, which is described as follows.

$$\begin{cases} P(V) = P_0 \left(1 + \frac{3V^2}{U_{\text{tip}}^2} \right) + \frac{P_i v_0}{V} + \frac{1}{2} d_0 \rho s A V^3, & \text{flight } (V \neq 0), \\ P_h = P_0 + P_i, & \text{hovering } (V = 0), \end{cases} \quad (8)$$

where $P_0 = \delta/8\rho s A \Omega^3 R^3$, $P_i = (1+l)W^{3/2}/\sqrt{2\rho A}$, which represents the profile power and induced power of UAV in hovering state, respectively. δ is airfoil drag coefficient, W is the weight of the UAV (Newton), Ω is blade angular velocity (radians/second), R is rotor radius (m), l is an incremental correction factor of the induced power, U_{tip} represents the tip velocity of the suspension blade, v_0 represents the average rotor induced velocity during hovering, and d_0 and s represent the fuselage resistance ratio and rotor compactness,

respectively, while ρ and A represent air density and rotor disc area, respectively.

For UAV _{n} , when it arrives at BS after finishing its data collection tasks, the total motion energy consumption, denoted by E_n^M , can be calculated as follows.

$$E_n^M = \int_0^{T_n^F} P(\|V_n(t)\|) dt. \quad (9)$$

Specially, when n is in flying state from $f(k_n^i)$ to $f(k_n^{i+1})$ ($i \in [0, |K_n|]$) with constant speed V_0 , its propulsion power consumption remains constant, and the energy consumption can be expressed by $P_n(V_0) \cdot T_n^f(i, i+1)$, where $T_n^f(i, i+1)$ represents the flying time spent from $f(k_n^i)$ to $f(k_n^{i+1})$. When n is in the hovering state while reconnaissance target $f(k_n^i)$, its energy consumption can be expressed by $P_h^n \cdot T_n^h(i)$, where $T_n^h(i)$ represents the hovering time spent over $f(k_n^i)$. On this basis, equation (9) can be reorganized as

$$\begin{aligned} E_n^M &= \sum_{i=0}^{|K_n|} (P_n(V_0) \cdot T_n^f(i, i+1) + P_h^n \cdot T_n^h(i)) \\ &= P_n(V_0) \cdot T_n^f + P_h^n \cdot T_n^h. \end{aligned} \quad (10)$$

Combining equation (2) and equation (10), E_n^M can be calculated by

$$E_n^M = P_n(V_0) \cdot \frac{\|\mathcal{L}_n\|}{V_0} + P_h^n \cdot \sum_{i=1}^{|K_n|} \frac{I_n^i}{C_n}. \quad (11)$$

(2) *Communication Energy Consumption.* UAV communication-related energy consumption mainly occurs during the process of signal processing, signal radiation, signal reception, etc. Here, we assume that the transmitted power of UAVs remain constant and use equation (12) [29] to calculate the communication energy consumption of transmitting the data about k_n^i from n to UAV _{r} :

$$E_{n,r}(k_n^i) = I_n^i (d_{n,r}(t_n^{i,0}))^\alpha e_{tx}, \quad (12)$$

where $d_{n,r}(t_n^{i,0})$ represents the communication distance between n and UAV _{r} when n collects data from k_n^i and e_{tx} represents the energy consumption generated by transmitting 1 bit data by 1 meter.

Based on the above analysis, the total energy consumption of data collection UAVs, denoted by E_{sum} , can be calculated by

$$E_{\text{sum}} = \sum_{n=1}^N \left(E_n^M + \sum_{i=1}^{|K_n|} E_{n,r}(k_n^i) \right). \quad (13)$$

2.1.5. User Satisfaction Model. When evaluating the satisfaction of users, several factors, such as information acquisition time and information acquisition quality (quantity, precision, etc.), are usually considered. Here, we suppose the information acquisition quality can be well guaranteed, and the information acquisition time is mainly considered when describing user satisfaction. To quantifying user satisfaction, the concept of soft time window is used. For user m , when

$x_{m,n}^i = 1$, denote the expected time window for obtaining the required information as $[0, t_{m,n}^{i,e}]$, and the acceptable time window for obtaining the information as $(t_{m,n}^{i,e}, t_{m,n}^{i,a}]$. When $t_{m,n}^i \in [0, t_{m,n}^{i,e}]$, take the information acquisition satisfaction, denoted by $S_{m,n}^i$, as 1; when $t_{m,n}^i \in [t_{m,n}^{i,e}, t_{m,n}^{i,a}]$, an exponential is designed to calculate the value of $S_{m,n}^i$; when $t_{m,n}^i \in [t_{m,n}^{i,a}, \infty)$, take $S_{m,n}^i$ as 0. Here, we use time 0 to represent the time when BS receives user's request, as well as the approximate time when data collection UAVs take off from BS.

Besides, considering different users usually possess different service priorities, their priorities are also considered when describing user satisfaction. Here, we suppose users' priorities to be known when BS conducts task assignment process, and the priority of m is denoted as P_m . Among them, P_m is a positive actual number, and the higher the value of P_m , the higher the priority of m . The priority-driven user satisfaction model with a soft time window is described as

$$S_{m,n}^i = \begin{cases} 1, & t_{m,n}^i \in [0, t_{m,n}^{i,e}], \\ A_m \cdot \exp\left(\frac{t_{m,n}^{i,a} - t_{m,n}^i}{t_{m,n}^{i,a} - t_{m,n}^{i,e}}\right) + B_m, & t_{m,n}^i \in (t_{m,n}^{i,e}, t_{m,n}^{i,a}], \\ 0, & \text{others,} \end{cases} \quad (14)$$

where $S_{m,n}^{i,\max} = 1$ represents the maximum satisfaction of m toward the information about target $f(k_n^i)$, $A = (1 - S_{m,n}^{i,\min}) / (\exp(t_{m,n}^{i,a} - t_{m,n}^{i,e}) - 1)$, and $B = S_{m,n}^{i,\min} - A$.

As shown in Figure 2, we appoint that the higher the priority of m , the smaller the value of the $S_{m,n}^{i,\min}$. In addition, we suppose that the higher the user's priority, the more time-sensitive their satisfaction is. That is, when m_1 and m_2 both require the information about $f(k_n^i)$ and their time window of acquiring information is the same, if $t_{m_1,n}^i > t_{m_2,n}^i$, then the satisfaction of high-priority users declines even faster over information acquisition time.

On this basis, a priority-oriented strategy is adopted to determine the value of $S_{n,\min}$, which is described as

$$S_{m,n}^{i,\min} = \frac{\text{Max}\{P_j\} - P_m + \varepsilon}{\text{Max}\{P_j\} - \text{Min}\{P_j\} + \varepsilon}, \quad j \in [1, M], \quad (15)$$

where ε is a positive real number used to make sure the formula always makes sense. Besides, it makes users, with different priorities, have different gradients of satisfaction on the information acquisition time. This is helpful to ensure that the satisfaction of high-priority users can be better guaranteed by the algorithm described in Section 2.2. In this paper, we choose ε as $(\text{Max}\{P_j\} - \text{Min}\{P_j\})/2$, i.e., the value of the minimum satisfaction of the highest priority users is $S_{m,n}^{i,\max}/3$, and that of the lowest priority users is 1.

Generally, in addition to considering the total satisfaction of users, we also need to ensure that the satisfaction of high-priority users is as high as possible when assigning tasks. To this end, users' satisfaction is weighed accordingly based on their priorities, and the weighted satisfaction of users, denoted by S_{sum}^W , can be described in equation (16). Among them, α is an amplification factor that is bigger than 1.

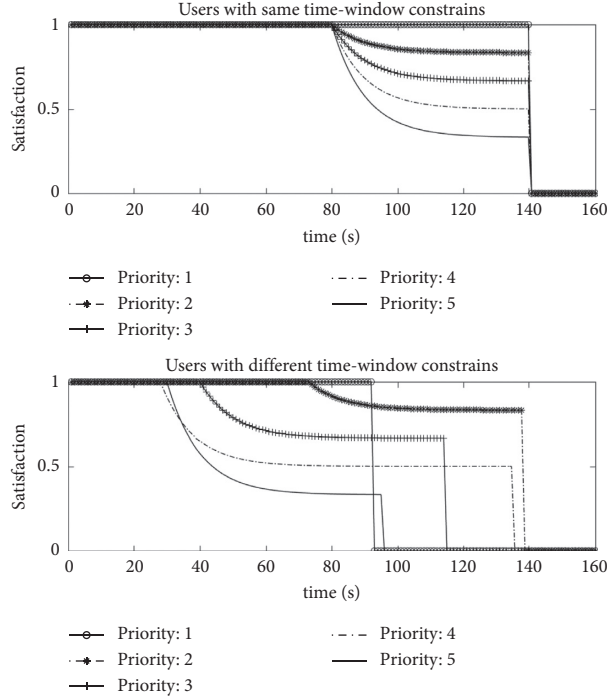


FIGURE 2: Change trend of users' satisfaction with information acquisition time.

$$S_{\text{sum}}^W = \sum_{m=1}^M \sum_{n=1}^N \sum_{i=1}^{|K_n|} P_m^\alpha \cdot x_{m,n}^i \cdot S_{m,n}^i \quad (16)$$

2.2. Problem Formulation. Our goal is to maximize S_{sum}^W while minimizing E_{sum} through practical task assignment, subject to the constraints of target visiting in (1) and the constraints of flight time in (4). Since it is difficult to meet the two objectives of maximizing user satisfaction and minimizing energy consumption at the same time, we introduce two weight factors ω_1 and ω_2 to achieve a trade-off between them. Among them, $\omega_1, \omega_2 \in [0, 1]$ and $\omega_1 + \omega_2 = 1$. On this basis, our optimization objective is formulated as

$$P1: \max_{a \in \mathcal{A}} \omega_1 \frac{S_{\text{sum}}^W}{S_{\text{max}}^W} + \omega_2 \frac{E_{\text{min}}}{E_{\text{sum}}}, \quad (17)$$

s.t. (1)(4),

where \mathcal{A} represents the set of task assignment strategies and $a \in \mathcal{A}$ represents a feasible solution in \mathcal{A} . S_{max}^W represents the maximum user satisfaction that can be achieved if taking user satisfaction maximization as optimization objective only, while E_{min} represents the minimum energy consumption that can be achieved if taking energy consumption minimization as optimization objective only. In particular, if we take $\omega_1 = 1$, problem P1 will degenerate into a satisfaction maximization problem, while problem P1 will degenerate into an energy consumption minimization problem if $\omega_1 = 0$.

Obviously, problem P1 is a nonconvex optimization problem due to the nonconvexity of the objective function,

and it is not easy to be solved directly to obtain an optimal solution. In this paper, we propose a multipopulation co-operation-based genetic algorithm (MPCGA), which preserves the advantages of genetic algorithm, i.e., simple, efficient, and fast convergence, and combines the advantages of swarm intelligence, i.e., solid global search-ability and the ability to jump out of locally optimal solutions.

3. Algorithm Description

3.1. Overview of GA. GA is a random search algorithm that simulates the genetic and evolutionary process of organisms, while it is suitable to deal with complex nonlinear optimization problems, such as COP, that are difficult to be solved by traditional search algorithms [41]. In GA, initial solutions are first generated randomly to form an initial population, where a solution is represented as a chromosome (or an individual). Then, the next population is generated through several evolutionary operators, i.e., selection operator, crossover operator, and mutation operator. In which process, a fitness function, which is closely related to the optimization objective, is needed to evaluate the performance of the solutions. After a certain number of iterations, with new populations constantly generated to renew their previous population, the algorithm can converge to the chromosome/individual with the best fitness value, which can be viewed as the optimal or suboptimal solution of the problem.

GA has the advantages of simple structure, high efficiency, fast convergence, etc., but it is easy to fall into local optimal solutions prematurely, which leads to insufficient global search-ability. By adopting the idea of "exploration-exploitation" which is widely used in current swarm

Input: \mathcal{N} , \mathcal{M} , $x_{m,n}^i$, \mathcal{K} , $\{l_k^e\}$, l_r , population size N_0 , population Number N_p , maximum iteration times Iter.

Output: the optimal feasible solution a_{opt} .

- (1) Calculate S_{max^W} and E_{min} , respectively
- (2) Population initialization
- (3) **for** $i = 1$ to N_p **do**
- (4) $a_i^0 \leftarrow$ the task strategy $a_{(i,1)}$ represented by the first individual in the i -th population;
- (5) **end**
- (6) $a_{\text{opt}}^0 \leftarrow \text{Max}\{a_{(i,1)} | i \in [1, N_p]\}$;
- (7) $l \leftarrow 1$;
- (8) **while** $l < \text{iter}$ **do**
- (9) **for** $i = 1$ to N_p **do**
- (10) **for** $j = 1$ to N_0 **do**
- (11) Calculate $S_{\text{sum},(i,j)}^W$ and $E_{\text{sum},(i,j)}$ achieved based on the task strategy $a_{(i,j)}$ that is represented by the j -th individual;
- (12) Calculate the fitness of $a_{(i,j)}$ through $f(a_{(i,j)})$;
- (13) *Evolutionary operation:* selection, crossover, and mutation;
- (14) **end**
- (15) Calculate the fitness of each individual in the current population;
- (16) Update a_i^l ;
- (17) **end**
- (18) Update a_{opt}^l ;
- (19) $l \leftarrow l + 1$;
- (20) **end**
- (21) $a_{\text{opt}} \leftarrow a_{\text{opt}}^l$;
- (22) **Return** a_{opt} .

ALGORITHM 1: MPCGA for multi-UAV task assignment.

Input: \mathcal{N} , \mathcal{K} , $\{l_k^e\}$, l_r .

Output: E_{min} .

- (1) Calculate the shortest flight distance from BS to GS k ($k \in \mathcal{K}$), denoted as $\text{dis}_{B,k}$, and that between $k1$ and $k2$ ($k1, k2 \in \mathcal{K}$), denoted as $\text{dis}_{k1,k2}$, respectively
- (2) Obtaining the mileage that can be saved if visit $k1$ and $k2$ one after the other in the same flight path according to saving-mileage formula, i.e., $\Delta \text{dis}_{k1,k2} = \text{dis}_{B,k1} + \text{dis}_{B,k2} - \text{dis}_{k1,k2}$
- (3) Sort the saving-mileage in descending order
- (4) According to the constraints of flight time and energy consumption, as well as the value of saving-mileage, connect each GS sequentially to finally determine the flight routes of data collection UAVs, as well as the number of UAVs used
- (5) Calculate E_{min} according to (13)
- (6) Return E_{min}

ALGORITHM 2: MSA for calculating E_{min} .

intelligence algorithms, such as ant colony algorithm, particle swarm algorithm, and artificial fish swarms algorithm [20, 42, 43]. The MPCGA is proposed and described in detail as follows.

3.2. Description of MPCGA. In this section, the MPCGA is proposed, and its specific steps are shown in Algorithm 1. Following are the description of some details:

S_{max^W} is calculated by assuming that all users' satisfaction toward their desired information is 1, i.e., $S_{\text{max}^W} = \sum_{m=1}^M \sum_{n=1}^N \sum_{i=1}^{|K_n|} P_m^\alpha \cdot x_{m,n}^i$.

Since the difference of UAVs' total energy consumption under different task allocation strategies is mainly caused by the difference of UAVs' total flying distance.

Here, we use a mileage-saving algorithm (MSA) [44], which is more accurate than GAs, to calculate E_{min} and its specific steps is as showed in Algorithm 2.

a_i^l represents the local optimal feasible solution of the i th population in the l th iteration.

a_{opt}^l represents the global optimal feasible solutions of the N_p populations till the l -th iteration.

$f(\cdot)$ is the fitness function based on S_{sum}^W and E_{sum} , where

$$f(a_{(i,j)}) = \omega_1 (S_{\text{sum},(i,j)}^W / S_{\text{max}^W}) + \omega_2 (E_{\text{min}} / E_{\text{sum},(i,j)}).$$

Evolutionary operators: (1) selection operator: when conducting selection operation for the i -th population, a proportional roulette selection operator is used within the population. While the a_i^l and a_{opt}^l can also be selected as a parent with a certain probability that is

independent of each other. (2) crossover operator: sequential crossover operator is used in the algorithm. (3) mutation operator: “two-element swap” operation is used in the algorithm.

We implement different crossover and mutation probabilities for different populations to enhance the global search-ability of our algorithm.

3.3. Time Complexity Analysis of MPCGA. The time complexity of MPCGA is mainly dependent on two parts: (1) the complexity of calculating S_{\max}^w and E_{\min} ; (2) the complexity of multipopulation cooperation based GA. Generally, the complexity of calculating S_{\max}^w is $O(1)$, and the complexity of calculating E_{\min} is $O(K^2)$. Besides, the complexity of multipopulation cooperation-based GA can be viewed as $O(N_p N_0 \text{Ite})$. Then, the time complexity of MPCGA can be approximate as $O(N_p N_0 \text{Ite} + K^2)$, which is polynomial.

4. Simulation Results

In this section, the performance of our proposed MPCGA for the multi-UAV task assignment problem is evaluated, while traditional GA with/without user’ satisfaction considered is also simulated for comparison. Among them, the GA with user’ satisfaction considered is denoted as BGA, while the algorithm without considering user satisfaction is denoted as GAWS.

4.1. Parameter Setting. The targets are assumed to be randomly distributed in a 1.5 km \times 1 km rectangular area, while the circular area’s center is 1.75 km far from BS. Regarding the coordinate of BS as (0, 0, 25), set up a coordinate system with the line between BS and the center of the circular area as the X axis. That is, $x_k^e \in [1000 \ 2500]$ and $y_k^e \in [-500 \ 500]$. Besides, H_k^e is supposed to follow a uniform distribution within [0 60]. The default number of targets, users, and available UAV_s is set as 50, 10, and 5, respectively. Other major simulation parameters are as shown in Table 1.

4.2. Performance Evaluation. Figure 3 shows the convergence of MPCGA. It can be seen that, with the increase of the number of populations, the global search ability of the MPCGA will be enhanced to some extent, but when the number of populations is greater than a certain value, e.g., 8, the optimal solution converges to almost the same value, which means that to increase the number of populations too much does not make much sense. Besides, too large several populations will also increase the calculation complexity of the algorithm obviously. To show this more clearly, Table 2 displays the execution time of MPCGA with different population numbers, where N_0 is set as 40, and Ite is set as 200. From the simulation results, one can see that the execution time of MPCGA will increase near linearly as the number of population increases, which is in line with the theoretical analysis results in Section 3. And, it indicates that, in the case of a similar convergence rate, a smaller

population number is beneficial to reduce the running time of MPCGA.

Table 3 displays the comparative results, i.e., the total energy consumption of UAVs, total weighted user satisfaction, and the max completion time of tasks, of MPCGA, BGA, and GAWS with $\omega_1 = 0.7$ and $\omega_2 = 0.3$. Among them, the situation with loose time window constraints means that most users can obtain their required information within their expected time window when UAVs visit the GSs according to the shortest path. In contrast, the situation with tight time window constraints means that most users cannot obtain their required information within their expected time window when UAVs visit the GSs according to the shortest path. The simulation results show that our proposed algorithm performs best in both situations, i.e., the highest weighted user satisfaction can be achieved at the cost of a small amount of energy consumption. In particular, when in the situation with tight time window constraints, our proposed algorithm can improve the weighted user satisfaction by about 47% compared with GAWS, while the energy consumption only increased by about 6%. Besides, compared with BGA, our proposed algorithm can also improve the weighted user satisfaction by about 5% with almost the same energy consumption.

To better show users’ satisfaction with different priorities, the simulation results of each user’s satisfaction when using different algorithms under the situation with tight time window constraints are shown in Figure 4. Among them, users’ priorities are showed as labels, and the situations where users are with the same/different time-window constraints are shown, respectively. We can see from the simulation results that users with higher priorities usually can obtain higher satisfaction when completing task assignments by using MPCGA than using BGA or GAWS in both situations. For example, user 7 and user 8 can realize the highest satisfaction as they are with the highest priority, which is consistent with our original intention. Although user 7 is more satisfied than user 8 in some cases, while the opposite is true in other cases, considering their equal priority, this is an acceptable task assignment result. In addition, the average weighted user satisfaction of users with different priorities is shown in Figure 5. Sometimes users with higher priorities may achieve higher satisfaction when conducting task assignments by using BGA than MPCGA, e.g., the satisfaction of user 8 in Figure 4(a). However, from the average satisfaction of users with the same priority, MPCGA still performs better than BGA.

Figure 6 shows the weighted user satisfaction as well as total energy consumption of UAVs with different N , where $M = 10$ and $K = 50$. One can see that, with the increase of N , the weighted user satisfaction will increase correspondingly, while it finally converges since all the users can get their required data within their expected time window as long as N is big enough. For the total energy consumption of UAVs, it will still increase with the increase of N even after the user satisfaction has converged, which means that more data collection UAVs is not always better as it will cause extra energy consumption when N is large, and each UAV is assigned a mission set. On this basis, if we want to meet

TABLE 1: Simulation parameters.

Notation	Physical meaning	Value
V_0	Flight speed of UAVs	20 m/s
$\{H_n\}$	Range of UAVs' flight height	100~150 m
$\{I\}$	Range of data quantity collected from one target	40~80 Mbit
B	Communication bandwidth between GSs and UAVs	2 MHz
C_n, r	Data transmission rate between UAV _s and UAV _r	4 Mbps
C_r, B	Data transmission rate between UAV _r and the BS	8 Mbps
T_{max}	Maximum flight duration of UAV _s	600 s
P_m	The priority of user m	1~5
α	The amplification factor for users' priority	2
e_{tx}	Energy consumption parameter of communication	10 pJ/(m bit)
$\{H_k^z\}$	Height range of GSs	0~60 m
σ^2	Additive white Gaussian noise (AWGN) power	-174 dBm
δ	Airfoil drag coefficient	0.012
W	Weight of UAV	20 N
Ω	Blade angular velocity	300 rad/s
R	Rotor radius	0.4 m
U_{tip}	Tip speed of the rotor blade	120 m/s
v_0	Mean rotor induced velocity in hovering	4.03
ρ	Air density	1.225 kg/m ³
A	Rotor disc area	0.503 m ²
d_0	Fuselage resistance ratio	0.6
s	Rotor solidity	0.05
p_0	Transmit power of GSs	5 mW
β_0	Power gain at the reference distance $d_0 = 1$ m	-50 dB

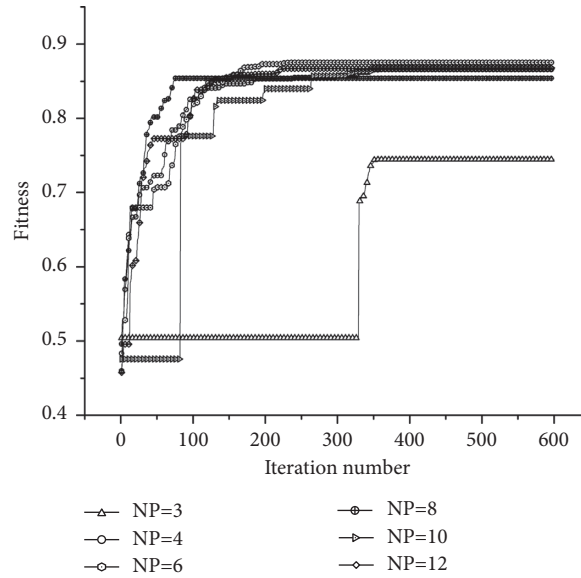


FIGURE 3: Convergence of MPCGA.

TABLE 2: Execution time of MPCGA with different population numbers.

NP	Execution time (s)
3	6.48
4	8.78
6	12.59
8	17.17
10	20.95
12	25.09

TABLE 3: Simulation results of MPCGA, BGA, and GAWS.

Algorithms		Energy consumption (J)	Task completion time (S)	Weighted user satisfaction
MPCGA	Loose time window constraints	$3.21 \cdot 10^5$	251	2235.80
	Tight time window constraints	$4.20 \cdot 10^5$	365	1854.92
BGA	Loose time window constraints	$3.60 \cdot 10^5$	271	2112.75
	Tight time window constraints	$4.26 \cdot 10^5$	339	1767.31
GAWS	Loose time window constraints	$3.12 \cdot 10^5$	314	1951.44
	Tight time window constraints	$3.95 \cdot 10^5$	350	1259.36

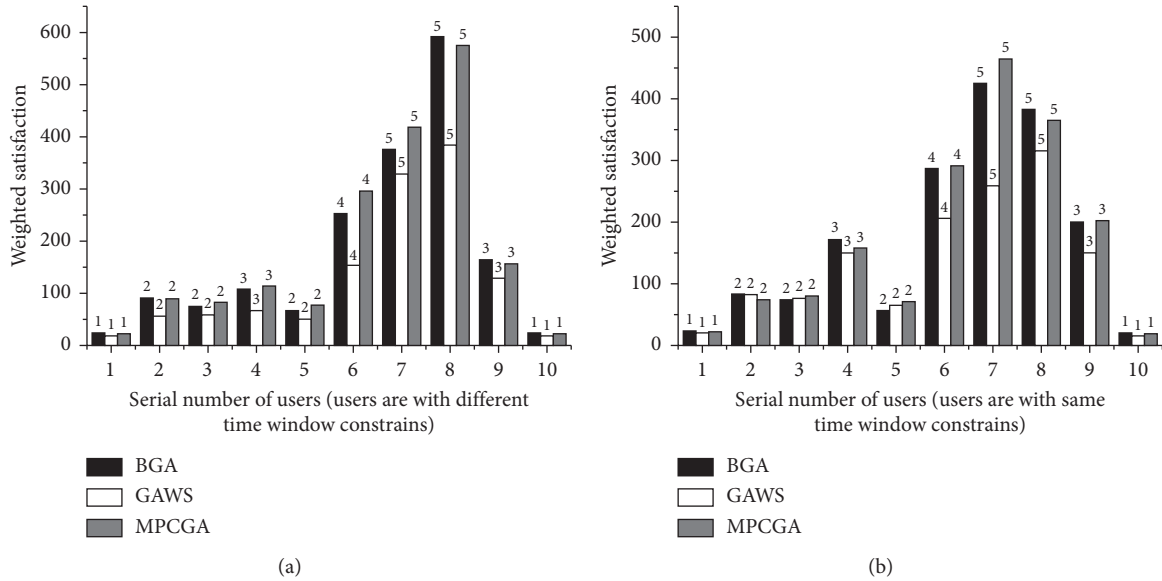


FIGURE 4: Weighted satisfaction of users (a) with different time window constraints and (b) with the same time window constraints.

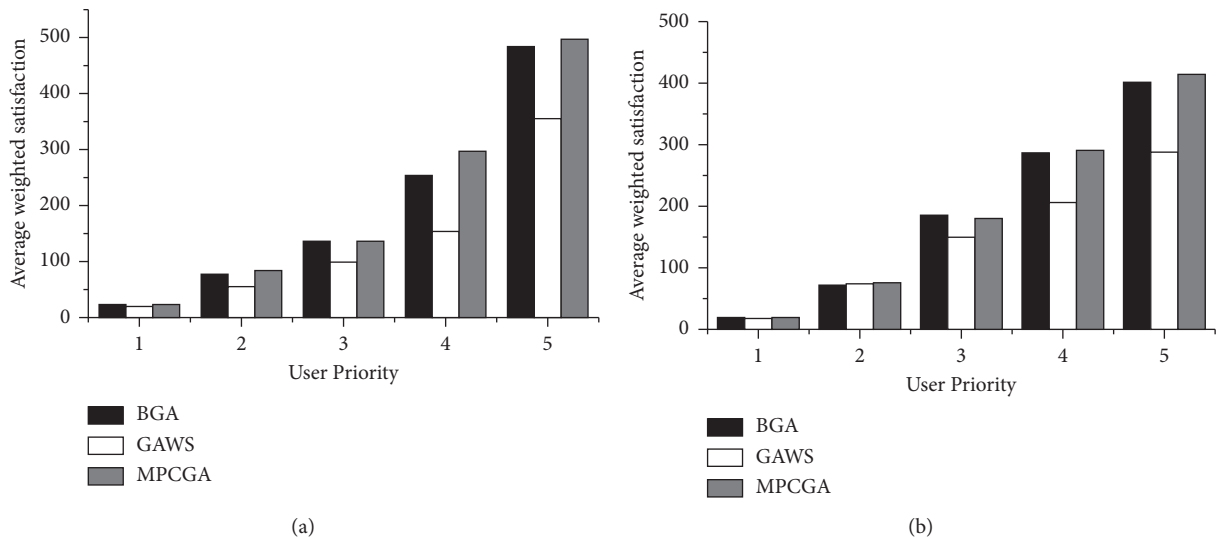


FIGURE 5: Average weighted satisfaction of users with different priorities (a) with different time window constraints and (b) with the same time window constraints.

users' needs as much as possible while saving energy consumption, the reasonable number of UAVs that are needed can be determined.

Figure 7 displays the average weighted user satisfaction as well as total energy consumption of UAVs with different K , where $N = 5$ and $M = 10$. It can be seen that, with the

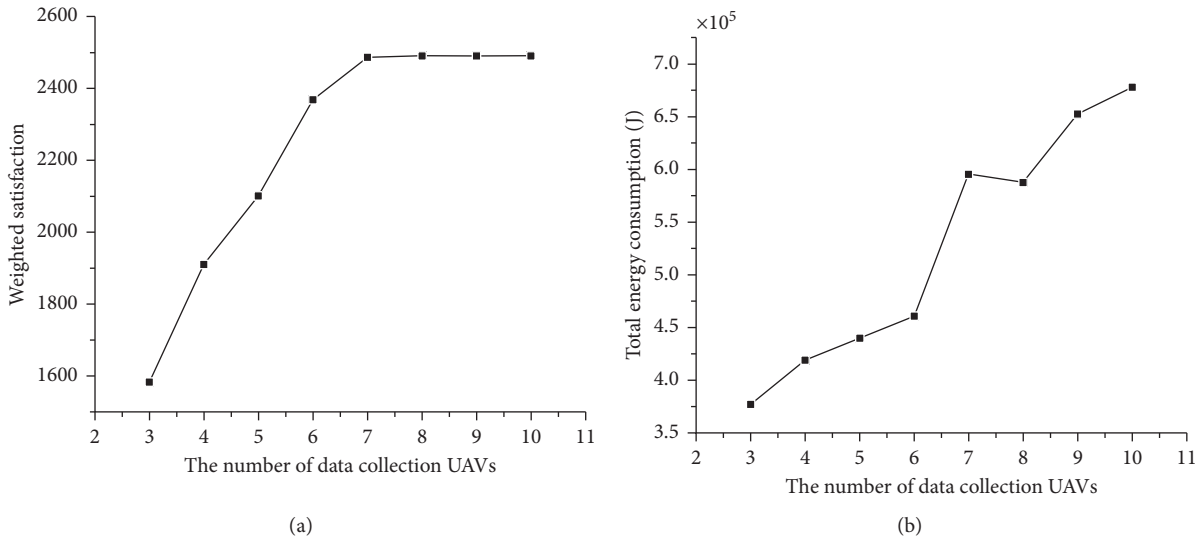


FIGURE 6: Variation trend of weighted satisfaction and energy consumption with the number of UAVs.

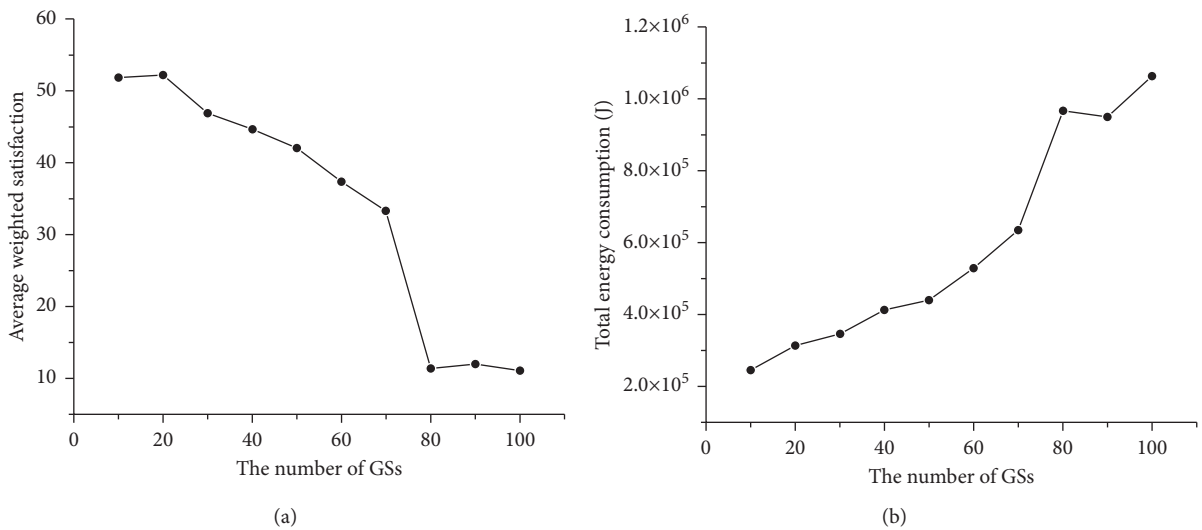


FIGURE 7: Variation trend of weighted satisfaction and energy consumption with the number of GSs.

increase of K , the average weighted user satisfaction will decrease while the total energy consumption of UAVs will increase. The main reason is that with the increase of K , the UAVs need to visit more GSs, which will cause more flight time and hover time, resulting in more energy consumption. Besides, as it takes longer for users to get the data they need, their satisfaction will decrease correspondingly.

Figure 8 shows the weighted user satisfaction as well as total energy consumption of UAVs with different I , where $N = 5$, $M = 10$, and $K = 50$. From the simulation results, one can see that, with the increase of I , the average weighted user satisfaction will decrease since the data collection and data transmission time will increase, which results in a long time for users to get their required data. Besides, the total energy consumption of UAVs presents an upward trend as

the hover energy consumption, and communication energy consumption of UAVs will increase. It is noted that since the minimization of energy consumption is considered in the objective function, the total energy consumption of UAVs does not always increase with the increase of I .

The impact of M on weighted user satisfaction as well as total energy consumption of UAVs is displayed in Figure 9. It can be seen that, with the increase of M , although the total user satisfaction will increase, the average satisfaction is about the same, and the total energy consumption of UAVs tends to stabilize. The main reason is that the increase of M may affect the order of UAVs' visiting GSs, but not the number of GSs visited and the time of data collection and transmission. As a result, the total distance traveled by UAVs will vary, but there will not be significant fluctuations.

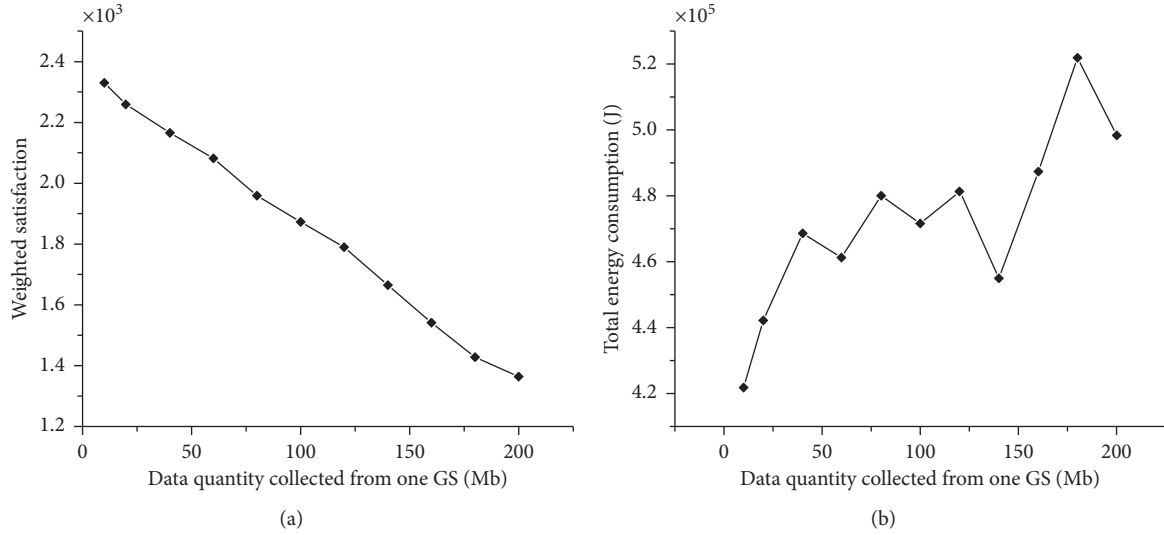


FIGURE 8: Variation trend of weighted satisfaction and energy consumption with the data quantity collected from one GS.

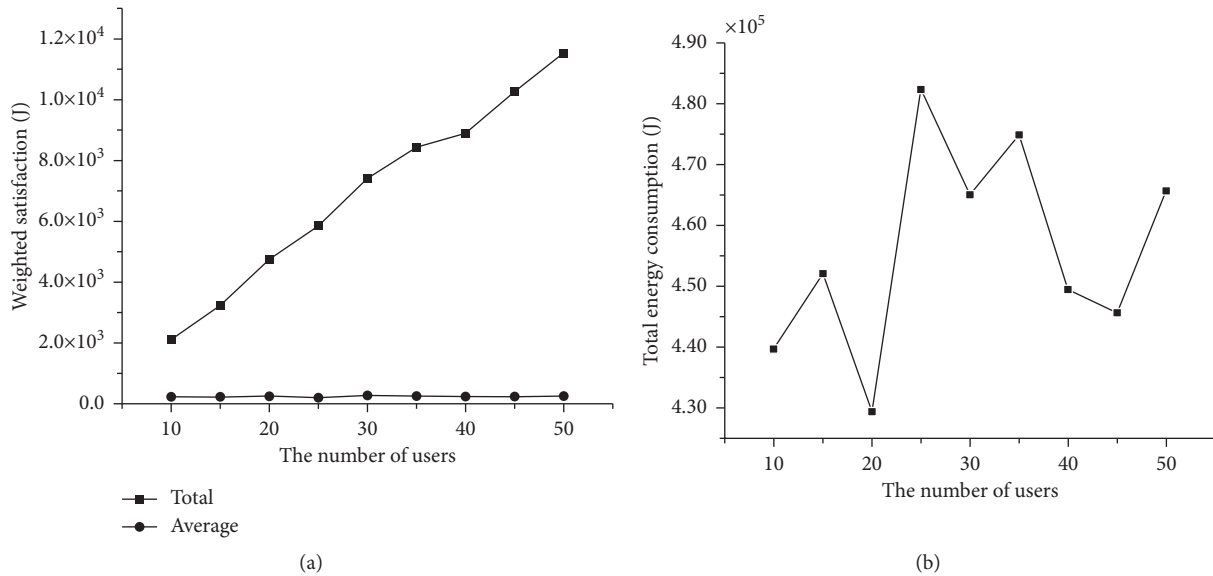


FIGURE 9: Variation trend of weighted satisfaction and energy consumption with the number of users.

5. Conclusion

This paper studied a multi-UAV-based sensor network where multiple UAVs need to collect data from multiple GSs in sequence and transmit the information back to BS through a relay UAV. In the process of task assignment, considering different users' diverse priorities and corresponding priority-related satisfaction, a priority-driven user satisfaction model was constructed, where a piecewise function considering soft time window and users' priority levels was designed to describe user satisfaction. A combinatorial optimization problem with multiple constraints was formulated, where the objective is maximizing the priority-weighted satisfaction of users while minimizing the total energy consumption of UAVs. Furthermore, a multi-population-based cooperation genetic algorithm (MPCGA)

was proposed by adopting the idea of "exploration-exploitation" into traditional GA. Simulation results showed the convergence and the effectiveness of our proposed algorithm.

In the follow-up work, we will consider the distribution features of GSs and the priority-based fairness problem between users to improve our algorithm's effectiveness and applicability further.

Data Availability

No data were used to support this study.

Disclosure

Hua Yang and Cuntao Liu are co-first authors.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

Hua Yang and Cuntao Liu contributed equally to this work.

References

- [1] J. Qiu, D. Grace, G. Ding, M. D. Zakaria, and Q. Wu, "Air-ground heterogeneous networks for 5G and beyond via integrating high and low altitude platforms," *IEEE Wireless Communications*, vol. 26, no. 6, pp. 140–148, 2019.
- [2] Z. Qin, C. Dong, A. Li, H. Dai, Q. Wu, and A. Xu, "Trajectory planning for reconnaissance mission based on fair-energy UAVs cooperation," *IEEE Access*, vol. 7, pp. 91120–91133, 2019.
- [3] J. Dai, J. Cheng, and M. Song, "Cooperative task assignment for heterogeneous multi-UAVs based on differential evolution algorithm," in *Proceedings of the 2009 IEEE International Conference on Intelligent Computing and Intelligent Systems*, pp. 163–167, Shanghai, China, November 2009.
- [4] X. Zheng, F. Zhang, T. Song, and D. Lin, "Heterogeneous multi-UAV distributed task allocation based on CBBA," in *Proceedings of the 2019 IEEE International Conference on Unmanned Systems (ICUS)*, pp. 704–709, Beijing, China, October 2019.
- [5] Z. Qin, C. Dong, H. Wang et al., "Trajectory planning for data collection of energy-constrained heterogeneous uavs," *Sensors*, vol. 19, no. 22, p. 4884, 2019.
- [6] C. Sampedro, H. Bavle, J. L. Sanchez-Lopez et al., "A flexible and dynamic mission planning architecture for UAV swarm coordination," in *Proceedings of the 2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 355–363, Arlington, VA, USA, June 2016.
- [7] N. H. Motlagh, M. Bagaa, and T. Taleb, "Energy and delay aware task assignment mechanism for UAV-based IoT platform," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6523–6536, 2019.
- [8] Y. Cai, "Artificial fish school algorithm applied in a combinatorial optimization problem," *International Journal of Intelligent Systems and Applications*, vol. 2, no. 1, pp. 37–43, 2010.
- [9] T. Shima, S. J. Rasmussen, A. G. Sparks, and K. M. Passino, "Multiple task assignments for cooperating uninhabited aerial vehicles using genetic algorithms," *Computers & Operations Research*, vol. 33, no. 11, pp. 3252–3269, 2006.
- [10] A. Arsie, K. Savla, and E. Frazzoli, "Efficient routing algorithms for multiple vehicles with no explicit communications," *IEEE Transactions on Automatic Control*, vol. 54, no. 10, pp. 2302–2317, 2009.
- [11] Q. Xia, S. Liu, M. Guo, H. Wang, Q. Zhou, and X. Zhang, "Multi-uav trajectory planning using gradient-based sequence minimal optimization," *Robotics and Autonomous Systems*, vol. 137, no. 4, Article ID 103728, 2021.
- [12] Y. Chen, D. Yang, and J. Yu, "Multi-UAV task assignment with parameter and time-sensitive uncertainties using modified two-Part Wolf pack search algorithm," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 54, no. 6, pp. 2853–2872, 2018.
- [13] G. Yang and V. Kapila, "A dynamic-programming-styled algorithm for time-optimal multi-agent task assignment," in *Proceedings of the 40th IEEE Conference on Decision and Control (Cat. No.01CH37228)*, vol. 2, pp. 1959–1964, Orlando, FL, USA, December 2001.
- [14] C. Ramirez-Atencia, G. Bello-Orgaz, M. D. R. Moreno, and D. Camacho, "Branching to find feasible solutions in unmanned air vehicle mission planning," in *Proceedings of the 15th International Conference on Intelligent Data Engineering and Automated Learning*, vol. 8669, Hong Kong, China, December 2014.
- [15] H. A. Le Thi, D. M. Nguyen, and T. Pham Dinh, "Globally solving a nonlinear uav task assignment problem by stochastic and deterministic optimization approaches," *Optimization Letters*, vol. 6, no. 2, pp. 315–329, 2012.
- [16] L. Cheng, L. Zhong, S. Tian, and J. Xing, "Task assignment algorithm for road patrol by multiple UAVs with multiple bases and rechargeable endurance," *IEEE Access*, vol. 7, pp. 144381–144397, 2019.
- [17] J. Wang, Y. F. Zhang, L. Geng, J. Y. H. Fuh, and S. H. Teo, "A heuristic mission planning algorithm for heterogeneous tasks with heterogeneous uavs," *Unmanned Systems*, vol. 03, no. 03, pp. 205–219, 2015.
- [18] G. Jia, W. Jianfeng, W. Peng, C. Qingyang, and W. Yujie, "Using multi-layer coding genetic algorithm to solve time-critical task assignment of heterogeneous UAV teaming," in *Proceedings of the 2019 International Conference on Control, Automation and Diagnosis (ICCAD)*, pp. 1–5, Grenoble, France, July 2019.
- [19] S. Boumediene, C. Samira, and A. Hassane, "Fuzzy swarm trajectory tracking control of unmanned aerial vehicle," *Journal of Computational Design and Engineering*, vol. 7, no. 4, pp. 435–447, 2020.
- [20] F. Yan, "Autonomous vehicle routing problem solution based on artificial potential field with parallel ant colony optimization (aco) algorithm," *Pattern Recognition Letters*, vol. 116, pp. 195–199, 2018.
- [21] H. Qie, D. Shi, T. Shen, X. Xu, Y. Li, and L. Wang, "Joint optimization of multi-UAV target assignment and path planning based on multi-agent reinforcement learning," *IEEE Access*, vol. 7, pp. 146264–146272, 2019.
- [22] D. J. Kwak, S. Moon, S. Kim, and H. J. Kim, "Optimization of decentralized task assignment for heterogeneous uavs," *IFAC Proceedings Volumes*, vol. 46, no. 11, pp. 251–256, 2013.
- [23] P. Chandler, M. Pachter, S. Rasmussen, and C. Schumacher, "Multiple task assignment for a UAV team," in *Proceedings of the 2009 Aiaa Guidance, Navigation, & Control Conference & Exhibit*, Monterey, California, August 2002.
- [24] H. Dai, H. Zhang, C. Li, and B. Wang, "Efficient deployment of multiple uavs for iot communication in dynamic environment," *China Communications*, vol. 17, no. 1, pp. 89–103, 2020.
- [25] Y. Zeng, R. Zhang, and T. J. Lim, "Wireless communications with unmanned aerial vehicles: opportunities and challenges," *IEEE Communications Magazine*, vol. 54, no. 5, pp. 36–42, 2016.
- [26] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Mobile unmanned aerial vehicles (UAVs) for energy-efficient internet of things communications," *IEEE Transactions on Wireless Communications*, vol. 16, no. 11, pp. 7574–7589, 2017.
- [27] Y. Zeng, J. Xu, and R. Zhang, "Energy minimization for wireless communication with rotary-wing UAV," *IEEE Transactions on Wireless Communications*, vol. 18, no. 4, pp. 2329–2345, 2019.
- [28] Y. Wu, B. Zhang, S. Yang, X. Yi, and X. Yang, "Energy-efficient joint communication-motion planning for relay-assisted wireless robot surveillance," in *Proceedings of the*

- IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, pp. 1–9, Atlanta, GA, USA, May 2017.
- [29] Z. Qin, C. Dong, A. Li, H. Dai, Q. Wu, and A. Xu, "Trajectory planning for reconnaissance mission based on fair-energy UAVs cooperation," *IEEE Access*, vol. 7, pp. 91120–91133, 2019.
- [30] Z. Qin, A. Li, C. Dong, H. Dai, and Z. Xu, "Completion time minimization for multi-uav information collection via trajectory planning," *Sensors*, vol. 19, no. 18, p. 4032, 2019.
- [31] H. S. Yavuz, H. Göktas, H. Çevikalp, and H. Sarıbaş, "Optimal task allocation for multiple UAVs," in *Proceedings of the 2020 28th Signal Processing and Communications Applications Conference (SIU)*, pp. 1–4, Gaziantep, Turkey, October 2020.
- [32] R. L. Oliver, *Satisfaction: A Behavioral Perspective on the Consumer*, M. E. Sharpe, Armonk, NY, USA, 2010.
- [33] P. Kotler, S. H. Ang, and C. T. Tan, "Marketing management: an asian perspective," *Australasian Marketing Journal*, vol. 14, no. 2, p. 52, 2003.
- [34] R. N. Cardozo, "An experimental study of customer effort, expectation, and satisfaction," *Journal of Marketing Research*, vol. 2, no. 3, pp. 244–249, 1965.
- [35] M. Fasoulakis, E. E. Tsiropoulou, and S. Papavassiliou, "Satisfy instead of maximize: improving operation efficiency in wireless communication networks," *Computer Networks*, vol. 159, pp. 135–146, 2019.
- [36] C. Laurent, Z. Altman, E. Patouni et al., "Coordination of self-organizing network mechanisms: framework and enablers," in *International Conference on Mobile Networks and Management*, pp. 174–184, Springer, Berlin, Germany, 2011.
- [37] A. Gao, Y. Hu, W. Liang, Y. Lin, L. Li, and X. Li, "A QoE-oriented scheduling scheme for energy-efficient computation offloading in UAV cloud system," *IEEE Access*, vol. 7, pp. 68656–68668, 2019.
- [38] X. Liu, M. Chen, and C. Yin, "Optimized trajectory design in UAV based cellular networks for 3D users: a double Q-learning approach," *Journal of Communications and Information Networks*, vol. 4, no. 1, pp. 24–32, 2019.
- [39] D. Liu, J. Wang, K. Xu et al., "Task-driven relay assignment in distributed UAV communication networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 11, pp. 11003–11017, 2019.
- [40] E. E. Tsiropoulou, P. Vamvakas, and S. Papavassiliou, *Resource Allocation in Multi-Tier Femtocell and Visible-Light Heterogeneous Wireless Networks*, *Resource Allocation in Next-Generation Broadband Wireless Access Networks*, IGI Global, Hershey, PA, USA, 2017.
- [41] Y. Maknoon and G. Laporte, "Vehicle routing with cross-dock selection," *Computers & Operations Research*, vol. 77, pp. 254–266, 2017.
- [42] P. Yan and Bo Tan, "Evolutionary group theoretic tabu search approach to task allocation of autonomous unmanned aerial vehicles," in *Proceedings of the 2013 10th IEEE International Conference on Control and Automation (ICCA)*, Hangzhou, China, June 2013.
- [43] Y. Chen, N. Li, C. Wang, W. Xie, and J. Xv, "A 3D placement of unmanned aerial vehicle base station based on multi-population genetic algorithm for maximizing users with different QoS requirements," in *Proceedings of the 2018 IEEE 18th International Conference on Communication Technology (ICCT)*, pp. 967–972, Chongqing, China, October 2018.
- [44] M. J. Zhu and J. W. Zhang, "Optimization study based on the mileage-saving method of oil distribution," *Advanced Materials Research*, vol. 779–780, pp. 1805–1808, 2013.

Research Article

Remote Sensing–Based Urban Green Space Detection Using Marine Predators Algorithm Optimized Machine Learning Approach

Nhat-Duc Hoang ^{1,2} and Xuan-Linh Tran ^{1,2}

¹Institute of Research and Development, Duy Tan University, Da Nang 550000, Vietnam

²Faculty of Civil Engineering, Duy Tan University, Da Nang 550000, Vietnam

Correspondence should be addressed to Nhat-Duc Hoang; hoangnhatduc@duytan.edu.vn

Received 2 March 2021; Revised 9 April 2021; Accepted 14 May 2021; Published 24 May 2021

Academic Editor: Qingzheng XU

Copyright © 2021 Nhat-Duc Hoang and Xuan-Linh Tran. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Information regarding the current status of urban green space is crucial for urban land-use planning and management. This study proposes a remote sensing and data-driven solution for urban green space detection at regional scale via employment of state-of-the-art metaheuristic and machine learning approaches. Remotely sensed data obtained from Sentinel 2 satellite in the study area of Da Nang city (Vietnam) are used to construct and verify an intelligent model that hybridizes Marine Predators Algorithm (MPA) and support vector machines (SVM). SVM are employed to generalize a decision boundary that separates features characterizing statistical measurements of remote sensing data into two categories of “green space” and “nongreen space”. The MPA metaheuristic is used to optimize the SVM training phase by identifying an appropriate set of the SVM’s hyperparameters including the penalty coefficient and the kernel function parameter. Experimental results show that the proposed model which processes information provided by all of the Sentinel 2 satellite’s spectral bands can deliver a better performance than those obtained from the model based on vegetation indices. With a good classification accuracy rate of roughly 93%, an F1 score = 0.93, and an area under the receiver operating characteristic = 0.98, the newly developed model is a promising tool to assist local authority to obtain up-to-date information on urban green space and develop plans of sustainable urban land use.

1. Research Background and Motivation

In many regions around the globe, fast pace of urbanization leads to various problems including traffic congestion, poor air quality, and noise pollution. As pointed out by Xian et al. [1], urbanization significantly transforms the landscape from the natural surface types to impervious surface such as housing, commercial building, and infrastructures. This transformation is happening in a large spatial extent and an increasing speed due to a burgeoning pressure on additional housing and commercial/industrial areas. Moreover, developments of urban lands consume areas of green land areas and bring about negative impacts on the urban environments [2–4]. Worsening living environment caused by lack of green space is a major issue for human health because roughly 54% of people in the world are living in urban areas [5].

Urban green space is generally defined as green infrastructure that contains vegetated spaces including urban parks, road, and workplace green space [6]. Previous works have recognized the crucial role of green space for reducing the adverse impacts of urbanization in both aspects of urban ecosystem and socioeconomics [7–15]. The World Health Organization has identified that green spaces are innovative methods for enhancing the quality of urban environments via improvements of local resilience and promotions of sustainable lifestyles [15]. Turaga et al. [16] state that urban green spaces become a critical asset because they deliver various benefits including aesthetics enhancement, pollution reduction, positive effects on physical and mental health of citizens, urban heat island reduction, and groundwater recharge. Therefore, there is a raising societal support for protection and development of green space in urban areas.

Therefore, up-to-date spatial information regarding the current status of urban green space is crucial for urban land-use planning and management. This information has become increasingly difficult to obtain via conventional landscape surveying approaches since green spaces have been constantly modified, fragmented, and dispersed due to the fast pace of urbanization. Moreover, surveying tasks at a regional scale are daunting because of both time and labor consumptions required for field data acquisition, processing, and report. Thus, there is a pressing need for advanced methods to automate the green space surveying task.

Recently, medium-resolution imagery coupled with advanced machine learning methods has provided effective solution for urban landscape survey [17–22]. Remote sensing data used with geographic information system (GIS) can be used to generate thematic maps to assess green vegetation cover at a regional scale. Data extracted from such map can be helpful for further data analyzing processes regarding the size, shape, and other landscape pattern of urban green space [23–28].

Rafiee et al. [13] relies on Landsat Thematic Mapper images to study the patterns of green areas; this study employs combined techniques of remote sensing image classification, landscape metrics assessment, and vegetation indices. El Garouani et al. [29] employ the maximum likelihood supervised classification to analyze data obtained from Landsat's bands with a spatial resolution of 30 m; the authors investigate the relationship between urbanization and land use changes as well as the effect of the increase in impervious surface areas. Urban green space distribution has been modeled in [7] with the use of remote sensing, GIS technology, and normalized difference vegetation index. Do et al. [3] relies on Landsat 8 OLI (Operational Land Imager) image datasets provided the United States Geological Survey to study green space patterns; the support vector machine (SVM) has been used by the authors for the task of image data classification; the overall accuracy of the proposed machine learning model is 82.70%.

Li et al. [30] construct land-use and land-cover maps including green spaces using Landsat Operational Land Imager (OLI) and Enhanced Thematic Mapper Plus (ETM+) imagery; convolutional neural network, random forest, and SVM are the employed machine learning models used for image data classification; this study reports a classification accuracy of 84.40% on the testing dataset. Dinda et al. [31] construct an integrated model for studying urban growth and associated green space loss; the model relies on maximum likelihood classifier, artificial neural network, and SVM for performing pattern recognition task; the SVM model has attained the most desired classification accuracy and the area under the receiver operating characteristic curve (0.906).

It is noted that besides SVM, deep neural networks (DNNs) have also been successfully applied in remote sensing-based land-use classification [32–34]. DNNs are highly appropriate for image categorization due to its convolution operator based autonomous feature extraction phase [35]. However, successful implementations of DNNs often require a large number of training image samples. The

computational expense of DNNs is generally significant due to the time-consuming training process used to fine-tune the networks' weights. Moreover, because deep learning models have a quiet large number of hyperparameters (e.g., the number of hidden layers, the number and the size of convolution operations, the size of the pooling operations, etc.), the process of identifying a suitable network architecture can be tedious. Moreover, since the extracted features are represented as numerical data in this study, the application of SVM can be highly appropriate. It is because SVM has been proven to be a capable tool for classifying extracted numerical datasets [19, 36–39].

Based on literature review, there is an increasing trend of applying machine learning in remote sensing-based urban green space study. Since the problem of interest is challenging due to the involvement of multivariate and non-linear data analysis, other advanced machine learning solutions need to be investigated to improve the urban green space detection accuracy. Moreover, the current literature also points out that individual machine learning methods are the commonly employed approach. Hybrid machine learning models that harness advantages of various computational intelligence techniques are rarely investigated to construct urban green space detection models. Specifically, previous studies have mainly relied on the individual machine learning approach [3, 13, 29, 31], and the employment of metaheuristic algorithms used for optimizing machine learning based remote sensing data classification has rarely been proposed and investigated. Therefore, the original contribution of the current work is proposing a hybridization of SVM machine learning and metaheuristic optimization used for remote sensing-based urban green space detection.

SVM [40] is considered to be a capable pattern recognizer with excellent generalization capability. It is due to the fact that the model structure of this machine learning method is learnt via the framework of structural risk minimization which is resilient to overfitting and noisy data [41]. Nevertheless, the model construction phase of a SVM model requires a proper setting of its two hyperparameters including the penalty coefficient and the kernel function parameter. The former specifies the amount of penalty imposing on data samples having classification errors. The later determines the locality of the employed kernel function which directly influences the generalization of the constructed model.

The task of determining hyperparameters of a machine learning model is known as model selection [41] and can be modeled as an optimization problem. For the case of a SVM model, this is a challenging task because of several reasons. First, the landscape of the objective function is unknown and not differentiable. Second, the hyperparameters must be searched in continuous space; therefore, there is an indefinite number of feasible solutions. This fact means that an exhaustive search on the hyperparameters is infeasible. Therefore, various scholars have resorted to metaheuristic algorithms for dealing with the model selection problems. The role of metaheuristic algorithms in the task of hyperparameter setting (also called model selection) is indeed

crucial. These algorithms are used to optimize the performance of machine learning model to achieve a balance between model accuracy and model generalization.

The employed metaheuristic approaches include symbiotic organisms search [42], particle swarm optimization [43, 44], the forensic-based investigation optimization [45], equilibrium optimization [20], Harris hawks optimization [46], simulated annealing [47], social spider optimization [48, 49], gray wolf optimization [38, 50], teaching-learning-based algorithm [51], salp swarm algorithm [52, 53], artificial bee colony [54], pigeon-inspired optimization [55], cuckoo search optimization [56], imperialist competitive algorithm [57], moth flame optimization [58], and cuckoo search algorithm [59]. Those previous works have demonstrated the effectiveness of metaheuristic algorithms in optimizing machine learning models and solving complex tasks in various application domains.

Marine Predators Algorithm (MPA), first introduced in [60], is a recently proposed nature-inspired metaheuristic inspired from the foraging strategy of marine predators. This metaheuristic is characterized by a novel combination of Lévy and Brownian movements used for enhancing the optimization performance. The capability of MPA has been demonstrated by various optimization tasks [60]. Nevertheless, the performance of this metaheuristic used in optimizing machine learning models has rarely been investigated. Hence, this study proposes to hybridize MPA and SVM to form an integrated intelligent model used for remote sensing-based urban green space detection.

Remote sensing data obtained from Sentinel 2 satellite in the study area of Da Nang city is used to train and verify the MPA-SVM hybrid model. In this work, the MPA optimized SVM model trained by remote sensing data with all of the Sentinel 2's spectral bands is compared with the models that use commonly employed vegetation indices including normalized difference vegetation index (NDVI) [61], normalized difference water index (NDWI) [62], soil-adjusted vegetation index (SAVI) [63], and MERIS terrestrial chlorophyll index (MTCI) [64].

The rest of the article is organized as follows: Section 2 reviews the research methodology and material. Section 3 presents the proposed MPA optimized SVM used for remote sensing-based urban green space detection. Experimental results are reported in Section 4. Concluding remarks of the current study are summarized in Section 5.

2. Research Methodology and Material

2.1. General Description of the Study Area and Remote Sensing Data. As mentioned earlier, urban green spaces play a significant role in the urban living environment; they serve a variety of functions including climatic modification, aesthetics, recreation, and physical/mental health improvement. Nevertheless, due to the physical expansion of Da Nang city (Vietnam), certain areas of green spaces have been replaced by impervious surface such as buildings and roads. Therefore, the current status of urban green space in this city needs to be updated in a timely manner and this city has been selected as the study area of this research work.

Da Nang is a crucial coastal city located in Central Vietnam. Da Nang's location is at 15°55' to 16°14' North and 107°18' to 108°20' East [3]. It is the third largest city within the nation with a population of about 1 million. Da Nang urban center (refer to Figure 1) is located in the eastern section of the area and consists of six districts: Hai Chau, Cam Le, Thanh Khe, Lien Chieu, Ngu Hanh Son, and Son Tra [65]. The rural districts of Hoa Vang and Hoang Sa also belong to Da Nang city but are not included in the Da Nang urban center; therefore, these two rural districts are excluded from the study area.

To survey the urban green space status of Da Nang city, remote sensing data in form of spectral bands have been collected from Sentinel 2 on July 16, 2020. These spectral bands (see Table 1) are provided openly by USGS [66]; they can be processed and analyzed by Sentinel Application Platform (SNAP) software package [67] as well as ENVI software package [68]. Using the open-accessed tools of SNAP, the original Sentinel 2's spectral bands are converted to TIF format via the geometric operation of resampling. Moreover, it is noted that the used map projection of the obtained images is Universal Transverse Mercator (UTM) within Zone 48N-Datum World Geodetic System (WGS) 84. Images of the Sentinel-2 spectral bands are demonstrated in Figure 2. This figure demonstrates the 13 spectral bands obtained from the Sentinel-2 on July 16, 2020. These bands are coastal aerosol, blue, green, red, red-edge 1, red-edge 2, red-edge, near infrared, near infrared narrow, water vapour, shortwave infrared/cirrus, shortwave infrared 1, and shortwave infrared 2. The wavelength range and resolution of each spectral band are provided in Table 1.

2.2. Remote Sensing-Based Vegetation Indices. In remote sensing field, vegetation indices have been widely used to extract vegetation biophysical information from satellite image data [69]. Previous works have demonstrated the effectiveness of vegetation indices in remote sensing-based green space mapping [7, 13, 70–72]. Therefore, this study relies on such conventional indices as a means of urban green space detection. The employed vegetation indices include normalized difference vegetation index (NDVI) [61], soil-adjusted vegetation index (SAVI) [63], normalized difference water index (NDWI) [62], and MERIS terrestrial chlorophyll index (MTCI) [64]. These indices can be computed as follows [61, 69, 73]:

$$\text{NDVI} = \frac{\rho_{\text{nir}} - \rho_{\text{red}}}{\rho_{\text{nir}} + \rho_{\text{red}}}, \quad (1)$$

where ρ_{red} and ρ_{nir} represent the red reflected radiant flux (Sentinel 2's band 4) and near-infrared radiant flux (Sentinel 2's band 8a), respectively.

$$\text{SAVI} = \frac{(\rho_{\text{nir}} - \rho_{\text{red}})}{(\rho_{\text{nir}} + \rho_{\text{red}} + L)} (1 + L), \quad (2)$$

where $L = 0.5$ denotes the soil brightness correction factor [74].

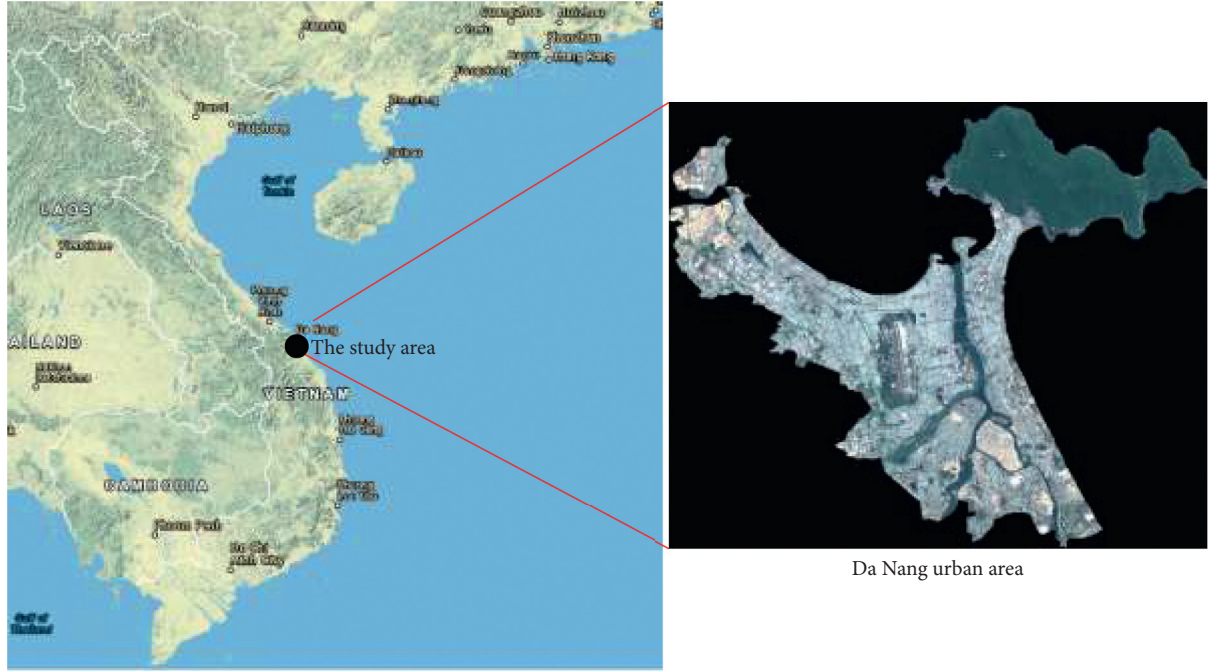


FIGURE 1: Da Nang urban area.

TABLE 1: The Sentinel-2 spectral bands.

Band number	Description	Wavelength range (nm)	Resolution (m)
B1	Coastal aerosol	433–453	60
B2	Blue	458–523	10
B3	Green	543–578	10
B4	Red	650–680	10
B5	Red-edge 1	698–713	20
B6	Red-edge 2	733–748	20
B7	Red-edge	773–793	20
B8	Near infrared (NIR)	785–900	10
B8a	Near infrared narrow (NIRn)	855–875	20
B9	Water vapour	935–955	60
B10	Shortwave infrared/Cirrus	1360–1390	60
B11	Shortwave infrared 1 (SWIR1)	1565–1655	20
B12	Shortwave infrared 2 (SWIR2)	2100–2280	20

$$NDWI = \frac{\rho_{nir} - \rho_{swir}}{\rho_{nir} + \rho_{swir}}, \quad (3)$$

where ρ_{red} and ρ_{swir} represent the near-infrared radiant flux (Sentinel 2's band 8a) and shortwave infrared (Sentinel 2's band 11), respectively.

$$MTCI = \frac{B6 - B5}{B5 - B4}, \quad (4)$$

where ρ_{red} is Sentinel 2's B4; ρ_{red_edge} is Sentinel 2's B5; ρ_{nir} is Sentinel 2's B6 [74, 75].

2.3. The Used Metaheuristic and Machine Learning Approaches

2.3.1. Marine Predators Algorithm. Marine Predators Algorithm (MPA), first introduced in [60], is a stochastic

global optimization algorithm inspired from the widespread foraging strategy of marine species such as sharks and tunas. The foraging strategy of these marine species effectively utilizes Lévy and Brownian movements along with optimal encounter rate policy in biological interaction between predator and prey [76–78].

The searching process of MPA consists of three phases considering three scenarios: (i) high velocity ratio when a prey is moving faster than a predator, (ii) unit velocity ratio when the rates of movement of a prey and a predator are similar, and (iii) low velocity ratio when the rate of movement of a predator is higher than that of a prey. The searching operation of the MPA metaheuristic is demonstrated in Figure 3. Let XE be the position of predators and XP be the position of preys within a marine ecosystem. The 1st phase aims at search space exploration and is applied for the first one-third of the searching iteration number; the

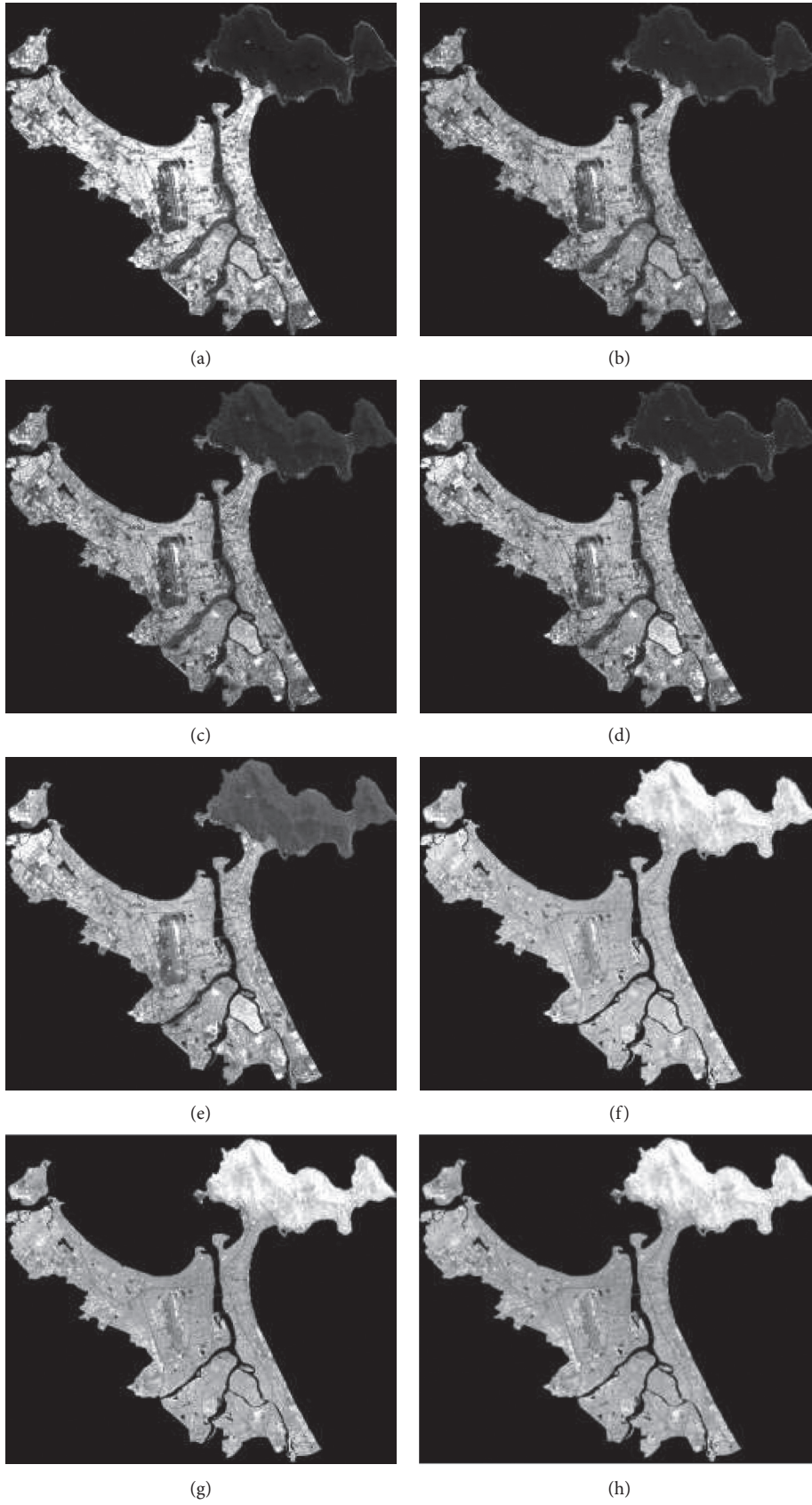


FIGURE 2: Continued.

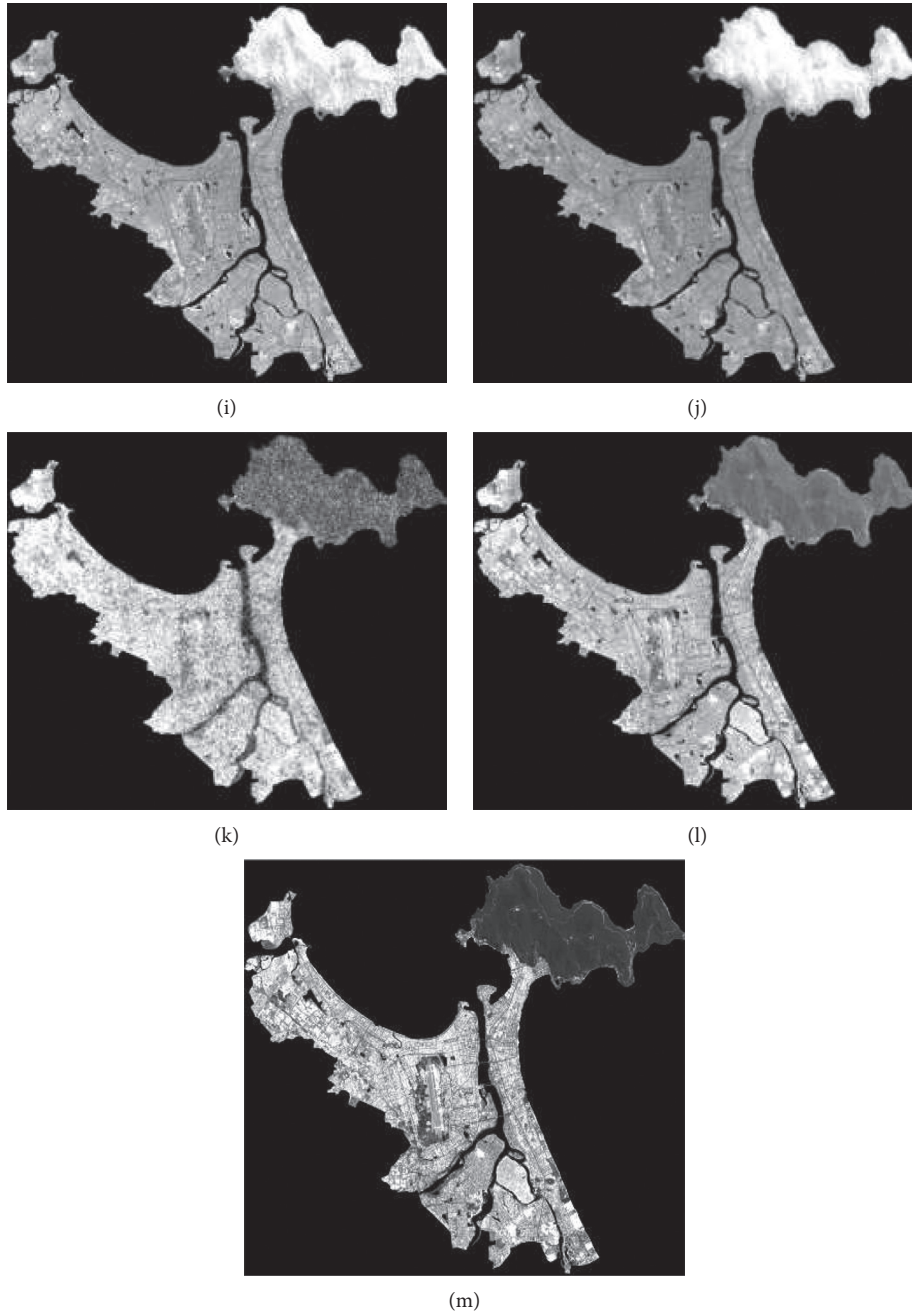


FIGURE 2: Gray-scale image demonstrated the Sentinel 2's spectral bands: (a) B1, (b) B2, (c) B3, (d) B4, (e) B5, (f) B6, (g) B7, (h) B8, (i) B8a, (j) B9, (k) B10, (l) B11, and (m) B12.

mathematical equation used to revise the prey position is given by

$$XP_i = XP_i + 0.5 \otimes R \otimes R_B \otimes (XE_i - R_B \otimes XP_i), \quad (5)$$

$$i = 1, 2, \dots, NP,$$

where \otimes is an entry-wise multiplication operator. R_B is a vector including random numbers generated from a normal distribution which mimics the Brownian motion. i denotes the index of population members. R represents a vector of

uniform random number within $[0, 1]$. NP is the number of population members.

The 2nd phase serves as an intermediate phase and occurs within the second one-third of the searching iteration number. The positions of the first half of the population members are updated as follows:

$$XP_i = XP_i + 0.5 \otimes R \otimes R_L \otimes (XE_i - R_L \otimes XP_i), \quad (6)$$

$$i = 1, 2, \dots, \frac{NP}{2},$$

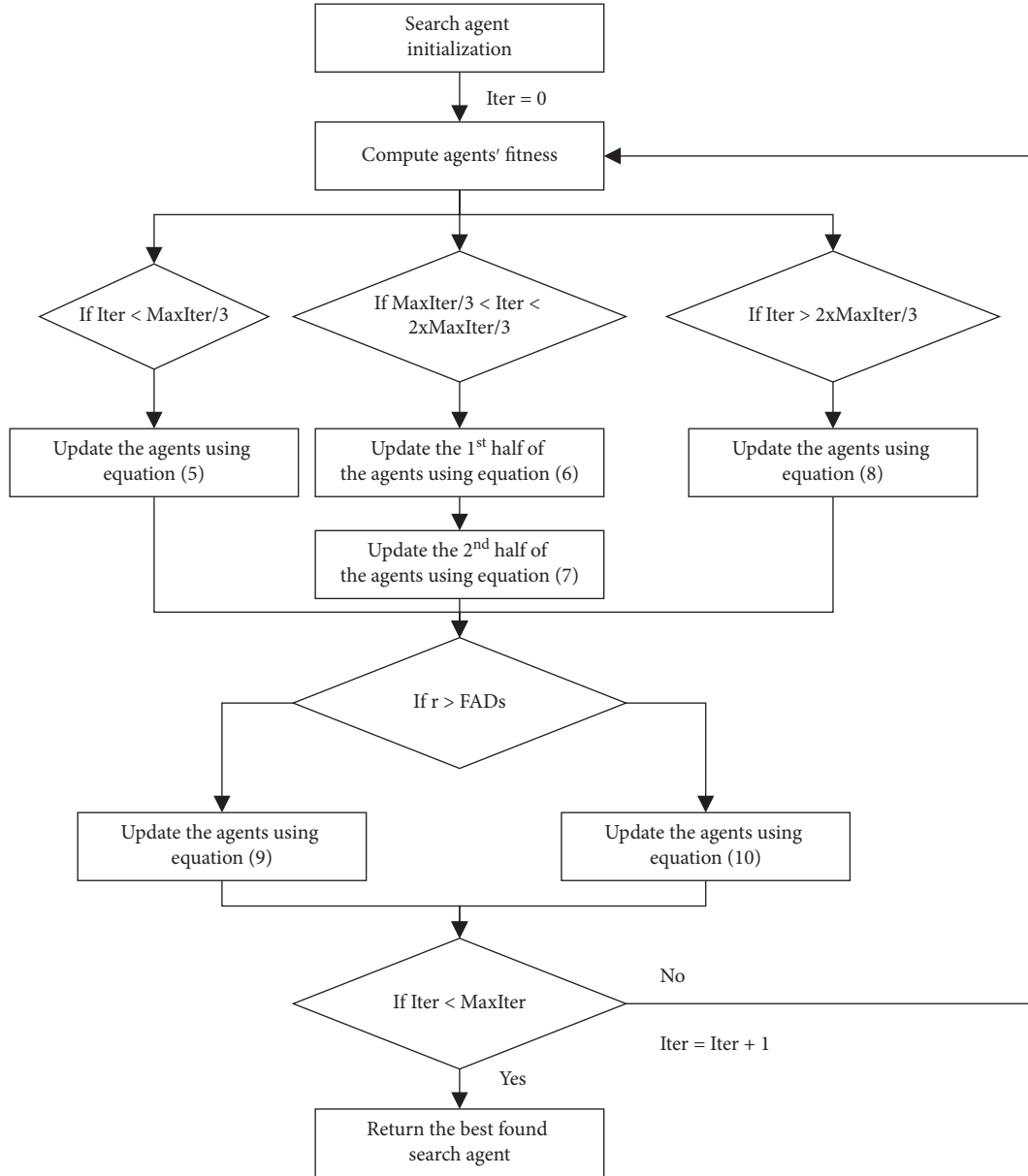


FIGURE 3: Flowchart of the MPA algorithm.

where R_L denotes a vector of random numbers generated from the Lévy distribution which represents the Lévy movement.

The positions of the second half of the population members are updated as follows:

$$XP_i = XE_i + 0.5 \times CF \otimes R_B \otimes (R_B \otimes XE_i - XP_i),$$

$$i = \frac{NP}{2}, \frac{NP}{2} + 1, \dots, NP, \quad (7)$$

where $CF = (1 - (\text{Iter}/\text{MaxIter}))^{2(\text{Iter}/\text{MaxIter})}$; Iter and MaxIter are the current iteration count and the maximum number of iterations, respectively.

The last phase of the optimization process aims at exploitation of the search space. The population members' positions are updated in the following equation:

$$XP_i = XE_i + 0.5 \times CF \otimes R_L \otimes (R_L \otimes XE_i - XP_i),$$

$$i = 1, 2, \dots, NP. \quad (8)$$

In addition, to model behavior shift in marine predators according to the eddy formation or Fish Aggregating Devices (FADs) effects [79], the MPA metaheuristic employs the following operation:

$$XP_i = XP_i + CF[LB + R \otimes (UB - LB)] \otimes U \text{ if } r \leq \text{FADs},$$

$$XP_i = XP_i + [\text{FADs}(1 - r) + r][XP_{r_1} - XP_{r_2}] \text{ if } r > \text{FADs}, \quad (9)$$

where $\text{FADs} = 0.2$ denotes the probability of the FADs effect. U represents a random binary vector. r is a random number within $[0, 1]$. LB and UB are vectors of lower and upper

boundaries of the searched variables. $r1$ and $r2$ denote two random indices.

2.3.2. Support Vector Machine. Introduced by Vapnik [40], support vector machines (SVM) have gained attentions of the academic community and have become a preeminent pattern recognition approach [55, 80–90]. Given a data sample set S drawn from a data universe X_U , a hidden target function $f: X \rightarrow \{0, 1\}$, we first create a labeled training dataset D , where $D = \{(x, y) | x \in S \text{ and } y = f(x)\}$. The SVM machine learning can be used to estimate the target function $f(x)$ by constructing a function $\hat{f}(x): X \rightarrow \{0, 1\}$ based on data samples stored in D so that $\hat{f}(x) \cong f(x)$ for all x in X . Herein, for the task of urban green space detection, the data label can be modeled as “0” = nongreen space (the negative class) and “1” = green space (the positive class). The input data X are properties of the Sentinel 2’s spectral bands.

To construct a SVM model, it is required to solve the following constrained optimization problem [91]:

$$\text{Minimize } J_p(w, e) = \frac{1}{2}w^T w + C \frac{1}{2} \sum_{k=1}^N e_k^2,$$

$$\text{Subjected to } y_k(w^T \varphi(x_k) + b) \geq 1 - e_k \quad k = 1, \dots, N, \quad e_k \geq 0,$$
(10)

where $w \in R^n$ and $b \in R$ are used to construct a classification hyperplane used for pattern classification. e is the vector of slack variables. C denotes the penalty coefficient. $\varphi(x)$ denotes a nonlinear data mapping used for dealing with data that cannot be linearly separated.

One advantage of the SVM method is that the explicit formula of $\varphi(x)$ is not required. To construct a SVM model, only the dot product of $\varphi(x)$ is necessary. The dot product of two data samples x_k and x_l is represented as a kernel function $K(x_k, x_l)$:

$$K(x_k, x_l) = \varphi(x_k)^T \varphi(x_l). \quad (11)$$

For multivariate and nonlinear data classification problem, the radial basis kernel function (RBKF) is commonly utilized:

$$K(x_k, x_l) = \exp\left(-\frac{\|x_k - x_l\|^2}{2\sigma^2}\right), \quad (12)$$

where σ denotes a hyperparameter of the RBKF.

By solving a Lagrangian dual of the aforementioned constrained optimization problem and using a quadratic programming solver, the SVM model used for data classification can be expressed compactly as follows [91]:

$$y(x_l) = \text{sign}\left(\sum_{k=1}^{SV} \alpha_k y_k K(x_k, x_l) + b\right), \quad (13)$$

where α_k represents the solution of the optimization problem; SV denotes the number of support vectors.

3. The Proposed Marine Predators Algorithm Optimized Machine Learning Approach for Urban Green Space Detection

This section of the article is dedicated to describing the integrated model used for remote sensing-based urban green space detection. The core of the proposed model is a hybridization of the MPA metaheuristic and the SVM machine learning. These two methods work synergistically to analyze patterns hidden in a set of remotely sensed data collected for the study area of Da Nang urban center. In detail, SVM is used to construct a decision boundary that separates the input data space into two distinctive regions of “nongreen space” and “green space”.

To further enhance the performance of the SVM model, MPA is utilized to autonomously fine-tune the SVM training process by identifying a set of appropriate model hyperparameters. The optimized hyperparameters include the penalty coefficient and the RBKF parameter. In this study, the searching range of the penalty coefficient is [1, 100]; the searching range of the RBKF parameter is [0.1, 100].

These two hyperparameters strongly influence the learning and the predictive capability of the integrated urban green space detection model. A too large penalty coefficient or a too small RBKF parameter leads to overfitted models. On the other hand, a too small penalty coefficient and a too large RBKF parameter tends to construct underfitted models [92]. Therefore, the role of the MPA is to find a set of the penalty coefficient and the RBKF parameter which features a balance between predictive accuracy and modeling generalization. Accordingly, it is expected that the constructed model will not suffer from either overfitting or underfitting.

The overall model structure is presented in Figure 4. The 13 spectral bands are obtained from the Sentinel 2 satellite and processed via SNAP [67] and ENVI [68] software packages. To accelerate the computing process, the images of spectral bands are divided into blocks with the size 5×5 pixels. For the purpose of data classification, the average (μ_c) and the standard deviation (σ_c) of gray intensity of each image block are computed and used as numerical features by the integrated MPA-SVM model.

The average and standard deviation of gray intensity are given by [93]

$$\mu_c = \sum_{i=0}^{NL-1} I_{i,c} \times P_c(I),$$

$$\sigma_c = \sqrt{\sum_{i=0}^{NL-1} (I_{i,c} - \mu_c)^2 \times P_c(I)}, \quad (14)$$

where $I_{i,c} = 0, 1, 2, \dots, 255$. $NL = 256$ represents the number of discrete color values. $P(I)$ is the first-order histogram of an image block [94].

To construct the integrated MPA-SVM model for urban green space detection, it is necessary to prepare a training dataset with assigned ground truth labels. This study has

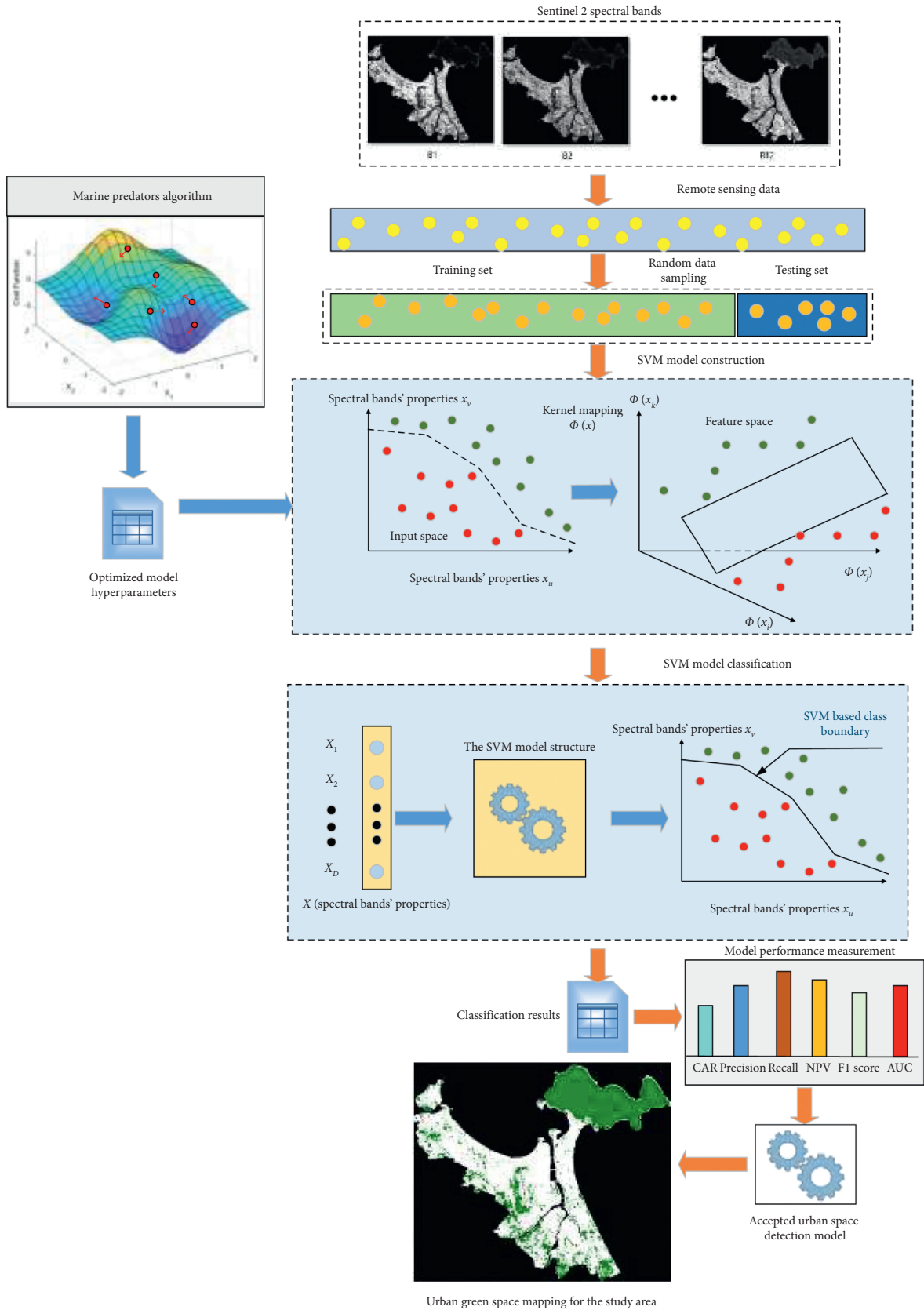


FIGURE 4: The proposed MPA optimized SVM for urban green space detection.

performed sampling process to collect data in the nongreen space and green space areas within the study area (demonstrated in Figure 5). It is noted that the ground truth label of each image data in this study has been verified by field trips and Google Earth Engine. Each block is sampled with the size of 25×25 pixels to generate nonoverlapped image patches with the size of 5×5 pixels. After the data sampling process, there are 1,000 image patches available for the feature extraction operator. Herein, each class label contains 500 image patches to guarantee a balanced classification. The collected dataset is illustrated in Table 2. Notably, each spectral band yields two statistical measurements (i.e., the mean and standard deviation) and there are 13 bands. Thus, the total number of features used for classification is $13 \times 2 = 26$.

It is worth noticing that the extracted dataset including the input features which characterize statistical properties of the spectral bands and the corresponding class labels has been randomly separated into a training (70%) dataset and a testing dataset (30%) [95]. The first set is used for model training and the second set is used to inspect the model predictive capability. In addition, to standardize the input data range, the Z-score equation is used as follows [96]:

$$X_Z = \frac{X_D - M_X}{STD_X}, \quad (15)$$

where X_Z and X_D denote the normalized and the original features, respectively. M_X and STD_X denote the mean value and the standard deviation of the features, respectively.

To optimize the SVM model used for urban green space detection, the objective function of the MPA metaheuristic has employed a 5-fold cross validation process and the indices of false negative rate (FNR) and false positive rate (FPR). This objective function (OF) is described as follows [96]:

$$OF = \frac{\sum_{k=1}^5 (FNR_k + FPR_k)}{5}, \quad (16)$$

where FNR_k and FPR_k denote FNR and FPR computed in the k th run, respectively.

The FNR and FPR indices are given by [96]

$$\begin{aligned} FNR &= \frac{FN}{FN + TP}, \\ FPR &= \frac{FP}{FP + TN}, \end{aligned} \quad (17)$$

where FN, FP, TP, and TN are the false negative, false positive, true positive, and true negative data samples, respectively.

In this study, the source code of the MPA metaheuristic is provided by Faramarzi et al. [60]. For the purpose of model optimization, the integrated MPA-SVM has been constructed in MATLAB. The SVM model with an optimized set of hyperparameters is then developed in Visual C#

.NET framework 4.7.2 to process and analyze remote sensing data. The SVM model in Visual C# .NET has been built with functions provided by the Accord.NET Framework [97]. Moreover, the program has been implemented with the ASUS FX705GE-EW165T (Core i7 8750H and 8 GB Ram) platform.

4. Experimental Results and Discussion

In this section, a set of performance measurement indices is used to express the model predictive accuracy. This set includes classification accuracy rate (CAR), precision, recall, negative predictive value (NPV), F1 score, and area under the receiver operating characteristic curve (AUC) [98, 99]. The calculation of AUC is described in [99]. The formulas used to compute CAR, precision, recall, NPV, and F1 score are given by

$$CAR = \frac{N_C}{N_A} 100\%, \quad (18)$$

where N_C and N_A are the numbers of correctly predicted data and the total number of data, respectively.

$$\begin{aligned} \text{Precision} &= \frac{TP}{TP + FP}, \\ \text{Recall} &= \frac{TP}{TP + FN}, \\ \text{NVP} &= \frac{TN}{TN + FN}, \\ \text{F1 Score} &= \frac{2TP}{2TP + FP + FN}. \end{aligned} \quad (19)$$

Besides the MPA-SVM model which utilizes information provided by 13 spectral bands, this study has employed the MPA-SVM models using the aforementioned vegetation indices as benchmark models. The MPA-SVM employing all of the Sentinel 2's bands is denoted as MPA-SVM-13B. The benchmark models that use the NDWI, NDVI, SAVI, and MTCI are denoted as MPA-SVM-NDWI, MPA-SVM-NDVI, MPA-SVM-SAVI, and MPA-SVM-MTCI, respectively. The MPA-SVM-13B utilizes the statistical information obtained from all of the 13 spectral bands (i.e., the mean and the standard deviation of each band). Meanwhile, the MPA-SVM-NDWI, MPA-SVM-NDVI, MPA-SVM-SAVI, and MPA-SVM-MTCI employ the statistical information of the vegetation indices of NDWI, NDVI, SAVI, and MTCI, respectively. Therefore, the feature extraction phase of the benchmark models is similar to that of the MPA-SVM-13B. This feature extraction phase also computes the two indices of mean and standard deviation of image patches. The model optimization processes of the constructed models are demonstrated in Figure 6. Herein, the maximum number of searching iteration (MaxIter) of the MPA metaheuristic has been set to be 100; the number of population members (NP) is fixed to be 20. The detailed

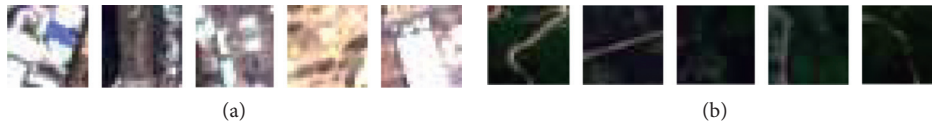
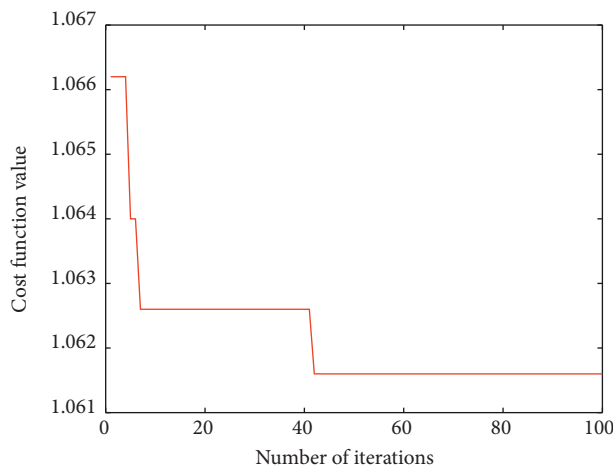


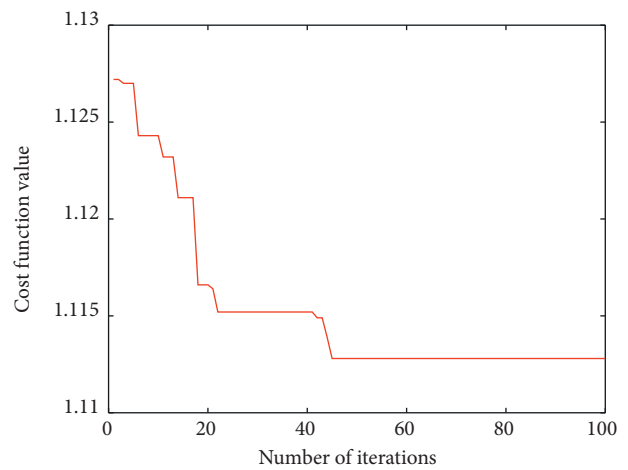
FIGURE 5: Demonstrations of the collected image samples using natural color composite: (a) nongreen space class and (b) green space class.

TABLE 2: Demonstration of the extracted dataset.

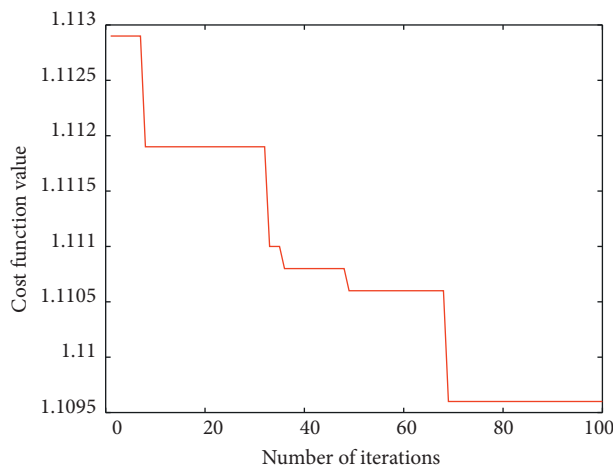
Sample	X1	X2	X3	X4	X5	...	X21	X22	X23	X25	X26	Y
0	165.96	152.68	145.48	168.60	168.72	15.58	13.33	5.21	35.76	20.36	20.32	0
1	152.96	101.28	95.32	138.36	158.08	17.27	7.10	9.08	31.78	12.11	16.31	0
2	133.12	90.92	83.68	143.56	160.76	14.02	11.95	6.57	21.24	29.14	31.10	0
3	122.40	91.84	84.52	140.20	168.60	12.01	5.77	3.92	18.13	22.55	26.75	0
4	119.40	90.52	76.00	104.08	124.00	13.80	11.45	9.80	0.00	21.92	21.31	0
...
996	23.08	24.68	25.76	26.20	67.60	16.22	6.67	11.00	21.87	10.96	7.93	1
997	25.16	25.56	31.16	25.64	69.84	16.18	9.55	9.90	18.37	6.81	3.75	1
998	25.24	30.76	30.16	35.68	72.16	8.07	5.14	8.14	18.49	7.19	5.39	1
999	16.80	19.52	21.00	18.12	63.16	14.17	16.83	5.39	18.13	12.39	6.39	1
1000	19.20	20.80	22.00	19.68	58.92	15.80	12.60	0.98	0.00	3.04	1.51	1



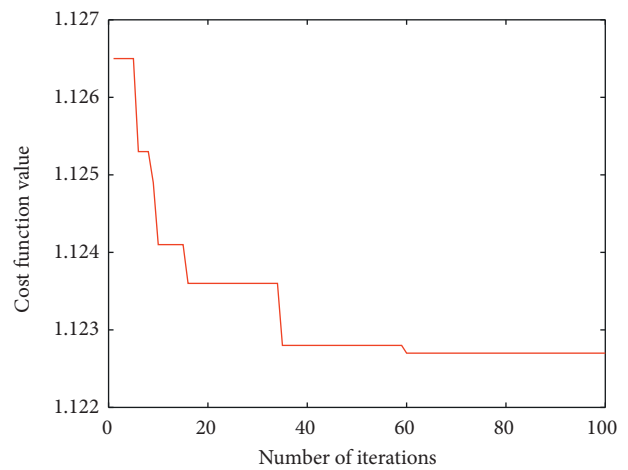
(a)



(b)



(c)



(d)

FIGURE 6: Continued.

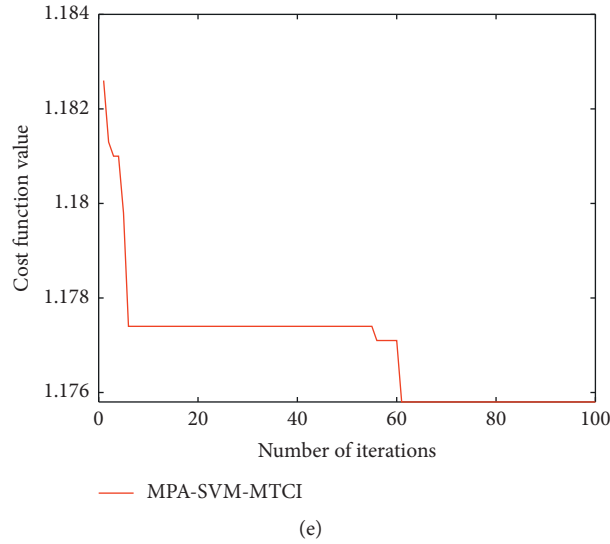


FIGURE 6: Optimization process of the MPA metaheuristic: (a) MPA-SVM-13B, (b) MPA-SVM-NDWI, (c) MPA-SVM-NDVI, (e) MPA-SVM-SAVI, and (e) MPA-SVM-MTCI.

TABLE 3: MPA-based optimization results.

Hyperparameters	MPA-SVM-13B	MPA-SVM-NDWI	MPA-SVM-NDVI	MPA-SVM-SAVI	MPA-SVM-MTCI
Penalty coefficient	79.782	3.938	89.886	9.605	32.495
RBFK	10.244	0.438	4.065	1.423	1.622

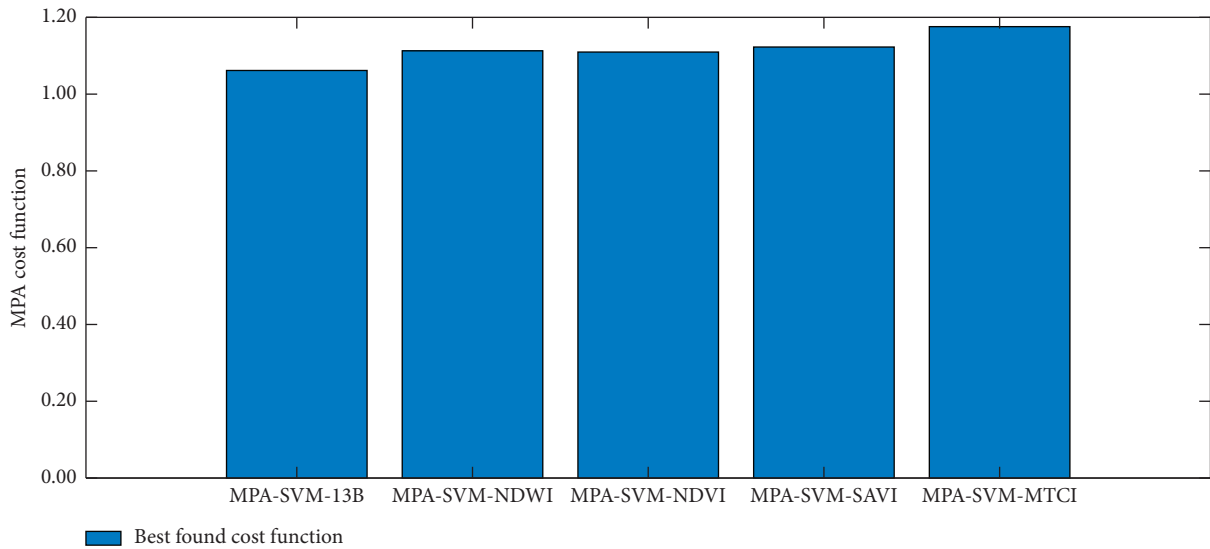


FIGURE 7: Cost function values optimized by MPA.

optimization results are reported in Table 3 which shows the best penalty coefficient and RBFK parameters for each model used for urban green space detection. In addition, the best found cost function values of the MPA-SVM-13B, MPA-SVM-NDWI, MPA-SVM-NDVI, MPA-SVM-SAVI, and

MPA-SVM-MTCI are provided in Figure 7. It can be observed from Figure 7 that the MPA-SVM using all of the 13 bands results in the lowest value of cost function.

As stated earlier, the constructed dataset has been randomly divided into a training set (70%) and a testing set

TABLE 4: Experimental result comparison.

Phase	Metrics	MPA-SVM-13B		MPA-SVM-NDWI		MPA-SVM-NDVI		MPA-SVM-SAVI		MPA-SVM-MTCI	
		Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
Training	CAR (%)	95.429	0.586	89.564	0.516	89.514	0.755	88.929	0.439	84.686	0.785
	Precision	0.941	0.009	0.882	0.008	0.885	0.011	0.881	0.008	0.819	0.009
	Recall	0.970	0.005	0.914	0.006	0.907	0.013	0.901	0.009	0.894	0.015
	NPV	0.968	0.005	0.911	0.007	0.905	0.011	0.898	0.007	0.882	0.015
	F1 score	0.955	0.006	0.897	0.006	0.896	0.008	0.891	0.005	0.855	0.008
	AUC	0.991	0.002	0.940	0.007	0.943	0.006	0.936	0.006	0.916	0.006
Testing	CAR (%)	93.100	1.411	89.400	1.356	89.300	1.729	88.983	1.062	83.850	1.698
	Precision	0.916	0.024	0.881	0.024	0.894	0.029	0.886	0.017	0.803	0.022
	Recall	0.947	0.018	0.911	0.017	0.895	0.020	0.893	0.023	0.888	0.030
	NPV	0.947	0.019	0.908	0.020	0.893	0.022	0.895	0.017	0.881	0.031
	F1 score	0.931	0.014	0.896	0.013	0.894	0.017	0.889	0.012	0.843	0.018
	AUC	0.979	0.006	0.926	0.014	0.946	0.013	0.937	0.012	0.910	0.016

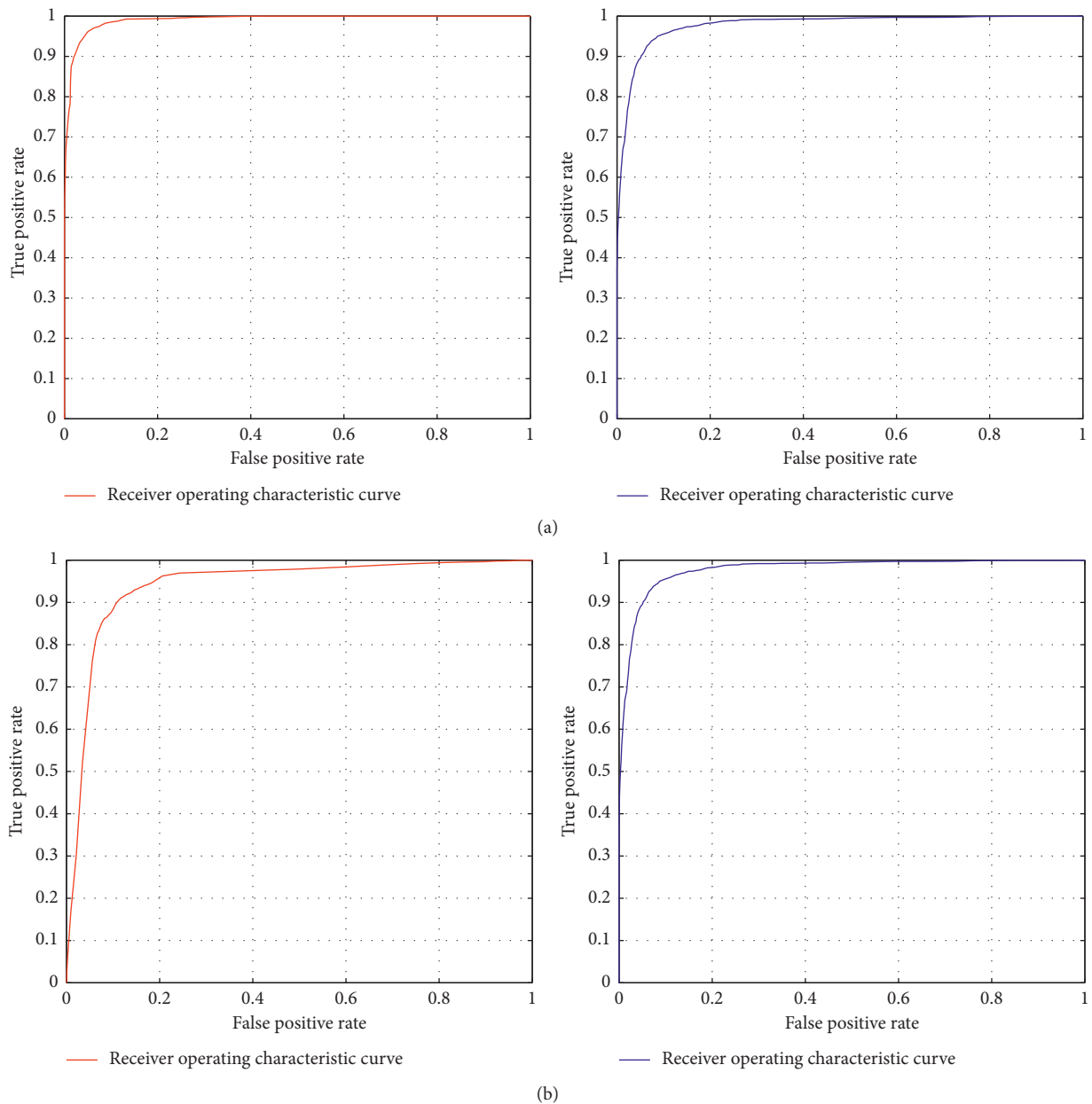
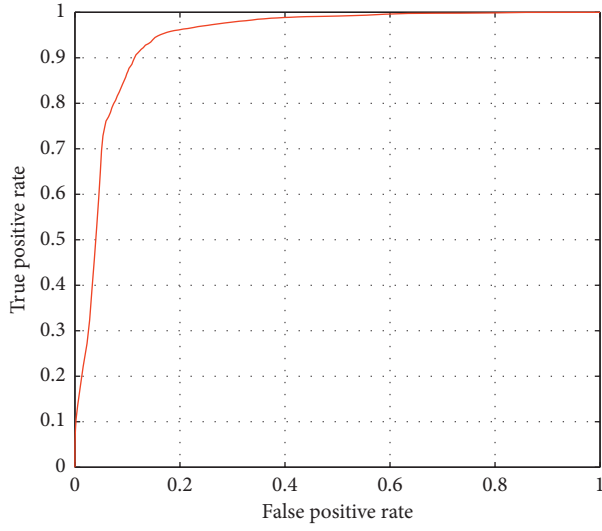
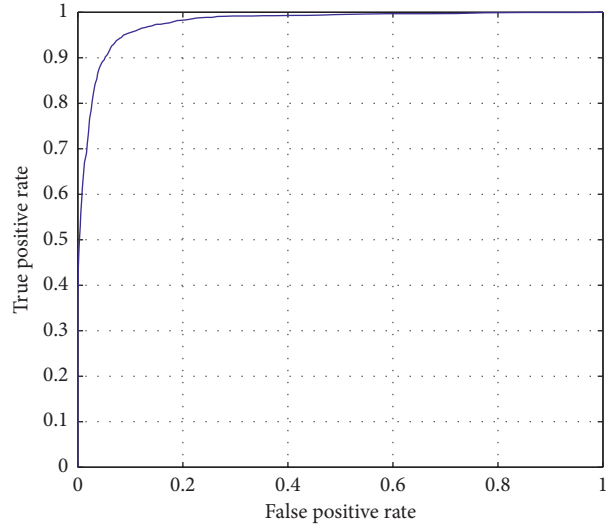


FIGURE 8: Continued.

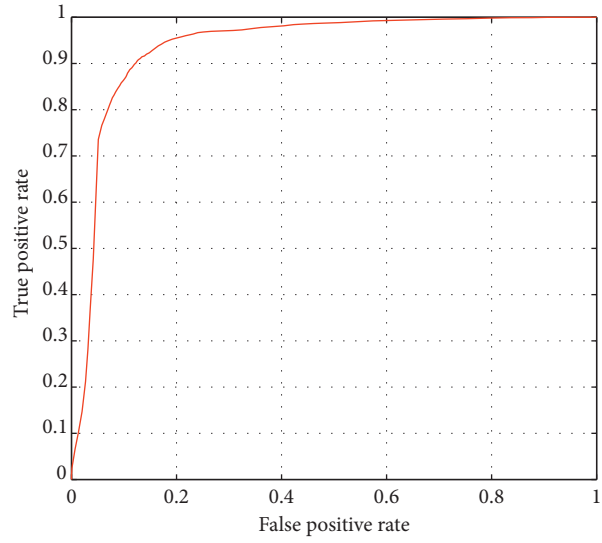


— Receiver operating characteristic curve

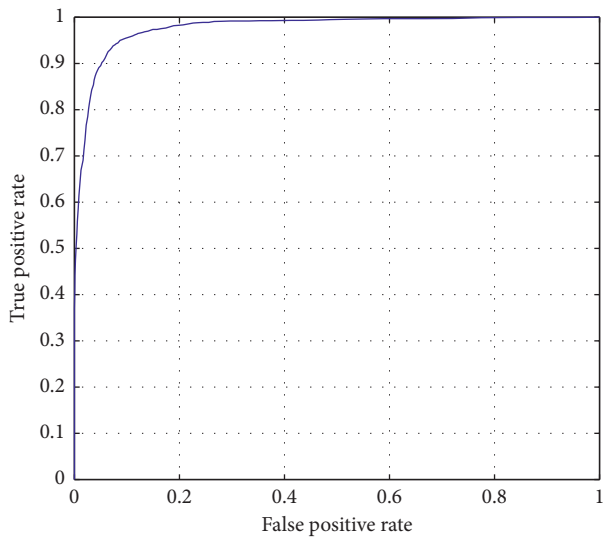


— Receiver operating characteristic curve

(c)



— Receiver operating characteristic curve



— Receiver operating characteristic curve

(d)

FIGURE 8: Continued.

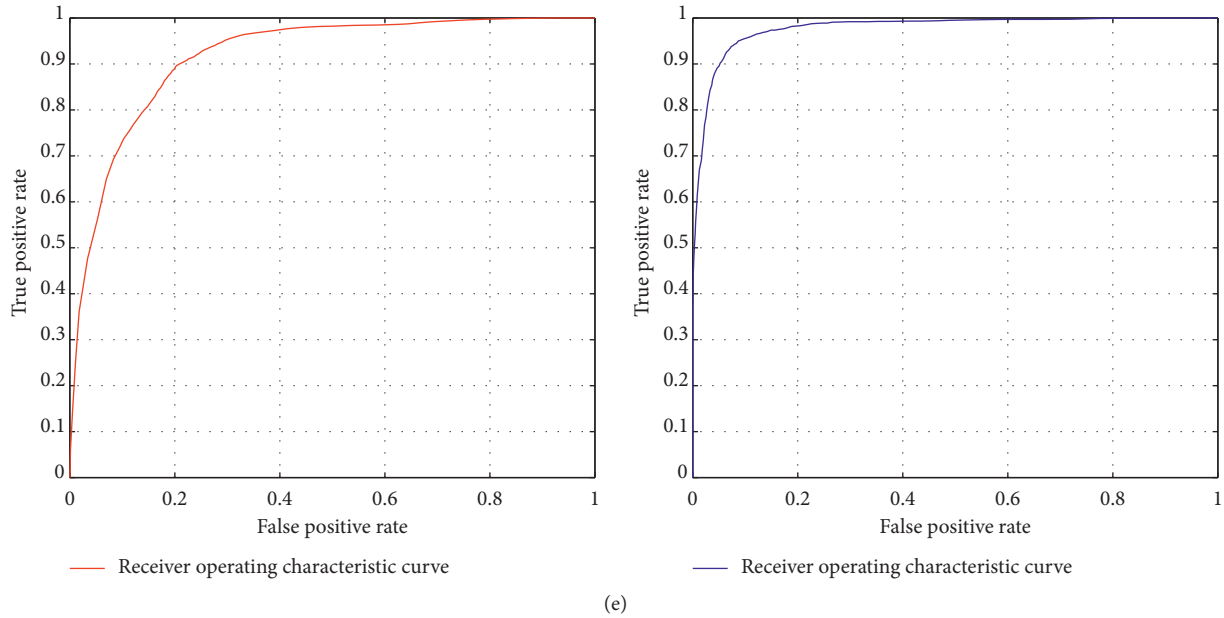


FIGURE 8: ROCs of the prediction models: (a) MPA-SVM-13B, (b) MPA-SVM-NDWI, (c) MPA-SVM-NDVI, (d) MPA-SVM-SAVI, and (e) MPA-SVM-MTCI.

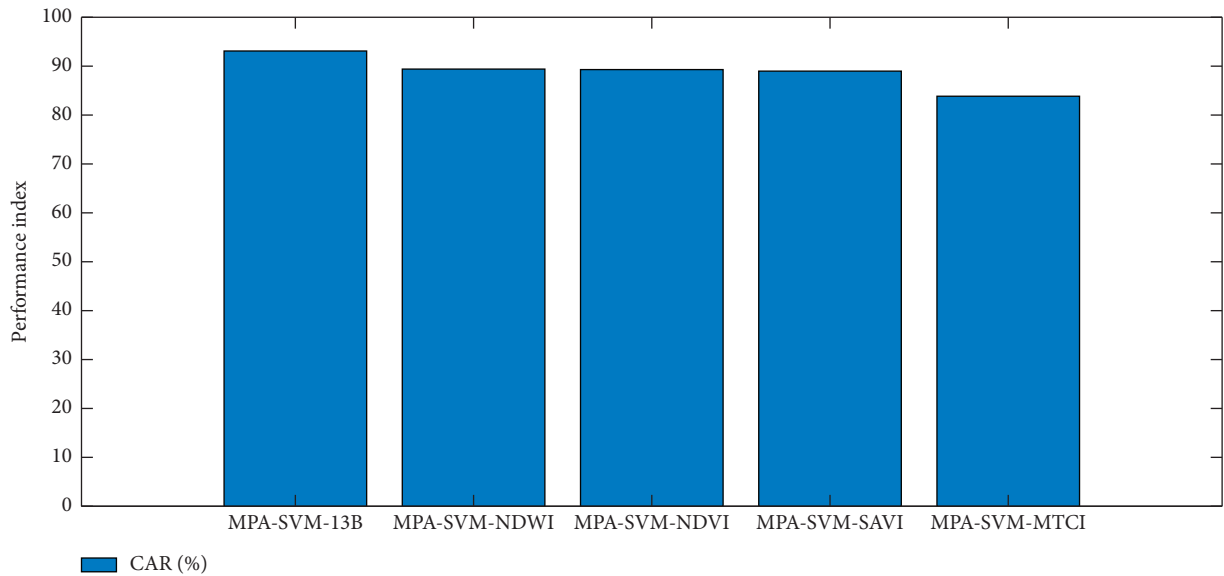


FIGURE 9: Result comparison in terms of CAR.

(30%). The first set is used for model training and the second set is reserved for model validation. Moreover, in order to reliably evaluate the model predictive performance, this study has repeated the model training and prediction processes 20 times. It is noted that the training and testing datasets are resampled in each run. The statistical

measurements obtained from this multiple model construction and validation phases are used for model assessment. This repeated process aims at diminishing the variation caused by the randomness in data sampling. The model prediction outcomes are summarized in Table 4 which shows the mean and standard deviation (Std) of

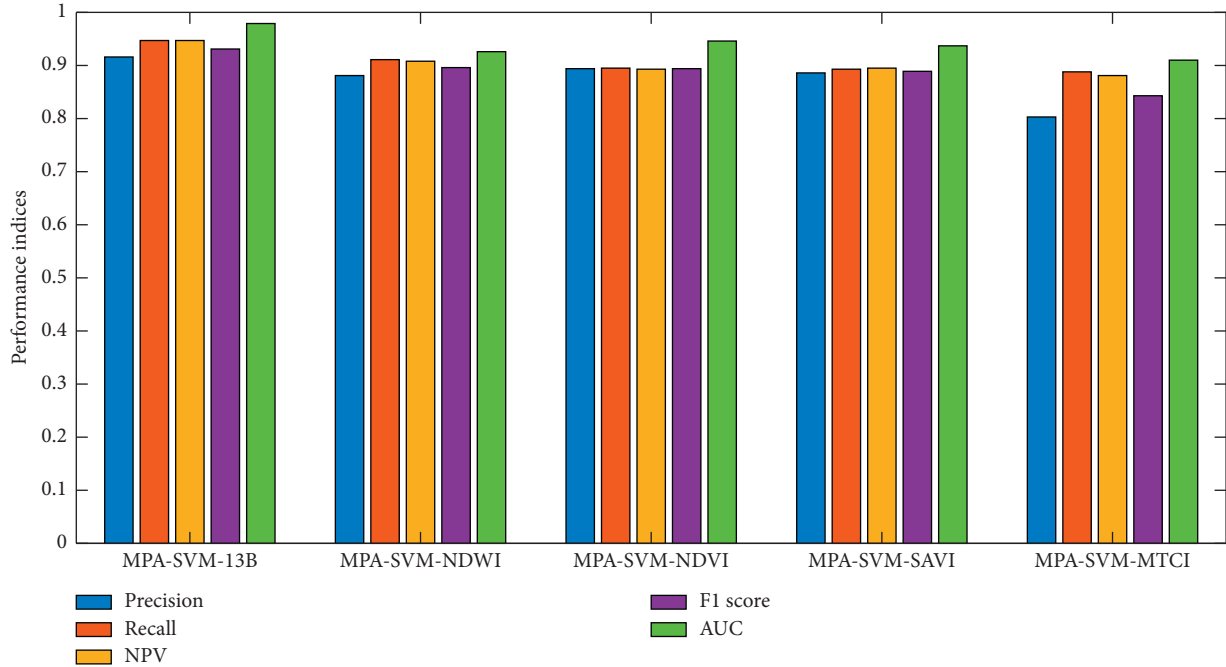


FIGURE 10: Result comparison in terms of precision, recall, NPV, F1 score, and AUC.

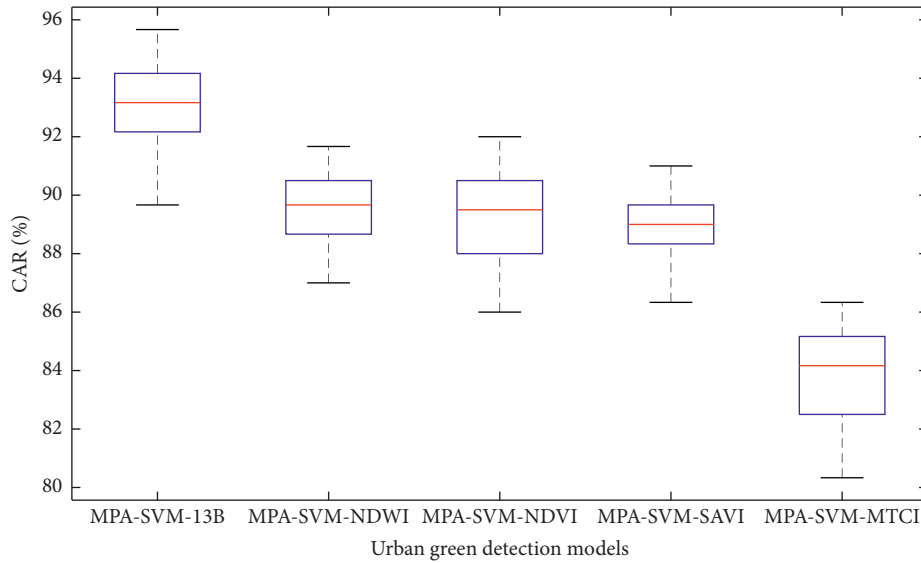


FIGURE 11: Box plot of CAR values obtained from the employed machine learning models.

the employed performance measurement indices. It can be observed that the MPA-SVM-13B with CAR=93.100%, precision=0.916, recall=0.947, NPV=0.947, F1 score=0.931, and AUC=0.979 has obtained the most desired urban green space detection performance. Compared with the proposed method, MPA-SVM-NDWI has gained a lower

CAR (89.400%) and F1 score (0.896), followed by MPA-SVM-NDVI (CAR= 89.300% and F1 score=0.894), MPA-SVM-SAVI (88.983% and F1 score=0.889), and MPA-SVM-MTCI (83.850% and F1 score=0.843).

In terms of AUC score, MPA-SVM-13B is the best model (AUC=0.979), followed by MPA-SVM-NDVI (AUC=

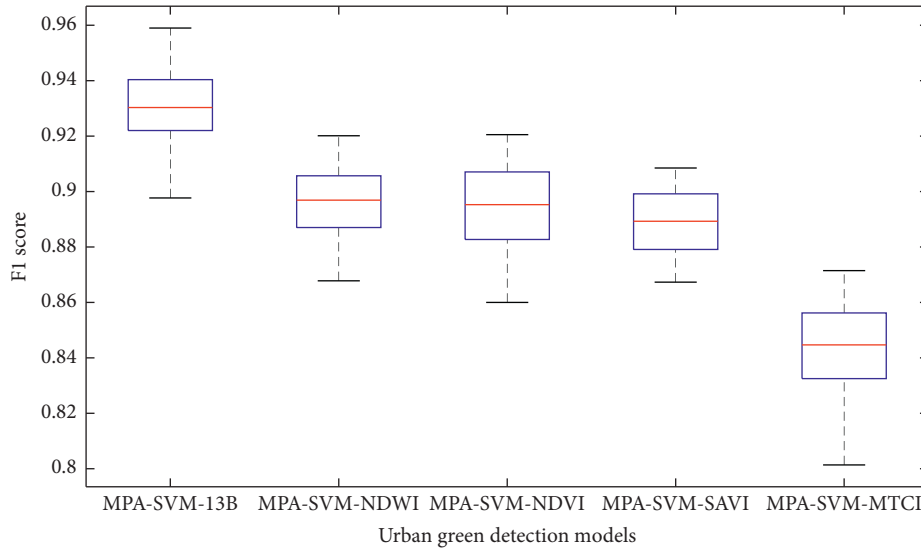


FIGURE 12: Box plot of F1 score values obtained from the employed machine learning models.

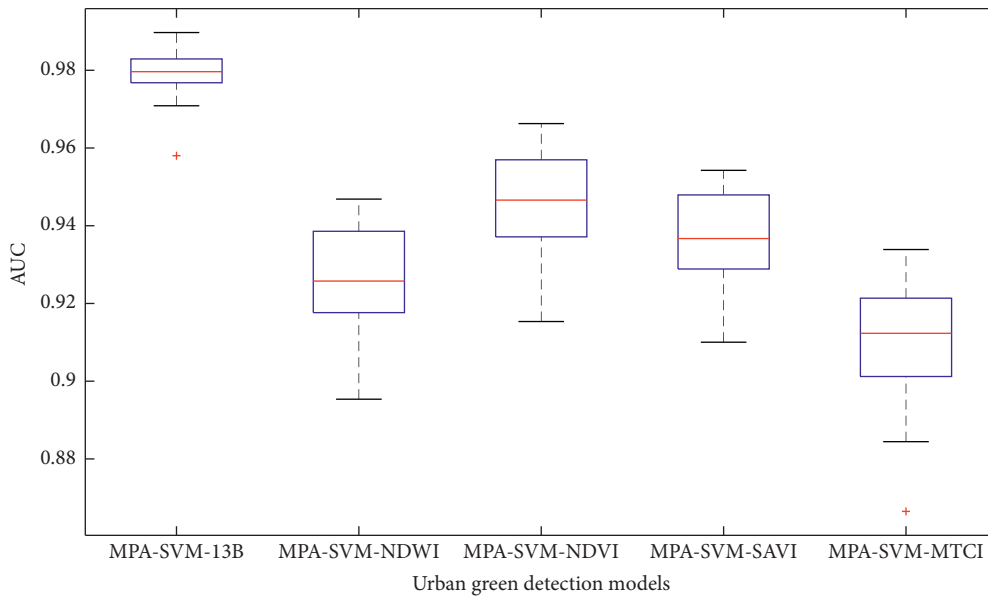


FIGURE 13: Box plot of AUC values obtained from the employed machine learning models.

TABLE 5: Wilcoxon signed rank test results with CAR index.

	MPA-SVM-13B	MPA-SVM-NDWI	MPA-SVM-NDVI	MPA-SVM-SAVI	MPA-SVM-MTCI
MPA-SVM-13B	0.00000	0.00010	0.00013	0.00013	0.00009
MPA-SVM-NDWI	0.00010	0.00000	0.76496	0.31380	0.00009
MPA-SVM-NDVI	0.00013	0.76496	0.00000	0.49869	0.00009
MPA-SVM-SAVI	0.00013	0.31380	0.49869	0.00000	0.00009
MPA-SVM-MTCI	0.00009	0.00009	0.00009	0.00009	0.00000

0.946), MPA-SVM-SAVI (AUC = 0.937), MPA-SVM-NDWI (AUC = 0.926), and MPA-SVM-MTCI (AUC = 0.910). The AUC values of the employed models used for urban green space detection are demonstrated in Figure 8. In

addition, the model result comparison in terms of CAR, precision, recall, NPV, and AUC is provided in Figures 9 and 10. The box plots of CAR, F1 score, and AUC are illustrated in Figures 11–13.

TABLE 6: Wilcoxon signed rank test results with F1 score index.

	MPA-SVM-13B	MPA-SVM-NDWI	MPA-SVM-NDVI	MPA-SVM-SAVI	MPA-SVM-MTCI
MPA-SVM-13B	0.00000	0.00009	0.00010	0.00010	0.00009
MPA-SVM-NDWI	0.00009	0.00000	0.70891	0.10843	0.00009
MPA-SVM-NDVI	0.00010	0.70891	0.00000	0.31346	0.00009
MPA-SVM-SAVI	0.00010	0.10843	0.31346	0.00000	0.00009
MPA-SVM-MTCI	0.00009	0.00009	0.00009	0.00009	0.00000

TABLE 7: Wilcoxon signed rank test results with AUC index.

	MPA-SVM-13B	MPA-SVM-NDWI	MPA-SVM-NDVI	MPA-SVM-SAVI	MPA-SVM-MTCI
MPA-SVM-13B	0.00000	0.00009	0.00009	0.00009	0.00009
MPA-SVM-NDWI	0.00009	0.00000	0.00132	0.02762	0.00642
MPA-SVM-NDVI	0.00009	0.00132	0.00000	0.06195	0.00010
MPA-SVM-SAVI	0.00009	0.02762	0.06195	0.00000	0.00010
MPA-SVM-MTCI	0.00009	0.00642	0.00010	0.00010	0.00000

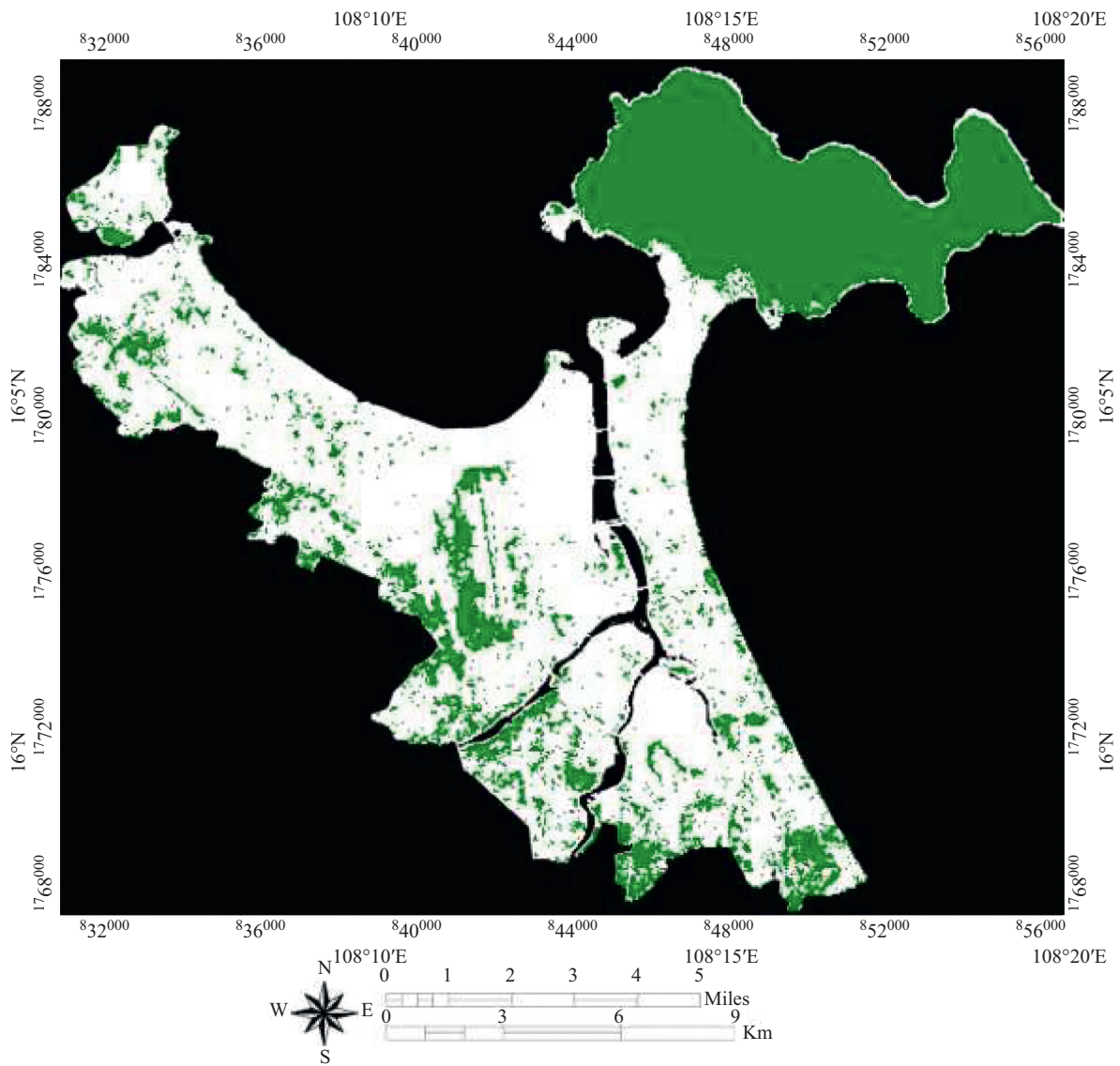


FIGURE 14: Urban green space detection map of the study area.

In addition, to confirm the superiority of the proposed MPA-SVM model that employs all of the Sentinel 2's spectral bands, the Wilcoxon signed-rank test [100] with the significant level (p value) = 0.05 is utilized in this study to express the statistical significance of the model performance indices. The test outcomes of pair-wise model comparison with respect to CAR, F1 score, and AUC are shown in Tables 5–7, respectively. Observably, with p values <0.05, the null hypotheses of insignificant model performances can be rejected. Therefore, MPA-SVM-13B is confirmed to be the best model which provides the classification performance on the collected dataset. Accordingly, the MPA-SVM-13B model is employed to construct an urban green space map for the whole study area. The mapping outcome is demonstrated in Figure 14. Based on the constructed map, it can be found that the green space occupies roughly 34.40% of the study area. Nevertheless, the green space is not evenly distributed in Da Nang. The majority of the green space is located in Son Tra peninsula within the Son Tra district.

5. Concluding Remarks

Urban green space plays a crucial role in improving the living quality of urban environment and has a positive effect on citizens' physical/mental health. Nevertheless, few researches have been dedicated to detecting, locating, and quantifying green space in the study of Da Nang urban center. This study is an attempt to fill this knowledge gap by developing a remote sensing and data-driven approach for urban green space detection applied in the study area. Remotely sensed data obtained from the Sentinel 2 satellite are used to train and validate a hybrid metaheuristic-machine learning approach of MPA-SVM. This hybrid method is employed to construct a decision boundary that separates the input space into two distinctive regions of green space and nongreen space.

The experimental results supported by the Wilcoxon signed-rank test show that the MPA-SVM model employing all of the spectral bands is superior to those of the models relying on individual vegetation indices. Good green space detection results with CAR = 93.100%, precision = 0.916, recall = 0.947, NPV = 0.947, F1 score = 0.931, and AUC = 0.979 demonstrate that the proposed method is highly suited for the task at hand. Moreover, the MPA metaheuristic is confirmed to be a capable method for optimizing machine learning models. Accordingly, the green space mapping of the entire study area can be constructed by the proposed hybrid approach. The information provided by the newly developed model can be helpful for local authority to evaluate the status of green spaces in Da Nang city.

Although MPA-SVM has attained a good predictive performance in urban green space mapping in the study area, the proposed approach also has several limitations. The first limitation is that the MPA-SVM model has not been integrated with feature selection algorithms used for dimensionality reduction. In addition, although the RBFK is widely used for SVM-based pattern recognition, the effectiveness of other sophisticated kernel functions (e.g., hybrid

kernel functions [101, 102]) in urban green space detection should be investigated. Accordingly, future extensions of the current study may include the following:

- (i) Investigating other state-of-the-art metaheuristic algorithms used for optimizing data-driven urban green space detection
- (ii) Studying the effects of the maximum number of searching iterations and the number of population members on the performance of the SVM-based urban green space detection models
- (iii) Employing other advanced texture descriptors to further meliorate the detection accuracy
- (iv) Performing detection tasks at different time periods to inspect changes and trends in urban green space
- (v) Performing urban green space detection using high-resolution satellite images
- (vi) Incorporating advanced feature selection algorithms and kernel functions into the current model structure

Data Availability

The dataset used to support the findings of this study has been deposited in the repository of GitHub (<https://github.com/NDHoangDTU/MPA-SVM-UGSD>).

Conflicts of Interest

The authors confirm that there are no conflicts of interest.

Acknowledgments

This research was funded by the Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant no. 105.99-2019.339.

References

- [1] G. Xian, M. Crane, and D. Steinwand, "Dynamic modeling of Tampa Bay urban development using parallel computing," *Computers & Geosciences*, vol. 31, pp. 920–928, 2005.
- [2] T. Chen, W. Lang, and X. Li, "Exploring the impact of urban green space on residents' health in guangzhou," *China Journal of Urban Planning and Development*, vol. 146, Article ID 05019022, 2020.
- [3] D. T. Do, J. Huang, Y. Cheng, and T. C. T. Truong, "Da Nang green space system planning," *An Ecology Landscape Approach Sustainability*, vol. 10, p. 3506, 2018.
- [4] I. Jarvis, S. Gergel, M. Koehoorn, and M. van den Bosch, "Greenspace access does not correspond to nature exposure: measures of urban natural space with implications for health research," *Landscape and Urban Planning*, vol. 194, Article ID 103686, 2020.
- [5] Ritchie H., Roser M. (2019) Urbanization Our World in Data <https://ourworldindata.org/>.
- [6] M. A. Benedict and E. T. McMahon, *Green Infrastructure: Linking Landscapes and Communities Urban Land*, Island Press, Washington, DC, USA, 2006.
- [7] M. Atasoy, "Monitoring the urban green spaces and landscape fragmentation using remote sensing: a case study in

- Osmaniye, Turkey,” *Environmental Monitoring and Assessment*, vol. 190, no. 12, p. 713, 2018.
- [8] C. Bertram and K. Rehdanz, “The role of urban green space for human well-being,” *Ecological Economics*, vol. 120, pp. 139–152, 2015.
- [9] A. Callaghan, G. McCombe, A. Harrold et al., “The impact of green spaces on mental health in urban settings: a scoping review,” *Journal of Mental Health*, pp. 1–15, 2020, inpress.
- [10] J. Davtalab, S. P. Deyhimi, V. Dessi, M. R. Hafezi, and M. Adib, “The impact of green space structure on physiological equivalent temperature index in open space,” *Urban Climate*, vol. 31, Article ID 100574, 2020.
- [11] K. De Ridder, “An integrated methodology to assess the benefits of urban green space,” *Science of The Total Environment*, vol. 334–335, pp. 489–497, 2004.
- [12] K. K. Lwin and Y. Murayama, “Modelling of urban green space walkability: eco-friendly walk score calculator,” *Computers, Environment and Urban Systems*, vol. 35, pp. 408–420, 2011.
- [13] R. Rafiee, A. Salman Mahiny, and N. Khorasani, “Assessment of changes in urban green spaces of Mashad city using satellite data,” *International Journal of Applied Earth Observation and Geoinformation*, vol. 11, pp. 431–438, 2009.
- [14] D.-h Shin and K.-s Lee, “Use of remote sensing and geographical information systems to estimate green space surface-temperature change as a result of urban expansion,” *Landscape and Ecological Engineering*, vol. 1, pp. 169–176, 2005.
- [15] <https://www.euro.who.int/WHO> (2021) Urban Green Spaces: A Brief for Action World Health Organization.
- [16] R. M. R. Turaga, U. Jha-Thakur, S. Chakrabarti, and D. Hossain, “Exploring the role of urban green spaces in ‘smartening’ cities in India,” *Impact Assessment and Project Appraisal*, vol. 38, pp. 479–490, 2020.
- [17] E. Barbierato, I. Bernetti, I. Capecchi, and C. Saragosa, “Integrating remote sensing and street view images to quantify urban forest ecosystem services,” *Remote Sensing*, vol. 12, no. 2, p. 329, 2020.
- [18] K. Chapi, V. P. Singh, A. Shirzadi et al., “A novel hybrid artificial intelligence approach for flood susceptibility assessment,” *Environmental Modelling & Software*, vol. 95, pp. 229–245, 2017.
- [19] 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.
- [20] P.-T. T. Ngo, T. D. Pham, N. D. Hoang et al., “A new hybrid equilibrium optimized SysFor based geospatial data mining for tropical storm-induced flash flood susceptible mapping,” *Journal of Environmental Management*, vol. 280, Article ID 111858, 2021.
- [21] V.-H. Nhu, “A new hybrid firefly-PSO optimized random subspace tree intelligence for torrential rainfall-induced flash flood susceptible mapping,” *Remote Sensing*, vol. 12, p. 2688, 2020.
- [22] S. A. Yeprintsev, M. A. Klevtsova, L. A. Lepeshkina, S. V. Shekoyan, and A. A. Voronin, “Assessment of the dynamics of urbanized areas by remote sensing,” *IOP Conference Series: Earth and Environmental Science*, vol. 115, Article ID 012034, 2018.
- [23] X. Cao, A. Onishi, J. Chen, and H. Imura, “Quantifying the cool island intensity of urban parks using ASTER and IKONOS data,” *Landscape and Urban Planning*, vol. 96, no. 4, pp. 224–231, 2010.
- [24] W. Kuang and Y. Dou, “Investigating the patterns and dynamics of urban green space in China’s 70 major cities using satellite,” *Remote Sensing Remote Sensing*, vol. 12, p. 1929, 2020.
- [25] J. Li, C. Song, L. Cao, F. Zhu, X. Meng, and J. Wu, “Impacts of landscape structure on surface urban heat islands: a case study of Shanghai,” *China Remote Sensing of Environment*, vol. 115, pp. 3249–3263, 2011.
- [26] S. Oliveira, H. Andrade, and T. Vaz, “The cooling effect of green spaces as a contribution to the mitigation of urban heat,” *A Case Study in Lisbon Building and Environment*, vol. 46, pp. 2186–2194, 2011.
- [27] V. Sathyakumar, R. Ramsankaran, and R. Bardhan, “Geospatial approach for assessing spatiotemporal dynamics of urban green space distribution among neighbourhoods: a demonstration in Mumbai,” *Urban Forestry & Urban Greening*, vol. 48, Article ID 126585, 2020.
- [28] N. H. Wong, S. Kardinal Jusuf, A. Aung La Win, H. Kyaw Thu, T. Syatia Negara, and W. Xuchao, “Environmental study of the impact of greenery in an institutional campus in the tropics,” *Building and Environment*, vol. 42, pp. 2949–2970, 2007.
- [29] A. El Garouani, D. J. Mulla, S. El Garouani, and J. Knight, “Analysis of urban growth and sprawl from remote sensing data: case of Fez,” *Morocco International Journal of Sustainable Built Environment*, vol. 6, pp. 160–169, 2017.
- [30] W. Li, R. Dong, H. Fu, J. Wang, L. Yu, and P. Gong, “Integrating Google Earth imagery with Landsat data to improve 30-m resolution land cover mapping,” *Remote Sensing of Environment*, vol. 237, Article ID 111563, 2020.
- [31] S. Dinda, N. Das Chatterjee, and S. Ghosh, “An integrated simulation approach to the assessment of urban growth pattern and loss in urban green space in Kolkata, India,” *A GIS-Based Analysis Ecological Indicators*, vol. 121, Article ID 107178, 2021.
- [32] K. Bhosle and V. Musande, “Evaluation of deep learning CNN model for land use land cover classification and crop identification using hyperspectral remote sensing images,” *Journal of the Indian Society of Remote Sensing*, vol. 47, no. 11, pp. 1949–1958, 2019.
- [33] M. Campos-Taberner, F. J. García-Haro, B. Martínez et al., “Understanding deep learning in land use classification based on Sentinel-2 time series,” *Scientific Reports*, vol. 10, no. 1, p. 17188, 2020.
- [34] M. Carranza-García, J. García-Gutiérrez, and J. Riquelme, “A framework for evaluating land use and land cover classification using convolutional neural networks,” *Remote Sensing*, vol. 11, no. 3, p. 274, 2019.
- [35] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning (Adaptive Computation and Machine Learning Series)*, The MIT Press, Cambridge, MA, USA, 2016.
- [36] 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.
- [37] Q. He, Q. Zhang, H. Wang, and C. Zhang, “Local similarity-based fuzzy multiple kernel one-class support vector machine,” *Complexity*, vol. 2020, Article ID 8853277, 12 pages, 2020.
- [38] X. Li, Y. Li, Y. Zhang, F. Liu, and Y. Fang, “Fault diagnosis of belt conveyor based on support vector machine and grey

- wolf,” *Optimization Mathematical Problems in Engineering*, vol. 2020, Article ID 1367078, 10 pages, 2020.
- [39] Z. Zhao, T. Liu, and X. Zhao, “Variable selection from image texture feature for automatic classification of concrete surface voids,” *Computational Intelligence and Neuroscience*, vol. 2021, Article ID 5538573, 10 pages, 2021.
- [40] V. N. Vapnik, *Statistical Learning Theory*, John Wiley & Sons, Inc, Hoboken, NJ, USA, 1998.
- [41] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*, Springer, Berlin, Germany, 2011.
- [42] D. Prayogo, M.-Y. Cheng, Y.-W. Wu, and D.-H. Tran, “Combining machine learning models via adaptive ensemble weighting for prediction of shear capacity of reinforced-concrete deep beams,” *Engineering with Computers*, vol. 36, pp. 1135–1153, 2019.
- [43] N. Kardani, A. Zhou, M. Nazem, and S.-L. Shen, “Estimation of bearing capacity of piles in cohesionless soil using optimized machine learning approaches,” *Geotechnical and Geological Engineering*, vol. 38, pp. 2271–2291, 2020.
- [44] H. Nguyen, N.-M. Nguyen, M.-T. Cao, N.-D. Hoang, and X.-L. Tran, “Prediction of long-term deflections of reinforced-concrete members using a novel swarm optimized extreme gradient boosting machine,” *Engineering with Computers*, 2021.
- [45] J.-S. Chou and N.-M. Nguyen, “Metaheuristics-optimized ensemble system for predicting mechanical strength of reinforced concrete materials,” *Structural Control and Health Monitoring*, vol. 28, no. 5, p. e2706, 2021.
- [46] Z. Yu, X. Shi, J. Zhou, X. Chen, and X. Qiu, “Effective assessment of blast-induced ground vibration using an optimized random forest model based on a Harris hawks optimization,” *Algorithm Applied Sciences*, vol. 10, p. 1403, 2020.
- [47] W. Yong, J. Zhou, D. Jahed Armaghani et al., “A new hybrid simulated annealing-based genetic programming technique to predict the ultimate bearing capacity of piles,” *Engineering with Computers*, 2020.
- [48] A. Sina and D. Kaur, “Short term load forecasting model based on kernel-support vector regression with social spider optimization algorithm,” *Journal of Electrical Engineering & Technology*, vol. 15, pp. 393–402, 2020.
- [49] D. Tuan Vu, X.-L. Tran, M.-T. Cao, T. Cuong Tran, and N.-D. Hoang, “Machine learning based soil erosion susceptibility prediction using social spider algorithm optimized multivariate adaptive regression,” *Spline Measurement*, vol. 164, Article ID 108066, 2020.
- [50] N.-T. Ngo, H. A. Le, and T.-P.-T. Pham, “Integration of support vector regression and grey wolf optimization for estimating the ultimate bearing capacity in concrete-filled steel tube columns,” *Neural Computing and Applications*, 2021.
- [51] W. Chen, X. Chen, J. Peng, M. Panahi, and S. Lee, “Landslide susceptibility modeling based on ANFIS with teaching-learning-based optimization and Satin bowerbird optimizer,” *Geoscience Frontiers*, vol. 12, no. 1, pp. 93–107, 2021.
- [52] E. Li, J. Zhou, X. Shi et al., “Developing a hybrid model of salp swarm algorithm-based support vector machine to predict the strength of fiber-reinforced cemented paste backfill,” *Engineering with Computers*, 2020.
- [53] Z. M. Yaseen, H. Faris, and N. Al-Ansari, “Hybridized extreme learning machine model with salp swarm algorithm,” *A Novel Predictive Model for Hydrological Application Complexity*, vol. 2020, Article ID 8206245, 14 pages, 2020.
- [54] Z.-Z. Zhu, Y.-W. Feng, C. Lu, and C.-W. Fei, “Reliability optimization of structural deformation with improved support vector regression,” *Model Advances in Materials Science and Engineering*, vol. 2020, Article ID 3982450, 8 pages, 2020.
- [55] S. Fan, S. Cao, and Y. Zhang, “Temperature prediction of photovoltaic panels based on support vector machine with pigeon-inspired optimization,” *Complexity*, vol. 2020, Article ID 9278162, 12 pages, 2020.
- [56] A. Gholami, S. M. Seyedali, and H. R. Ansari, “Estimation of shear wave velocity from post-stack seismic data through committee machine with cuckoo search optimized intelligence models,” *Journal of Petroleum Science and Engineering*, vol. 189, Article ID 106939, 2020.
- [57] Q. Fang, H. Nguyen, X.-N. Bui, and T. Nguyen-Thoi, “Prediction of blast-induced ground vibration in open-pit mines using a new technique based on imperialist competitive algorithm and M5Rules,” *Natural Resources Research*, vol. 29, no. 9, 2019.
- [58] J. Zhou, Y. Qiu, S. Zhu et al., “Optimization of support vector machine through the use of metaheuristic algorithms in forecasting TBM advance rate,” *Engineering Applications of Artificial Intelligence*, vol. 97, Article ID 104015, 2021.
- [59] Z. Yu, X. Shi, J. Zhou et al., “Feasibility of the indirect determination of blast-induced rock movement based on three new hybrid intelligent models,” *Engineering with Computers*, vol. 37, pp. 991–1006, 2021.
- [60] A. Faramarzi, M. Heidarinejad, S. Mirjalili, and A. H. Gandomi, “Marine predators algorithm: a nature-inspired metaheuristic,” *Expert Systems with Applications*, vol. 152, Article ID 113377, 2020.
- [61] J. W. Rouse, R. H. Haas, J. A. Schell, and D. W. Deering, “Monitoring Vegetation Systems in the Great Plains with ERTS,” in *Conference Proceedings of the Third ERTS Symposium, NASA SP-351*, pp. 309–317, Washington DC, USA, December 1973.
- [62] B.-c Gao, “NDWI—a normalized difference water index for remote sensing of vegetation liquid water from space,” *Remote Sensing of Environment*, vol. 58, pp. 257–266, 1996.
- [63] A. R. Huete, “A soil-adjusted vegetation index (SAVI),” *Remote Sensing of Environment*, vol. 25, pp. 295–309, 1988.
- [64] J. Dash and P. J. Curran, “The MERIS terrestrial chlorophyll index,” *International Journal of Remote Sensing*, vol. 25, no. 23, pp. 5403–5413, 2004.
- [65] MPI (2020) Da Nang City Ministry of Planning and Investment <http://www.mpigovvn/Pages/tinhthanhhचितiet.aspx?idTinhThanh=41>.
- [66] USGS (2020) Earth Explorer US Department of the Interior (<https://earthexplorer.usgs.gov/>).
- [67] ESA (2020) Sentinel Application Platform (SNAP) The European Space Agency <http://stepesaint/main/toolboxes/snap/>.
- [68] ENVI (2021) ENVI Tutorials Harris Geospatial Solutions, <https://www13harrisgeospatial.com/docs/tutorialshtml>.
- [69] J. R. Jensen, *Introductory Digital Image Processing A Remote Sensing Perspective*, Prentice Hall Press, Upper Saddle River, NJ, USA, 2015.
- [70] K. Abutaleb, M. Freddy Mudede, N. Nkongolo, and S. W. Newete, “Estimating urban greenness index using remote sensing data: a case study of an affluent vs poor suburbs in the city of Johannesburg,” *The Egyptian Journal of Remote Sensing and Space Science*, 2020, in press.
- [71] W. Chen, H. Huang, J. Dong, Y. Zhang, Y. Tian, and Z. Yang, “Social functional mapping of urban green space using

- remote sensing and social sensing data,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 146, pp. 436–452, 2018.
- [72] L. Rumora, I. Majić, M. Miler, and D. Medak, “Spatial video remote sensing for urban vegetation mapping using vegetation indices,” *Urban Ecosystems*, vol. 24, pp. 21–33, 2021.
- [73] T.-X. Zhang, J.-Y. Su, C.-J. Liu, and W.-H. Chen, “Potential bands of sentinel-2A satellite for classification problems in precision agriculture,” *International Journal of Automation and Computing*, vol. 16, pp. 16–26, 2019.
- [74] K. W. Pałaś and J. Zawadzki, “Sentinel-2 imagery processing for tree logging observations on the Białowieża forest world heritage site,” *Forests*, vol. 11, p. 857, 2020.
- [75] I. Kamenova and P. Dimitrov, “Evaluation of Sentinel-2 vegetation indices for prediction of LAI, fAPAR and fCover of winter wheat in Bulgaria,” *European Journal of Remote Sensing*, vol. 54, pp. 1–20, 2020.
- [76] F. Bartumeus, J. Catalan, U. L. Fulco, M. L. Lyra, and G. M. Viswanathan, “Optimizing the encounter rate in biological interactions: Lévy versus brownian strategies,” *Physical Review Letters*, vol. 88, no. 9, Article ID 097901, 2002.
- [77] N. E. Humphries, N. Queiroz, J. R. M. Dyer et al., “Environmental context explains Lévy and Brownian movement patterns of marine predators,” *Nature*, vol. 465, pp. 1066–1069, 2010.
- [78] G. M. Viswanathan, E. P. Raposo, and M. G. E. da Luz, “Lévy flights and superdiffusion in the context of biological encounters and random searches,” *Physics of Life Reviews*, vol. 5, pp. 133–150, 2008.
- [79] J. D. Filmlalter, L. Dagorn, P. D. Cowley, and M. Taquet, “First descriptions of the behavior of silky sharks, *Carcharhinus falciformis*, around drifting fish aggregating devices in the Indian Ocean,” *Bulletin of Marine Science*, vol. 87, pp. 325–337, 2011.
- [80] K. O. Achieng, “Modelling of soil moisture retention curve using machine learning techniques: artificial and deep neural networks vs support vector regression models,” *Computers & Geosciences*, vol. 133, Article ID 104320, 2019.
- [81] W. L. Al-Yaseen, Z. A. Othman, and M. Z. A. Nazri, “Multi-level hybrid support vector machine and extreme learning machine based on modified K-means for intrusion detection system,” *Expert Systems with Applications*, vol. 67, pp. 296–303, 2017.
- [82] W. Chen, Z. Shang, and Y. Chen, “A novel hybrid network traffic prediction approach based on support vector machines,” *Journal of Computer Networks and Communications*, vol. 2019, Article ID 2182803, 10 pages, 2019.
- [83] 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.
- [84] K.-W. Liao, N.-D. Hoang, and S.-C. Chang, “Estimating landslide occurrence via small watershed method with relevance vector machine,” *Earth Science Informatics*, vol. 13, pp. 249–260, 2019.
- [85] C. Liu, H. Li, and Q. Zhang, “Research on sewage monitoring and water quality prediction based on wireless sensors and support vector machines,” *Wireless Communications and Mobile Computing*, vol. 2020, Article ID 8852965, 10 pages, 2020.
- [86] A. Malik, Y. Tikhamarine, D. Souag-Gamane, O. Kisi, and Q. B. Pham, “Support vector regression optimized by metaheuristic algorithms for daily streamflow prediction,” *Stochastic Environmental Research and Risk Assessment*, vol. 34, 2020.
- [87] N. Naicker, T. Adeliyi, and J. Wing, “Linear support vector machines for prediction of student performance in school-based education,” *Mathematical Problems in Engineering*, vol. 2020, Article ID 4761468, 7 pages, 2020.
- [88] 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*, vol. 37, pp. 1485–1498, 2019.
- [89] P. Saha, P. Debnath, and P. Thomas, “Prediction of fresh and hardened properties of self-compacting concrete using support vector regression approach,” *Neural Computing and Applications*, vol. 32, pp. 7995–8010, 2019.
- [90] H. Tanyildizi, “Prediction of the strength properties of carbon fiber-reinforced lightweight concrete exposed to the high temperature using artificial neural network and support vector machine,” *Advances in Civil Engineering*, vol. 2018, Article ID 5140610, 10 pages, 2018.
- [91] L. H. Hamel, *Knowledge Discovery with Support Vector Machines*, John Wiley & Sons, Inc., Hoboken, NJ, USA, 2009.
- [92] M.-Y. Cheng and N.-D. Hoang, “Typhoon-induced slope collapse assessment using a novel bee colony optimized support vector classifier,” *Natural Hazards*, vol. 78, no. 3, pp. 1961–1978, 2015.
- [93] S. Theodoridis and K. Koutroubas, *Pattern Recognition*, Academic Press, Cambridge, MA, USA, 2009.
- [94] N.-D. Hoang, “Image processing based automatic recognition of asphalt pavement patch using a metaheuristic optimized machine learning approach,” *Advanced Engineering Informatics*, vol. 40, pp. 110–120, 2019.
- [95] N.-D. Hoang and D. T. Bui, “A novel relevance vector machine classifier with cuckoo search optimization for spatial prediction of landslides,” *Journal of Computing in Civil Engineering*, vol. 30, Article ID 04016001, 2016.
- [96] T. V. Dinh, H. Nguyen, X.-L. Tran, and N.-D. Hoang, “Predicting rainfall-induced soil erosion based on a hybridization of adaptive differential evolution and support vector machine classification,” *Mathematical Problems in Engineering*, vol. 2021, Article ID 6647829, 14 pages, 2021.
- [97] Accord (2019) Accord.NET Framework <http://accord-framework.net/>.
- [98] V. López, A. Fernández, S. García, V. Palade, and F. Herrera, “An insight into classification with imbalanced data: empirical results and current trends on using data intrinsic characteristics,” *Information Sciences*, vol. 250, pp. 113–141, 2013.
- [99] A. R. van Erkel and P. M. T. Pattynama, “Receiver operating characteristic (ROC) analysis: basic principles and applications in radiology,” *European Journal of Radiology*, vol. 27, pp. 88–94, 1998.
- [100] M. Hollander and D. A. Wolfe, *Nonparametric Statistical Methods*, John Wiley & Sons, Hoboken, NJ, USA, 1999.
- [101] D. Seger, M. Mbuthia, and A. Nyete, “Particle swarm optimized hybrid kernel-based multiclass support vector machine for microarray cancer data analysis,” *BioMed Research International*, vol. 2019, Article ID 4085725, 11 pages, 2019.
- [102] X. Wu, W. Tang, and X. Wu, “Support vector machine based on hybrid kernel function,” *Information Engineering and Applications*, Springer, London, UK, pp. 127–133, 2012.

Research Article

Virtual Screening of Drug Proteins Based on Imbalance Data Mining

Peng Li , Lili Yin, Bo Zhao, and Yuezhongyi Sun

School of Computer Science and Technology, Harbin University of Science and Technology, Harbin 150080, Heilongjiang, China

Correspondence should be addressed to Peng Li; printing3d@126.com

Received 30 January 2021; Accepted 11 May 2021; Published 24 May 2021

Academic Editor: Erik Cuevas

Copyright © 2021 Peng Li et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

To address the imbalanced data problem in molecular docking-based virtual screening methods, this paper proposes a virtual screening method for drug proteins based on imbalanced data mining, which introduces machine learning technology into the virtual screening technology for drug proteins to deal with the imbalanced data problem in the virtual screening process and improve the accuracy of the virtual screening. First, to address the data imbalance problem caused by the large difference between the number of active compounds and the number of inactive compounds in the docking conformation generated by the actual virtual screening process, this paper proposes a way to improve the data imbalance problem using SMOTE combined with genetic algorithm to synthesize new active compounds artificially by upsampling active compounds. Then, in order to improve the accuracy in the virtual screening process of drug proteins, the idea of integrated learning is introduced, and the random forest (RF) extended from Bagging integrated learning technique is combined with the support vector machine (SVM) technique, and the virtual screening of molecular docking conformations using RF-SVM technique is proposed to improve the prediction accuracy of active compounds in docking conformations. To verify the effectiveness of the proposed technique, first, HIV-1 protease and SRC kinase were used as test data for the experiments, and then, CA II was used to validate the model of the test data. The virtual screening of drug proteins using the proposed method in this paper showed an improvement in both enrichment factor (EF) and AUC compared with the use of the traditional virtual screening, for the test dataset. Therefore, it can be shown that the proposed method can effectively improve the accuracy of drug virtual screening.

1. Introduction

In recent years, the continuous discovery of natural and synthetic compounds and other small molecules has provided a large amount of data validation resources for the drug development process, but since the traditional validation process in drug development is still mainly by means of clinical trials, nothing can be done in the face of the large number of clinical trial datasets nowadays [1]. This has led to the phenomenon that although a very large amount of money is invested in the drug development process each year, very few drugs are actually produced [2]. In the drug development process, the type of lead compound and the quality of the compound have a direct impact on the ease and duration of drug development; for example, the better the quality of the lead compound, the lower the elimination

rate of the drug in the drug development process, and vice versa [3]. Since the 1980s, computer technology has been widely used in the drug development process due to the efficient computing power of computers [4]. The use of computer technology in drug development not only saves time and money but also, and most importantly, greatly improves the accuracy of the drug development process. One of the typical representatives of computer-aided drug design can be called virtual screening technology [5].

Virtual screening technology is actually a simulation of the drug development process on a computer, which makes it an efficient drug development aid due to the efficient performance of computers, and it can also improve the accuracy of drug development [6]. In experimental datasets, imbalance between inactive and active compound datasets can make the experimental data biased and seriously affect

the experimental results of virtual screening [7]. Therefore, the virtual screening process of drugs can be viewed as an imbalanced data classification problem. Traditional virtual screening techniques usually use two schemes, similarity search and sampling scoring, for the screening of lead compounds [8]. However, both schemes do not deal well with the problem of data imbalance caused by the disparity in the number of active and inactive compounds among the candidate compounds. In this paper, we introduce the problem of processed imbalanced data into the process of virtual screening of drug proteins, an idea that is still rare in the field of drug development. In the actual drug development process, we should focus more on the data of active compounds with less data volume in the dataset. Therefore, a detailed study of this imbalanced data prediction problem is carried out in this paper, and its study is applied to the drug virtual screening process, and a complete drug virtual screening scheme is established.

In the process of drug development, virtual screening of drug proteins using computer-aided drug design can improve the efficiency of drug development to a great extent [9]. To address the problem of imbalanced data arising from the large difference between the number of active and inactive compounds during the virtual screening approach, this paper proposes a genetic algorithm based on SMOTE to perform preprocessing on the imbalanced dataset as a way to reduce the imbalance ratio of the categorical data, by increasing the number of positive examples of data from a few classes, not only to ensure the integrity of the data but also reduce the imbalance ratio. In response to the problem that the traditional virtual screening technique is not effective in classification, this paper introduces the idea of integrated learning and proposes a more effective classification of the binding conformation of molecules based on RF-SVM classification algorithm, from which effective active compounds can be screened, and this method can not only improve the screening accuracy of lead compounds but also accelerate the drug development process and shorten the drug development cycle [10].

2. Genetic Algorithm Upsampling Technique Based on SMOTE

Data scarcity is evident during virtual screening experiments, where out-of-balance data are often required. This is because the crystal structures of many proteins have not yet been resolved. For example, the crystal structure of the resolved SRC kinase is only about 90, and if SRC is used as a target protein for virtual screening, it is easy to have a serious imbalance in the number of positive and negative example data samples in the dataset. The imbalance data problem in the classification problem is mainly manifested in the imbalance of the positive and negative data, often appearing in the classification process to the side of the majority of the negative data, and some may even cause the phenomenon of "data flooding." To address the imbalance problem, this paper combines the advantages of SMOTE algorithm and genetic algorithm and proposes a genetic algorithm upsampling technique based on SMOTE.

The SMOTE algorithm mainly aims to reduce the imbalance ratio by adding the number of data elements of the positive sample, but there is a certain blindness in generating new data. There are many uncertainties and errors in the process of actual use; for example, the data information around the positive sample is not considered in the process of sample synthesis, so the phenomenon of data confusion is easy to occur.

The genetic algorithm synthesizes new data samples by selecting the determined sample data by crossover variation, where the selection operator can be used to select the samples of the dataset to be manipulated and to control the number of new data to be synthesized in the end, the crossover operator of the genetic algorithm can keep the newly generated samples similar to the original sample data, and the diversity of the new samples can be maintained by the variation operator. By studying the advantages and disadvantages of these two algorithms in synthesizing new datasets, a new algorithm (GA-SMOTE) is proposed, which incorporates the ideas of the SMOTE algorithm into the genetic algorithm.

The specific steps of the new algorithm are as follows.

- (1) The degree of influence of positive and negative examples in the training dataset is coded according to the eigenvalues of the sample dataset
- (2) Two samples are selected from the sample data according to the fitness function
- (3) The selected two samples are crossed or mutated to produce a new sample dataset

The flow of the new algorithm is as follows.

- (1) Set the input original training sample TrainData; set the output path DataSet. Set the number of feature values in the sample dataset to n , the number of large class samples in the dataset to N , and the number of small class samples to T .
- (2) Set the number of samples NUM, that is, the number of minority class samples that need to be synthesized in order to balance the minority class samples and the majority class samples.
- (3) Coding of eigenvalues.
 - ① A sample is randomly selected from a small number of classes T_i
 - ② The k minority class sample sets $\{T_1, T_2, T_3, \dots, T_k\}$ around T_i and the k majority class sample datasets $\{N_1, N_2, N_3, \dots, N_k\}$ are selected by the Euclidean distance approach in the SMOTE algorithm, while the k sample data around T_i are selected, and if more than half of the k sample data are majority class samples, this data is assumed to be at the boundary position and no cross-variance operation is performed on it
 - ③ Calculate the average value Lt of the distance from T_i to $\{T_1, T_2, T_3, \dots, T_k\}$ and the average value Ln of the distance from T_i

to $\{N_1, N_2, N_3, \dots, N_k\}$, and calculate $(Lt/Ln) = L_1$

- ④ Remove a random eigenvalue from the sample, and then, calculate the average value Lt' of the distance from T_i to $\{T_1, T_2, T_3, \dots, T_k\}$ and the average value Ln' of the distance from T_i to $\{N_1, N_2, N_3, \dots, N_k\}$, respectively, after removing this eigenvalue. $(Lt'/Ln') = L_2$ is calculated
- ⑤ If $L_1 > L_2$, then this eigenvalue can be characterized as majority class; otherwise, this eigenvalue can be characterized as minority class, and then use 0 to characterize the majority class eigenvalue and 1 to characterize the minority class eigenvalue
- ⑥ Repeat the previously mentioned steps until all n eigenvalues are encoded and finished

The pseudocode is as follows (see Algorithm 1).

- (4) The minority class is ranked according to the fitness function of the minority class sample, and the top 50% is taken out, and then two samples of data are randomly selected from them, and the crossover variation is performed. If it is the majority class eigenvalue, the variation is performed by SMOTE algorithm, and if it is the minority class sample, the crossover is performed. The schematic diagram is shown in Figure 1, where x represents a random variable between 0 and 1.
- (5) Loop through the fourth step until the new sample data generated reaches NUM.

The flowchart of the improved genetic algorithm is shown in Figure 2.

2.1. Introduction of RF-SVM-Based Classification Algorithm.

Currently, support vector machines (SVM) have been able to solve most of the data classification problems with relatively small amount of data, distinct eigenvalue identification, and relatively balanced data distribution [11]. However, for the classification problem of imbalanced datasets, the effectiveness of SVM decreases significantly. It is mainly because of the uneven distribution of the training dataset, which leads to a serious imbalance in the ratio of positive and negative example sample data, and the majority class of negative example data samples dominates, making the final classification hyperplane tilted towards the negative example data. Experimental studies have shown that SVM is more effective in dealing with classification of imbalanced data compared to other classification models. The flow of SVM in dealing with classification problems is as follows.

The given sample dataset is denoted as $L = \{(x_1, y_1), (x_2, y_2), (x_3, y_3), \dots, (x_n, y_n)\}$, where $x_i \in R^d$, $y_i \in \{-1, 1\}$, $i = 1, 2, 3, \dots, n$. y_i denotes the class of sample x_i , d denotes the number of bits of sample data, and n denotes the number of sample data for training. The representation of the hyperplane is as follows:

$$w \cdot x + b = 0, \quad (1)$$

where w is the normal vector, representing the direction of the hyperplane, and b is the offset, representing the distance between the hyperplane and the origin. Therefore, a hyperplane can be represented by (w, b) . The distance of any individual x in the sample space to the hyperplane can be expressed as r :

$$r = \frac{|wx + b|}{\|w\|}. \quad (2)$$

The SVM algorithm learning problem can be formulated as a constrained optimality problem, as shown in the following equation:

$$\min \frac{\|w\|^2}{2}, \quad (3)$$

and $(\|w\|^2/2)$ denotes the maximization decision edge, which has the following constraints:

$$y_i * (w \cdot x + b) \geq 1, \quad i = 1, 2, 3, \dots, n. \quad (4)$$

This is a convex optimization problem, which can be solved by introducing the standard form of Lagrange multipliers with a modification of the objective function, as shown in the following equation:

$$L_p = \frac{\|w\|^2}{2} - \sum_{i=1}^n \lambda_i (y_i * (w \cdot x + b) - 1), \quad (5)$$

where λ_i denotes the introduced Lagrange multiplier. In order to minimize λ_i , w and b can be made equal to zero, and then, the derivative can be obtained as follows:

$$\frac{\partial L_p}{\partial w} = 0 \implies w = \sum_{i=1}^n \lambda_i y_i x_i, \quad (6)$$

$$\frac{\partial L_p}{\partial b} = 0 \implies \sum_{i=1}^n \lambda_i y_i = 0.$$

The problem can be transformed into the pairwise function formula shown in the following equation:

$$L_D = \sum_{i=1}^n \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j x_i \cdot x_j. \quad (7)$$

The decision boundary can also be expressed as shown in the following equation:

$$\left(\sum_{i=1}^n \lambda_i y_i x_i \cdot x \right) + b = 0. \quad (8)$$

After determining the parameters of the decision boundary, the final classifier formula is obtained as shown in the following equation:

$$f(z) = \text{sign}(w \cdot x + b) = \text{sign} \left(\sum_{i=1}^n \lambda_i y_i x_i \cdot z + b \right), \quad (9)$$

where sign denotes the symbolic function and z denotes the instance data to be detected.

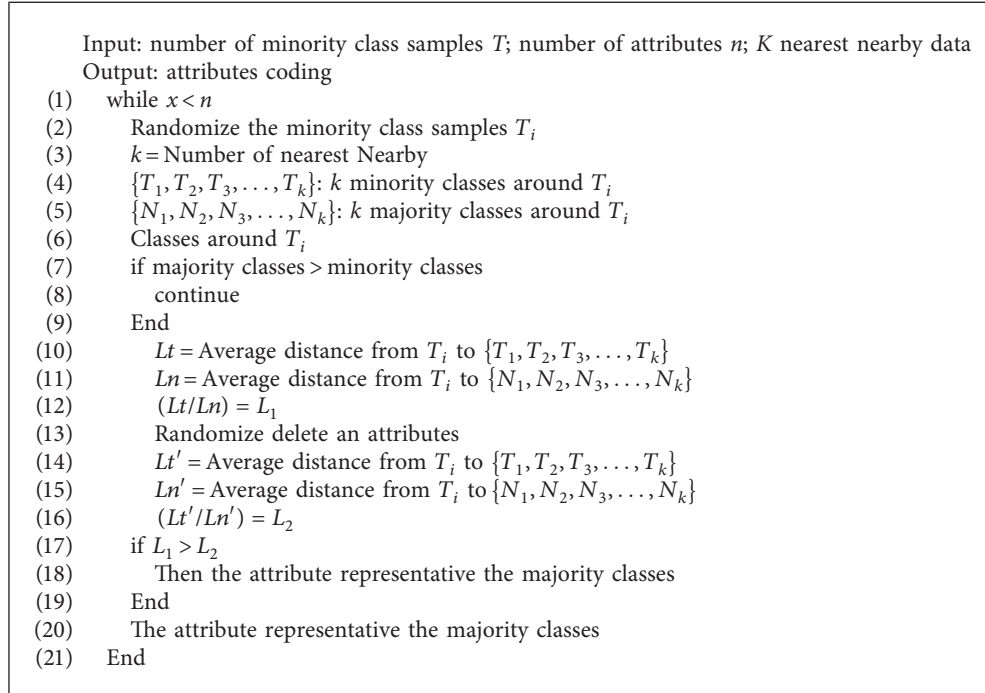
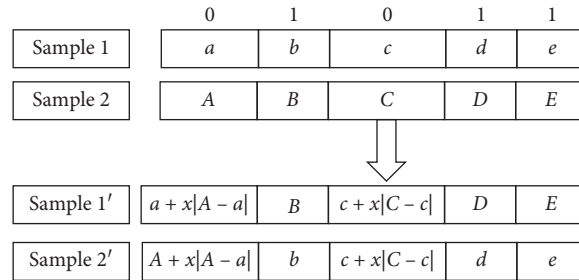
ALGORITHM 1: GA_SMOTE (T, n, k, x).

FIGURE 1: Eigenvalue cross-variation graph.

To address the shortcomings of SVM in dealing with imbalanced data problems, this paper introduces the idea of integrated learning. Since the Bagging classifier does not target a specific instance data in the training set, each sample is selected with equal probability. Bagging has a better fitting performance, and the independence between individual classifiers of Bagging is higher. Therefore, the classifiers can run in parallel with each other, and the speed is faster, almost the same as the speed of individual classifiers. The random forest algorithm is an extension of Bagging, which not only retains all the advantages of Bagging and adds automatic feature selection but is also simpler to implement than the Boosting family of algorithms. SVM shows good performance for processing small sample data, which is in line with the realistic data situation in the virtual screening process, and can also solve the high-dimensional data classification problem, so this paper chooses the combination of RF and SVM to build a classification model for the data of drug proteins.

The algorithmic idea is as follows: random forest is actually a combined classifier, combining individual

classifiers together using the voting method to get the final classifier, because RF is not sensitive to the noise data in the dataset; therefore, the noise data in the data are first filtered by the SVM algorithm, that is, the misclassified data or the data that are not easily distinguished are extracted and selected, and then the data are learned and trained by the RF algorithm.

Input data are as follows: the training sample dataset $L = \{(x_1, y_1), (x_2, y_2), (x_3, y_3), \dots, (x_n, y_n)\}$, $x_i \in R^d$, $y_i \in \{-1, 1\}$, $i = 1, 2, 3, \dots, n$ and the number of input iterations N .

Phase I:

- (1) First, a portion of the data in L is selected for training, and then, the remaining data is iteratively added
- (2) Classify the data through the hyperplane and remove the assumed noisy dataset according to a certain strategy
- (3) Repeat the above steps N times

Phase II: training sample dataset N ; number of base classifiers of RF X

- (4) The training sample data N is randomly divided into X points, and the subdata samples are assumed to be $T_i, i = 1, 2, 3, \dots, X$
- (5) Train the base classifier by base classifier CART decision tree
- (6) Repeat the previously mentioned steps X times
- (7) The final strong classifier is selected based on the voting method

It is demonstrated experimentally that the best training effect of RF is achieved when X is taken as 100, and the sampling ratio of training data can be taken as 0.63. The value of N not only affects the accuracy of SVM algorithm and the speed of SVM but also affects the judgment of noisy data. Therefore, the final classification result is optimal when the value of N is about 40 through the experimental constant tuning test.

3. Experimental Verification and Analysis

3.1. Evaluation Criteria. In this paper, we chose the enrichment factor (EF) as one of the evaluation indexes. EF is an important criterion for evaluating the effectiveness of virtual screening, and it is also the most general criterion for evaluating virtual screening techniques. In the case of relatively small sample data, a larger value of the enrichment factor indicates a greater number of active compounds, which can be detected [10]. The formula for calculating EF is as follows:

$$EF = \left(\frac{\text{Hits}_{\text{sampled}}^{x\%}}{N_{\text{sampled}}^{x\%}} \right) \cdot \left(\frac{N_{\text{total}}}{\text{Hits}_{\text{total}}} \right), \quad (10)$$

where $\text{Hits}_{\text{sampled}}^{x\%}$ denotes the number of active compounds in $x\%$ of the test set database, $N_{\text{sampled}}^{x\%}$ denotes the number of inactive decoys in $x\%$ of the test set database, N_{total} denotes the number of compounds in the entire test set, and $\text{Hits}_{\text{total}}$ denotes the number of active compounds in the entire test set.

In the actual drug screening process, only a small fraction of the compounds are screened by computer means. In this paper, EF is selected as the evaluation criteria for virtual screening, and ROC curve and AUC area are also introduced in the evaluation criteria of this paper. The ROC curve, also known as the “subject operating characteristic curve,” is a visual graphical evaluation criterion that graphically represents the correlation between sensitivity and specificity. The false positive rate is used as the horizontal coordinate and the true positive rate as the vertical coordinate. The AUC area represents the area of the graph enclosed by the ROC curve and the x -coordinate axis, allowing a more visual analysis of the model [12]. The formula for the AUC area is as follows:

$$AUC(f) = \sum_{i=1}^n \frac{1}{2} (y_{i-1} + y_i) \cdot (x_i - x_{i-1}). \quad (11)$$

The value of AUC area ranges from 0 to 1, but it is usually between 0.5 and 1. When the value of AUC reaches 1, it means that the classification effect of the classifier reaches the optimal situation, and the larger the value of AUC area is, the better the prediction effect of this model classifier is.

3.2. Experimental Data Acquisition. The two specific techniques proposed in this paper, “SMOTE-based genetic algorithm” and “RF-SVM-based classification algorithm,” were validated using HIV-1 protease and SRC kinase as experimental data. HIV-1 protease is one of the more critical proteases for the treatment of human immunodeficiency syndrome, and HIV-1 protease has two identical peptide chains, a homodimer composed of two polypeptides [13]. SRC kinase is widely present in tissue cells and can react with important molecules in signal transduction pathways, participating in cellular metabolic processes and regulating cell growth, development, and differentiation. This kinase has been implicated in the development of several cancers [14]. Activation of the SRC pathway has been detected in approximately more than 50% of various cancers, so the treatment of cancer can be achieved by inhibiting the activity of SRC [12]. The crystal structures of HIV-1 protease and SRC kinase can be obtained directly from the PDB database. The HIV-1 protease has a PID of 1PRO with a resolution of 1.80 Å [15] and the structure is shown in Figure 3. The SRC kinase has a PID of 2H8H with a resolution of 2.20 Å and the structure is shown in Figure 4.

In this paper, the experiments are divided into four cases, in which the two proteins are compared before and after processing by the “SMOTE-based genetic algorithm” and the comparison experiments using the two methods of SVM and RF-SVM. The data structures before and after sampling are shown in Tables 1 and 2.

For both datasets, HIV-1 and SRC, the ROCs of the previously mentioned two datasets for the comparison experiments are shown in Figure 5.

In order to prove the validity of the experiments, two sets of representative data were used as experimental data for model reliability comparison experiments, which were conducted before and after sampling preprocessing, SVM algorithm and RF-SVM algorithm, respectively. The differences in ROC curve, AUC area, enrichment factor (EF), and accuracy were obtained by the experimental results, which showed that the RF-SVM algorithm proposed in this paper was improved for the virtual screening experimental results of drug proteins.

The data of HIV-1 protease and SRC kinase were observed: when HIV-1 protease was used as the target protein, the area of AUC increased from 0.795 to 0.839, the enrichment factor increased from 6.58 to 7.16, and the accuracy increased from 0.651 to 0.746, indicating that the results of the virtual screening of drug proteins by RF-SVM were improved after the sampling preprocessing. When the virtual screening simulations were performed by the basic SVM algorithm and RF-SVM algorithm before the sampling preprocessing, the area of the AUC increased from 0.729 to 0.785, the enrichment factor increased from 5.93 to 6.51, and

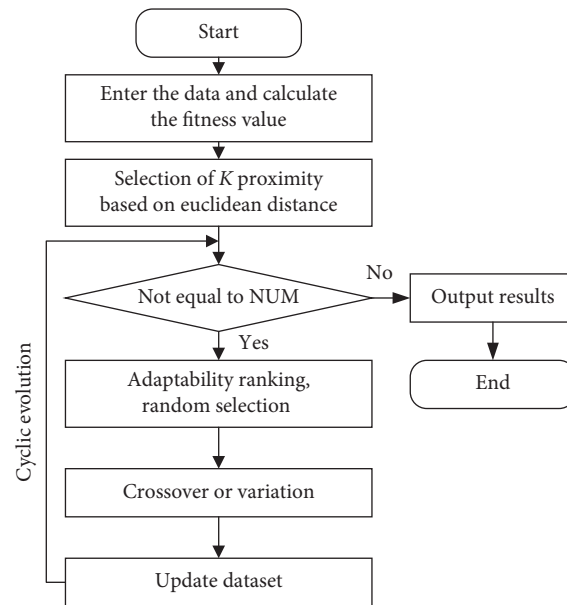


FIGURE 2: Improved genetic algorithm flowchart.

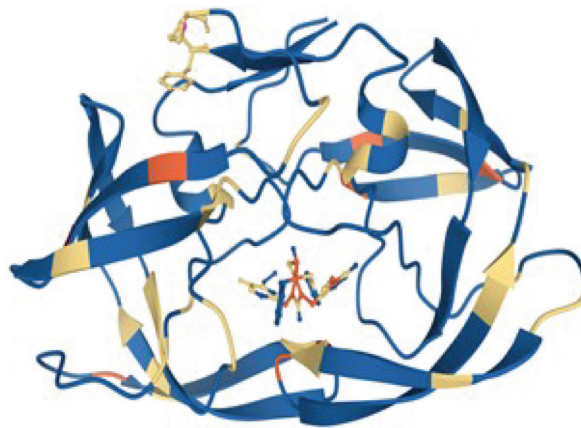


FIGURE 3: PID is 1RPO HIV-1 protease complex binding crystal.

accuracy increased from 0.547 to 0.647, indicating that the results of RF-SVM for virtual screening of drug proteins were improved when the sampling preprocessing was performed. When the virtual screening simulation experiments were performed using the basic SVM algorithm and the RF-SVM algorithm with SRC kinase as the target protein, the area of AUC increased from 0.741 to 0.828, the enrichment factor also increased from 6.25 to 6.96, and the accuracy also increased from 0.674 to 0.754 after sampling pretreatment. This indicates that, after sampling, the area of AUC increased from 0.722 to 0.769, the enrichment factor increased from 5.87 to 6.43, and accuracy increased from 0.544 to 0.679 when the basic SVM algorithm and RF-SVM algorithm were used for virtual screening simulation experiments before sampling pretreatment. The experimental results show that after pretreatment by sampling, the results of RF-SVM for virtual screening of drug proteins have been improved.

3.3. Experimental Verification. In order to verify the validity of the method proposed in this paper, the experimental results were verified using carbonic anhydrase II (CA II), a zinc-containing metalloenzyme that reversibly mediates the hydration of carbon dioxide, which maintains acid-base balance in blood and other tissues and helps tissues in the body to eliminate carbon dioxide. CA II has a PID of 1Z9Y and a resolution value of 1.66 Å. Its structure is shown in Figure 6.

The experimental results are summarized in Table 3.

The comparative results of the experiment are shown in Figure 7.

The stacking of ROC curves and AUC areas in the experimental results can be obtained: the virtual screening results have improved after sampling by sampling, which can verify the effectiveness of the method proposed in this paper. The AUC increased from 0.717 to 0.775, the enrichment factor increased from 5.83 to 6.36, and the

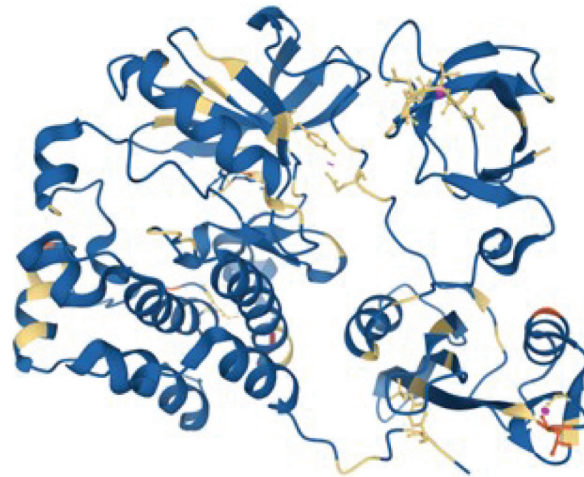


FIGURE 4: PID is 2H8H SRC kinase complex binding crystal.

TABLE 1: HIV-1 data classification results.

	Algorithm	Protein name	EF	AUC
Before sampling	SVM	HIV-1	5.93	0.729
Before sampling	RF-SVM	HIV-1	6.51	0.785
After sampling	SVM	HIV-1	6.58	0.795
After sampling	RF-SVM	HIV-1	7.16	0.839

TABLE 2: SRC data classification results.

	Algorithm	Protein name	EF	AUC
Before sampling	SVM	SRC	5.87	0.722
Before sampling	RF-SVM	SRC	6.43	0.769
After sampling	SVM	SRC	6.25	0.741
After sampling	RF-SVM	SRC	6.96	0.828

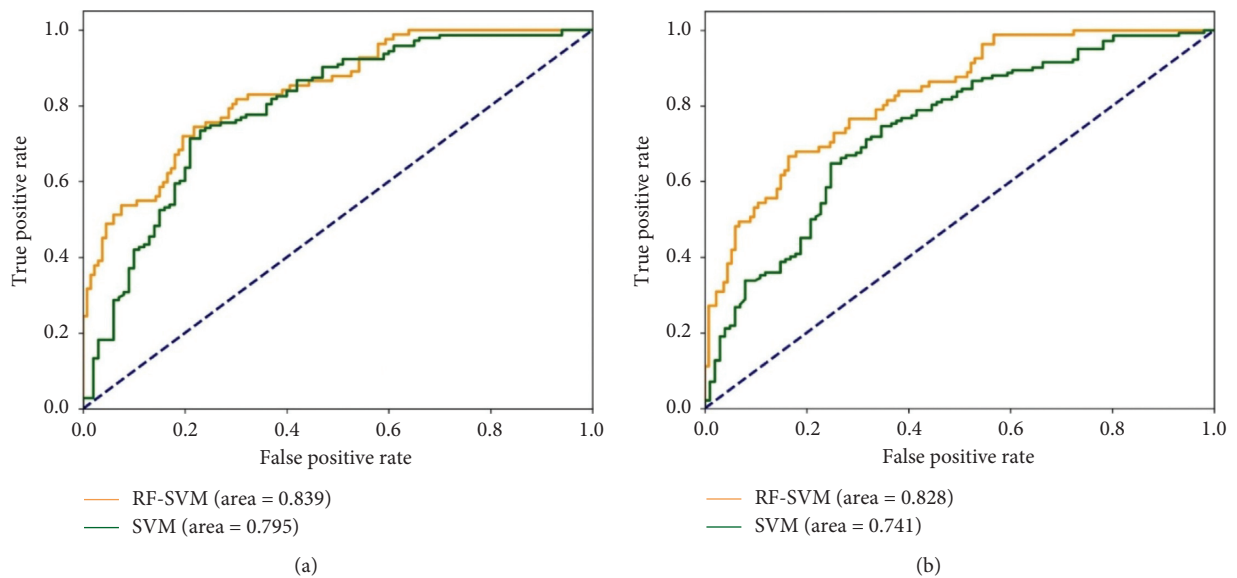


FIGURE 5: Continued.

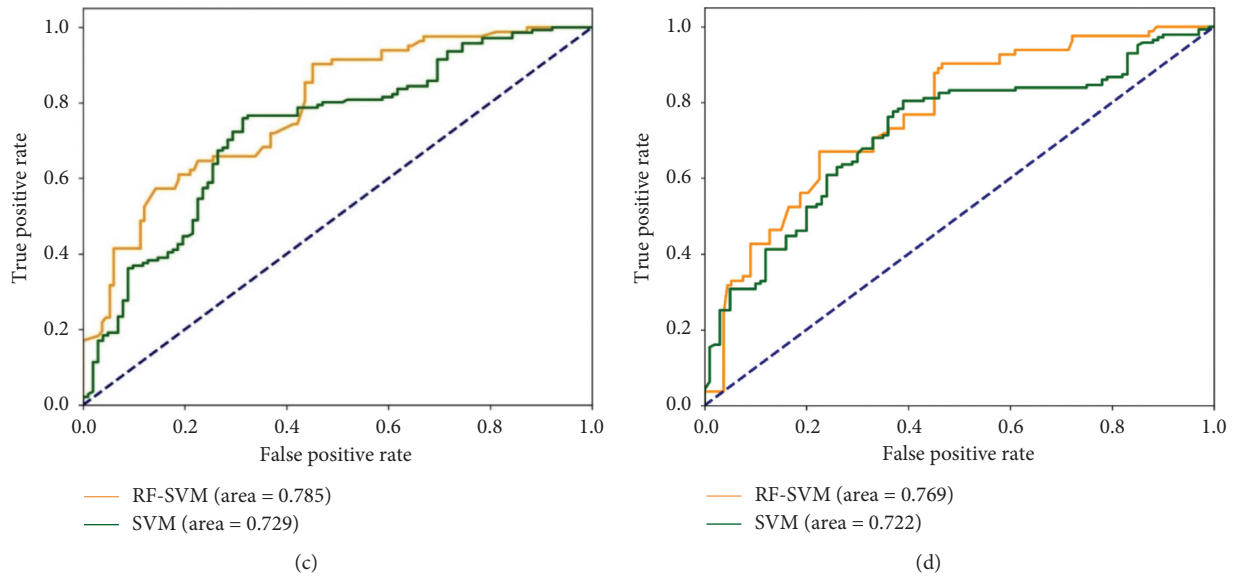


FIGURE 5: Graph of ROC comparison of two experimental datasets.

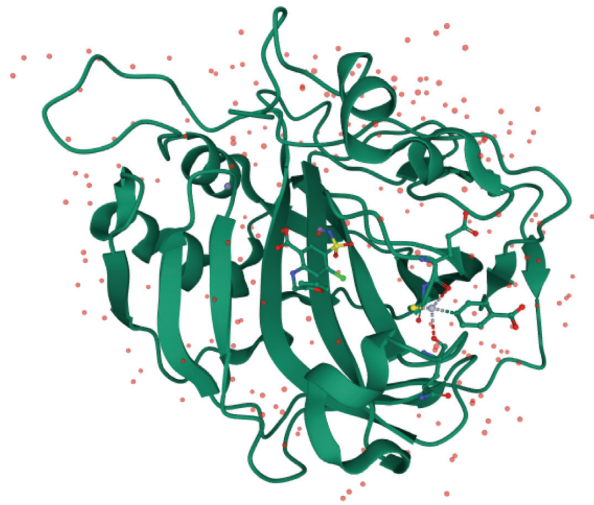


FIGURE 6: CA II with PID of 1Z9Y.

TABLE 3: CA II data classification results.

	Algorithm	Protein name	EF	AUC
Before sampling	SVM	CA II	5.83	0.717
After sampling	RF-SVM	CA II	6.36	0.775

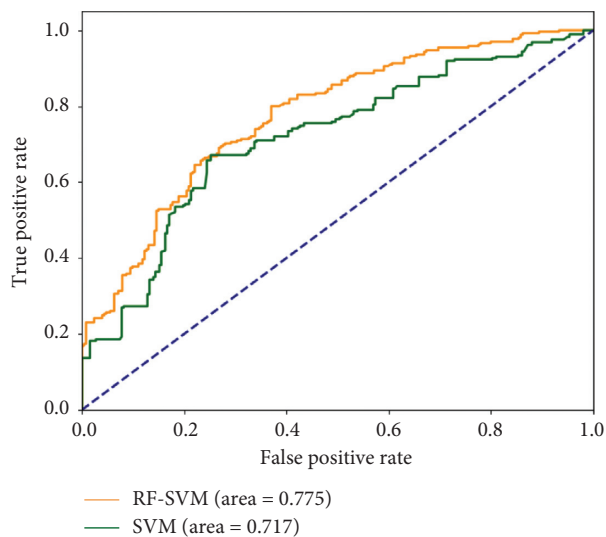


FIGURE 7: Comparative results of experimental data.

accuracy increased from 0.675 to 0.710 when SVM and RF-SVM were used for classification, indicating that the results of RF-SVM for virtual screening of drug proteins were improved.

4. Conclusion

In this paper, we studied the traditional molecular docking-based virtual screening technology and pointed out some problems of the present virtual screening methods, such as the accuracy of the scoring function and the data imbalance between inactive compounds and active compounds in docked conformations. In order to further improve the screening quality of the lead compounds from the virtual screening, this paper incorporates machine learning methods into the traditional virtual screening process. In order to solve the problem of imbalanced data in realistic virtual screening process, this paper proposes the preprocessing of active compounds by combining genetic algorithm with SMOTE. The solutions proposed in this paper provide new ideas for the study of actual virtual screening techniques. The main findings of this paper include the following.

First, for the situation that the number of active compounds is much less than the number of inactive compounds that occurs in the actual virtual screening process, this paper proposes a preprocessing method to deal with the imbalanced data problem by using a combination of SMOTE algorithm and genetic algorithm to upsample the data of minority class samples to balance the imbalanced dataset by increasing the number of minority class samples. This approach was chosen because the actual number of active compounds in the virtual screening process is very small, and this upsampling approach not only preserves the valid information of the data but also solves the data imbalance problem and avoids overfitting to some extent.

Second, in order to improve the quality of the lead compounds in the virtual screening process, this paper

introduces the idea of integrated learning into the virtual screening process of drug proteins and proposes the RF-SVM method. The dataset is optimally selected by support vector machine, and then, the selected dataset is used as the data input of random forest. By this way, the insensitivity of random forest to noise can be effectively avoided, thus improving the generalization ability of the classification algorithm. The simulation experiments of virtual screening were performed by two more important target proteins, HIV-1 protease and SRC kinase, and then, the models were validated by CA II, which showed that the EF of these three datasets improved, on average, by about 0.95, all of which can show that the proposed method in this paper is effective for improving the virtual screening method.

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 paper was supported by the Fundamental Research Foundation for Universities of Heilongjiang Province (no. LGYC2018JQ003) and University Nursing Program for Young Scholars with Creative Talents in Heilongjiang Province (no. UNPYSCT-2018208).

References

- [1] X. Han, Z. Bao, and H. Zhao, "High throughput screening and selection methods for directed enzyme evolution," *Industrial & Engineering Chemistry Research*, vol. 54, no. 16, pp. 4011–4020, 2015.

- [2] U. H. Manjunatha, A. T. Chao, F. J. Leong, and T. T. Diagana, "Cryptosporidiosis drug discovery: opportunities and challenges," *ACS Infectious Diseases*, vol. 2, no. 8, pp. 530–537, 2016.
- [3] H. S. Roy, G. Dubey, V. K. Sharma, P. V. Bharatam, and D. Ghosh, "Molecular docking and molecular dynamics to identify collagenase inhibitors as lead compounds to address osteoarthritis," *Journal of Biomedical Structure and Dynamics*, vol. 1, pp. 1–13, 2020.
- [4] J. Li, S. Fong, S. Mohammed, and J. Fiaidhi, "Improving the classification performance of biological imbalanced datasets by swarm optimization algorithms," *The Journal of Supercomputing*, vol. 72, no. 10, pp. 3708–3728, 2016.
- [5] A. Roy, B. Srinivasan, and J. Skolnick, "PoLi: a virtual screening pipeline based on template pocket and ligand similarity," *Journal of Chemical Information and Modeling*, vol. 55, no. 8, pp. 1757–1770, 2015.
- [6] Y. Wang, H. Guo, Z. Feng et al., "PD-1-targeted discovery of peptide inhibitors by virtual screening, molecular dynamics simulation, and surface plasmon resonance," *Molecules*, vol. 24, no. 20, pp. 3784–3791, 2019.
- [7] T. Pan, J. Zhao, W. Wu, and J. Yang, "Learning imbalanced datasets based on SMOTE and Gaussian distribution," *Information Sciences*, vol. 512, pp. 1214–1233, 2020.
- [8] K. A. Carpenter, D. S. Cohen, J. T. Jarrell, and X. Huang, "Deep learning and virtual drug screening," *Future Medicinal Chemistry*, vol. 10, no. 21, pp. 2557–2567, 2018.
- [9] L. Isaias, K. Palacio-Rodríguez, C. N. Cavasotto, and P. Cossio, "Flexi-pharma: a molecule-ranking strategy for virtual screening using pharmacophores from ligand-free conformational ensembles," *Journal of Computer-Aided Molecular Design*, vol. 34, no. 10, pp. 1063–1077, 2020.
- [10] A. Kumar and P. Kumar, "Identification of good and bad fragments of tricyclic triazinone analogues as potential PKC- θ inhibitors through SMILES-based QSAR and molecular docking," *Structural Chemistry*, vol. 32, pp. 149–165, 2020.
- [11] A. K. Jain, "Data clustering: 50 years beyond K -means," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 651–666, 2010.
- [12] S. Martellucci, L. Clementi, S. Sabetta, V. Mattei, L. Botta, and A. Angelucci, "SRC family kinases as therapeutic targets in advanced solid tumors: what we have learned so far," *Cancers*, vol. 12, no. 6, pp. 1448–1475, 2020.
- [13] M. M. Lawal, Z. K. Sanusi, T. Govender, G. E. M. Maguire, B. Honarparvar, and H. G. Kruger, "From recognition to reaction mechanism: an overview on the interactions between HIV-1 protease and its natural targets," *Current Medicinal Chemistry*, vol. 27, no. 15, pp. 2514–2549, 2020.
- [14] M. Kästle, C. Merten, R. Hartig et al., "Tyrosine 192 within the SH2 domain of the SRC-protein tyrosine kinase p56Lck regulates T-cell activation independently of Lck/CD45 interactions," *Cell Communication and Signaling: CCS*, vol. 18, no. 1, p. 183, 2020.
- [15] S. Chakraborty, M. Phu, T. Prado de Morais et al., "The PDB database is a rich source of alpha-helical anti-microbial peptides to combat disease causing pathogens," *F1000Research*, vol. 3, p. 295, 2015.

Research Article

Multiobjective Brain Storm Optimization Community Detection Method Based on Novelty Search

Xiaoying Pan,^{1,2} Jia Wang ,¹ Miao Wei,¹ and Hongye Li¹

¹*Xi'an University of Posts and Telecommunications, Xi'an 710121, China*

²*Shaanxi Key Laboratory of Network Data Analysis and Intelligent Processing,
Xi'an University of Posts and Telecommunications, Xi'an, Shaanxi 710121, China*

Correspondence should be addressed to Jia Wang; 17691032356@163.com

Received 1 February 2021; Revised 16 March 2021; Accepted 7 April 2021; Published 22 April 2021

Academic Editor: Qingzheng XU

Copyright © 2021 Xiaoying Pan et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

A complex network is characterized by community structure, so it is of great theoretical and practical significance to discover hidden functions by detecting the community structure in complex networks. In this paper, a multiobjective brain storm optimization based on novelty search (MOBSO-NS) community detection method is proposed to solve the current issue of premature convergence caused by the loss of diversity in complex network community detection based on multiobjective optimization algorithm and improve the accuracy of community discovery. The proposed method designs a novel search strategy where novelty individuals are first constructed to improve the global search ability, thus avoiding falling into local optimal solutions; then, the objective space is divided into 3 clusters: elite cluster, ordinary cluster, and novel cluster, which are mapped to the decision space, and finally, the populations are disrupted and merged. In addition, the introduction of a restarting strategy is introduced to avoid stagnation by premature convergence. Experimental results show that the algorithm with good global searchability can find the Pareto optimal network community structure set with uniform distribution and high convergence and excavate the network community with higher quality.

1. Introduction

Complex networks can be seen everywhere in real life, such as social networks [1], communication information networks [2], biological networks [3], and computer networks [4], which usually have the characteristics of small world [5], scale-free [6], and community structure [7]. The nodes within the community are closely connected, while the nodes between the communities are sparsely connected. The study of complex networks is to abstract networks into mathematical models formed composed of points and edges between points. Detecting community structure in complex network plays an important role in understanding and analyzing the topology structure of the entire network and discovering hidden functions in the network. Related research results have been successfully applied to many fields such as terrorist organization identification [8], protein function prediction, and public opinion analysis and treatment [9].

In recent years, more and more attention has been paid to the problem of complex network community detection. Many scholars have carried out a lot of research work on community detection, regarding community discovery as a single-objective optimization problem and adopting different heuristic algorithms or approximate algorithms to optimize community-related objective functions, thereby obtaining the community structure of the network. To identify the division of sparse connections and dense connections among communities, Tasgin et al. [10] optimized the community modularity criterion Q with genetic algorithm (GA) that has good performances in large networks, such as fast speed and no requirement on the number of communities.

Pizzutiz [11] defined community scores, which are used to judge the quality of network division, and utilized Genetic Algorithm in Networks (GA-Net) to optimize a simple and effective fitness function. The function can identify the connections between groups of nodes and its sparsity and

improve the variational operator, making it consider the actual correlation between nodes, thus significantly reducing the space for studying possible solutions. Lipczak and Milios [12] proposed agglomerative clustering genetic algorithm (ACGA) based on the hypothesis that communities are small enough and the number of communities is limited. Agglomerative clustering genetic algorithm based on the ordered table of neighbors of nodes was used as a particle coding method to explore the community through global search for the cluster.

Although the above single-objective optimization algorithm has the advantage of time efficiency and can mine the network satisfying a certain objective, the network community detection in practical application often needs to take into account multiple objectives, and there may be conflicts among these objectives. In view of the above limitations of the community detection algorithm based on single-objective optimization, the community detection based on multiobjective optimization begins to be concerned.

In solving complex network community detection problems, a large number of multiobjective evolutionary algorithms have emerged. Chen et al. [13] proposed a multiobjective discrete MODTLBO/D method based on teaching-learning-based Optimization (TLBO). The algorithm adopted the multiobjective decomposition mechanism and introduced neighbor-based mutation to solve the community detection problem of complex network, thus maintaining the diversity of population and avoiding local optimization.

Gong et al. [14] proposed multiobjective evolutionary algorithm with decomposition (MOEA/D-Net), making the multiobjective optimization problem transformed into a series of single-objective optimization subproblems and then, with the help of the information of a certain number of adjacent problems, employed evolutionary algorithm to optimize these subproblems at the same time. Li et al. [15] proposed a quantum-behavior discrete multiobjective particle swarm optimization algorithm for complex network clustering, which performed well in large networks. Gong et al. [16] proposed multiobjective discrete particle swarm optimization (MODPSO), which decomposed multi-objective network clustering problems into multiple scalar problems with the decomposition mechanism, and generated different individuals with high clustering efficiency in virtue of neighbor-based turbulence operator to promote diversity. Jiang et al. [17] proposed a community detection method based on a new link prediction strategy. The operation steps of the method were as follows. Firstly, the designed link prediction strategy based on the central node was used to add and remove edges in order to enhance the community structure of the network. Secondly, the community extension strategy was adopted to detect all communities in the network. However, the proposed link prediction strategy needed to calculate the similarity of a large number of node pairs, so the algorithm was very time-consuming. Aiming at the network Gong et al. [18] with fuzzy community structure, a nondominated neighbor immune algorithm-Net (NNIA-NET) was proposed to

optimize the density of both internal link and external link and discovered the community in the network by NNIA. Zhang et al. [19] considered critical node detection based on the cascade model as a biobjective optimization problem (BCVND) proposed an effective multiobjective evolutionary approach termed as MO-BCVND to solve BCVND. In MO-BCVND, a cost-reduced population initialization strategy was raised to increase the population diversity and an adaptive local search strategy was designed to accelerate the population convergence.

Zhou et al. [20] came up with multiobjective local search (MOLS-Net) algorithm, a multiobjective optimization algorithm based on local search, and designed different local search algorithms for different objectives. A hybrid genetic algorithm with link strength-based local search strategy (HGALS) is proposed by Malhotra [21] for solving the community detection problem. The local search method presented in the algorithm is faster than the traditional modularity-based search operations.

The above multiobjective community detection optimization algorithms are all intelligent optimization algorithms which need to solve the balance between global search and local search to achieve the optimal structure division in the community detection problem. Therefore, the difficulty of multiobjective optimization lies in how to maintain population diversity and avoid falling into local optimization. The brainstorm algorithm searches for the local optimal with the clustering idea, obtains the global optimal through comparing the local optimal results, and increases the diversity of the algorithm by adding the mutation operation.

To solve the problem of high complexity and slow speed of Gaussian, we put forward multiobjective brainstorming community detection methods based on novelty search, in which, by constructing novelty individuals, the objective space is divided into elite clusters, ordinary clusters, and novelty clusters and then mapped to the decision space. Novelty clustering took the individual with the smallest *NMI* value as the novelty solution to generate offspring through the cross fusion of the elite solution and the novelty solution. The novelty search mechanism has high search efficiency and excellent global optimization capabilities and adopts the restart strategy to help individuals escape from the local optimum and avoid premature convergence. "Premature" and stagnation form a single optimal solution, which improves the global search ability of the population. An external archive was established to save the Pareto optimal solution. Nondominated sorting was carried out on the population in each iteration, and the nondominated solution stored in the external archive was updated to finally get a set of Pareto optimal solutions. The brainstorming optimization algorithm has not been used to solve the problem of community detection. Therefore, the application of novelty search-based multiobjective brainstorming optimization algorithm to complex network community detection problems can better balance diversity and convergence and search for better community structure division, which is also the research motivation of this paper.

1.1. Section Arrangement. The rest of this paper was arranged as follows. In Section 2, the community detection problem, related work for multiobjective community detection, and brainstorming algorithms are introduced. Section 3 designs a multitarget brainstorming community detection method based on novelty search and gives a detailed algorithm flow and introduction. Section 4 shows experiments conducted on MOBSO-NS on four real network datasets by comparing them with the classic methods and presents detailed information and empirical results of the proposed algorithms. Section 5 summarizes the conclusions and discussed future work.

2. Related Works

2.1. Multiobjective Optimization Problem. The problem of multiobjective optimization is to find a compromise solution among the conflicting objectives through the mutual restriction of decision variables. The solution can simultaneously make multiple subobjective functions as optimal as possible. It can be modeled as the following equation:

$$\begin{aligned} \min F(X) &= [f_1(x), f_2(x), \dots, f_k(x)]^T \\ \text{s.t. } g_i(x) &\leq 0, i \\ h_{i=0}, i &= 1, 2, \dots, q, \end{aligned} \quad (1)$$

where $x = (x_1, x_2, \dots, x_m) \in \Omega$ represents the m dimension decision vector, where m means the dimension of the decision vector and Ω is the feasible region of the decision space. Multiobjective optimization problem is to optimize the function vector $\min F(X) = \min[f_1(x), f_2(x), \dots, f_k(x)]^T$, where k is the number of objective functions. The objective function $F(X)$ is composed of k functions mapped from the feasible region of the decision space to the objective space, where $g_i \leq 0, i = 1, 2, \dots, p$, is p inequality constraints, and $h_i = 0, i = 1, 2, \dots, q$, is q equality constraints.

The following are some important definitions of a domination-based multiobjective evolutionary algorithm.

Definition 1 (Pareto dominance). In the decision vector $X_A, X_B \in \Omega$, if and only if $i = 1, 2, \dots, k$, it is arbitrary, X_A is dominant X_B , and $f(x)$ dominant relation is consistent with x dominant relation;

Definition 2 (Pareto optimal solution). Assuming that there is a decision variable x^* in the feasible decision space Ω , if there is no decision variable $x \in \Omega$ like $x > x^*$, the decision variable x^* is called the Pareto optimal solution.

Definition 3 (Pareto optimal solution set). The set consisting of all noninferior optimal solutions is called the optimal solution set of the multiobjective optimization problem, also known as Pareto optimal solution set.

2.2. Multiobjective Community Detection. A concrete network can be modeled as an undirected graph $G = (V, E)$, where V is a set of nodes and E is a set of edges connecting two elements in V . If the network is composed of N nodes,

the graph can be represented by an $N \times N$ adjacency matrix A , and the undirected graph shown in Figure 1 is represented by a symmetric adjacency matrix. Its adjacency matrix A is a symmetric matrix. When $A_{ij} = 1$, it means that there is an edge connection between node I and node J , and when $A_{ij} = 0$, it means that there is an infinite connection between the two nodes.

Radicchi et al. [22] gave the definition of community in an undirected network. Supposing there is a subgraph $S \subset G$ and node i is a node in the subgraph S , the degree of node i is defined as $K_i(S) = K_i^{\text{in}}(S) + K_i^{\text{out}}(S)$. $K_i^{\text{in}}(S) = \sum_{j \in S} A_{ij}$ represents the internal degree of node i , that is, the number of edges between node i and other nodes in subgraph S . $K_i^{\text{out}}(S) = \sum_{j \notin S} A_{ij}$ represents the external degree of node i , namely, the number of node edges outside node i and subgraph S . Communities are obtained by categorizing the structural information of nodes in the network, so a community is a group of nodes, in which the connections between nodes in the same community are close, while the connections between communities are relatively sparse.

In the MOEA/D-net algorithm [19], the module density function D , as shown in formula (2), is decomposed into two different aspects of a society and are divided into 2 evaluation functions: Ration Association (RA) and Ration Cut (RC). In order to convert them into a minimization problem, the function Ration Association (RA) invert gets Negative Ratio Association (NRA), and the optimal solution is obtained by minimizing the NRA and RC set:

$$D = \sum_{i=1}^k \frac{L(V_i, V_i) - L(V_i, \bar{V}_i)}{|V_i|}, \quad (2)$$

where k is the number of community, $I \in \{1, 2, \dots, m\}$, V_i is the set of all nodes within community i , \bar{V}_i is the set of nodes connected externally by community i , $|V_i|$ is the number of internal nodes within community i , $L(V_i, V_i)$ is the number of edges within community i , and $L(V_i, \bar{V}_i)$ is the number of edges between community i and external nodes.

The ratio of community internal connection NRA and the ratio of community external connection RC are taken as the objective function. RC is the sum of the density of connections between communities, and NRA is the sum of the density of connections between nodes in the community, as shown in formula (3), where $V_i \in P, \bar{V}_i = P - V_i$, $L(V_i, V_i) = \sum_{i,j \in V_i} A_{ij}$, $L(V_i, \bar{V}_i) = \sum_{i \in V_i, j \in \bar{V}_i} A_{ij}$:

$$\min \begin{cases} \text{NRA} = -\sum_{i=1}^m \frac{L(V_i, V_i)}{|V_i|}, \\ \text{RC} = \sum_{i=1}^m \frac{L(V_i, \bar{V}_i)}{|V_i|}. \end{cases} \quad (3)$$

Through minimizing the NRA and RC functions, communities are found with dense internal connections and sparse internal connections. The optimization of RC function can reduce the number of communities and increase the number of nodes in the community. These two objective functions can balance the effect of reducing or increasing the number of communities.

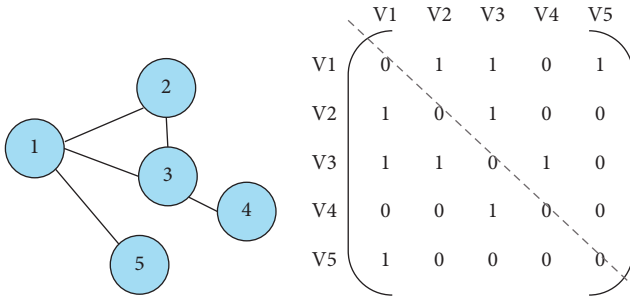


FIGURE 1: Adjacency matrix represents an undirected graph.

2.3. Brain Storm Optimization Algorithm. Brain storm optimization algorithm (BSO) is a random optimization algorithm based on human brain storm process where a group of people with different backgrounds are gathered to brainstorm on the same problem, propose a large number of ideas for the problem to be solved, and finally obtain the optimal solution through mutual communication and thought fusion. It has novel ideas and strong global searching capability. In the algorithm, the search strategy based on grouping operation enhances the search capability, and the individual update method increases the diversity of solutions and makes use of the advantages of human intelligence in dealing with problems. Thus, it can be seen that it is a very potential algorithm, which can produce results beyond the classical intelligent algorithm.

At present, the brain storm optimization algorithm has been successfully applied to solve complex, nonconvex, and NP-difficult problems with highly correlated variables. The brainstorming optimization algorithm clusters information with the clustering method. The clustering center in each class is the optimal value of the class. The cooperation among the classes and the perturbation of the clustering operation make the algorithm jump out of the local optimal and carry out the global search, thus ensuring the convergence performance of the algorithm through the optimization process of the clustering center.

The process of K-means clustering adopted by BSO algorithm is relatively complex. For this reason, Shi [23] proposed a clustering-based brain storm optimization algorithm in objective space (BSO-OS). After the objective space is sorted, several optimal individuals are selected as elite groups and other individuals as ordinary groups. The algorithm has been improved on the clustering operation, which makes the algorithm faster and more time-efficient. However, because the classification method is too simple, the population diversity cannot be guaranteed during the evolution process.

Multiobjective brain storm optimization (MOBSO) saves Pareto optimal solutions by setting the strategy of archive set and eventually obtains a set of uniform solutions close enough to the optimal frontier. MOBSO can solve the optimization problem of the two objectives well and be applied to practice. However, it is found that MOBSO increases the algorithm complexity and reduces the operation efficiency due to clustering and variation, while the increase of objective function has led to the evolution slowdown and

the low efficiency of diversity maintenance strategy. Therefore, it is necessary to redesign the objective space and design the novelty search mechanism to settle the above problems. Section 3 focuses on the novelty search mechanism.

3. Multiobjective Brain Storm Community Detection Method Based on Novelty Search

3.1. Novelty Search Mechanism. In each generation of evolutionary agents, traditional evolutionary algorithms, with a method of a single population search, select the best performing individual to produce offspring according to some metric, but this single selection will cause a loss of diversity. To solve this problem, a multipopulation parallel search based on elite, ordinary, and novelty clusters is used to enhance the diversity of the algorithm. In novelty searching, though the novelty individual is not based on the optimal fitness in the target search, it is the solution far away from the best individual, so it is selected as the starting point for further evolution in this mechanism. This search mechanism can correct for the loss of diversity and stagnation of evolution in a single evolutionary population.

Figure 2 shows the schematic diagram of novelty search mechanism, in which point A represents the local best, point B represents the global best, and point C represents the novelty solution. Obviously, between point C and the global optimal point B is less than that between point C and the local optimal point A. In the iteration process, the novelty searching mechanism searches from different positions and selects the solution opposite to the current optimal fitness as the novelty solution, which ensures the balance of population diversity and convergence to a great extent and has good robustness.

According to the above mechanism, in complex network community detection, traditional evolutionary algorithms often choose the maximum value of *NMI* as the starting point of continuous evolution, while here, the most novel individual, namely, the minimum value of *NMI*, is selected as the novelty solution by maximizing novelty measurement. Under the specific operation, the external archive set (storing elite solutions) and the current generated population are utilized to perform nondominant ranking, the *NMI* value of the solution obtained from the external archive is compared with that of the original population, and the one with the lowest *NMI* value is taken as the novelty individual. In the individual renewal strategy, two new solutions are generated as the elite solution and the novel solution by the two-point cross fusion of the external archived solution.

3.2. The Overall Flow of the Algorithm. In this paper, a multiobjective brain storm optimization based on novelty search (MOBSO-NS) community detection method is proposed to design a novelty search strategy, where the objective space is divided into elite cluster, the common cluster, and novelty cluster, among which, the novelty cluster is the smallest among the individuals with the *NMI*

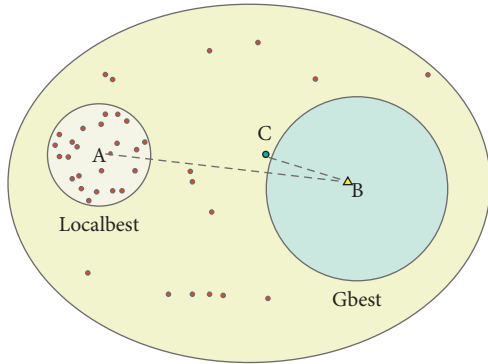


FIGURE 2: Novelty search mechanism.

value as a novelty solution. At present, many swarm intelligence optimization algorithms replace traditional selection models with mutation methods to generate new individuals, which increases the diversity of information, prevents the algorithm from falling into local optimum, and enhances the algorithm's global search ability. In MOBSO-OS, the offspring are generated by the cross-fusion of elite solutions and novel solutions, which effectively maintains the diversity of the population. As the search process progresses, the individual solutions gradually get better, making the gap between each solution smaller. At this point, even if the interaction of individuals in the population is used to reach the optimal solution, the population may fall into the local optimal solution. The strategy of restarting is used to help individuals escape from the local optimal point, avoid "precocity" caused by premature convergence, form a single optimal solution, and improve the global searching ability of the population. The external archive is set to save Pareto optimal solution. In each iteration, the nondominant solution is sorted into the nondominant solution in the external archive and then updated to finally obtain a set of Pareto optimal solutions.

The flowchart of multiobjective brainstorm community detection method based on novelty search is presented in Figure 3.

Steps of MOBSO-NS algorithm for detecting complex network communities:

Step 1: read the input network and initialize the algorithm by means of the neighbor node-based LAR coding, as shown in Figure 4.

Step 2: randomly generate the initial $popNum$ solutions and calculate the NRA and RC values of the initial solutions by formula (3).

Step 3: update the external archive EP including all the solutions in the population.

Step 4: elite individuals are disturbed. An individual $C1$ is randomly selected from the external archive EP to generate new individuals.

Step 5: obtain the novelty solution, calculate the NMI value between the solution in the external archive EP and the solution of the original population, and take the solution with small NMI value as the novelty solution.

Step 6: randomly select individuals $C1$ and $C2$ from the external archive and the current population to generate new individuals by two-point crossover fusion of elite solution and ordinary solution to generate new individuals.

Step 7: randomly select individuals $C1$ and $C2$ from external archives and novelty solution to generate new individuals by the cross fusion of elite individuals and novel individuals.

Step 8: calculate the NRA and RC values for the new population and update the external archive.

Step 9: when the external archive Q is not updated or the number of iterations p is reached, the restart operation is performed, returning to step 2.

Step 10: determine whether the termination condition is met. If so, calculate the Q value and the maximum NMI value of the external archive, otherwise, return to step 4.

Step 11: output a set of Pareto frontiers, that is, a set of partition network structures.

3.3. Coding Method Based on Neighbor Node. The encoding method based on neighbor nodes is adopted in this paper. All nodes in the graph are first numbered, then the neighbors of each node are sorted according to their numbers, and eventually the neighbor ordered table is obtained. In this coding method based on neighbor node, nodes located in the same connected subgraph are divided into a community so that each individual is valid and no illegal solution will occur, thus ensuring the convergence of the algorithm, and there is no need to know the number of partitioned communities in advance, which can be obtained through the decoding process.

The encoding method based on neighbor nodes consists of four steps: determining the neighbor set of each node, selecting the neighbor of each node, constructing the encoding list, and carrying out the decoding process. Taking Figure 4 as an example, the process is as follows.

3.3.1. Determine the Neighbor Set of Each Node. In a network topology, all nodes to which each node is connected via an edge are called the set of neighbors of that node. Figure 4(a) shows the topological structure of 11 nodes. Taking node 7 as an example, the number of the nodes it connects are 4, 6, and 8, respectively, that is, the neighbor set of node7 is {4, 6, 8}.

3.3.2. Select the Neighbor of Each Node. Each node will randomly select a neighbor from the neighbor set as the neighbor of the current node. If the corresponding gene value of the No. i node is j , it can be interpreted as there is an edge connection between node i and node j . According to the seventh node with ID 7 in Figure 4(b), the gene value is taken as 8. In the corresponding figure, there is an edge from between nodes 7 and 8.

3.3.3. Construct the Encoding List. The list of each node and its randomly selected neighbors is defined as the encoding list, as shown in Figure 4(b). Position refers to the node

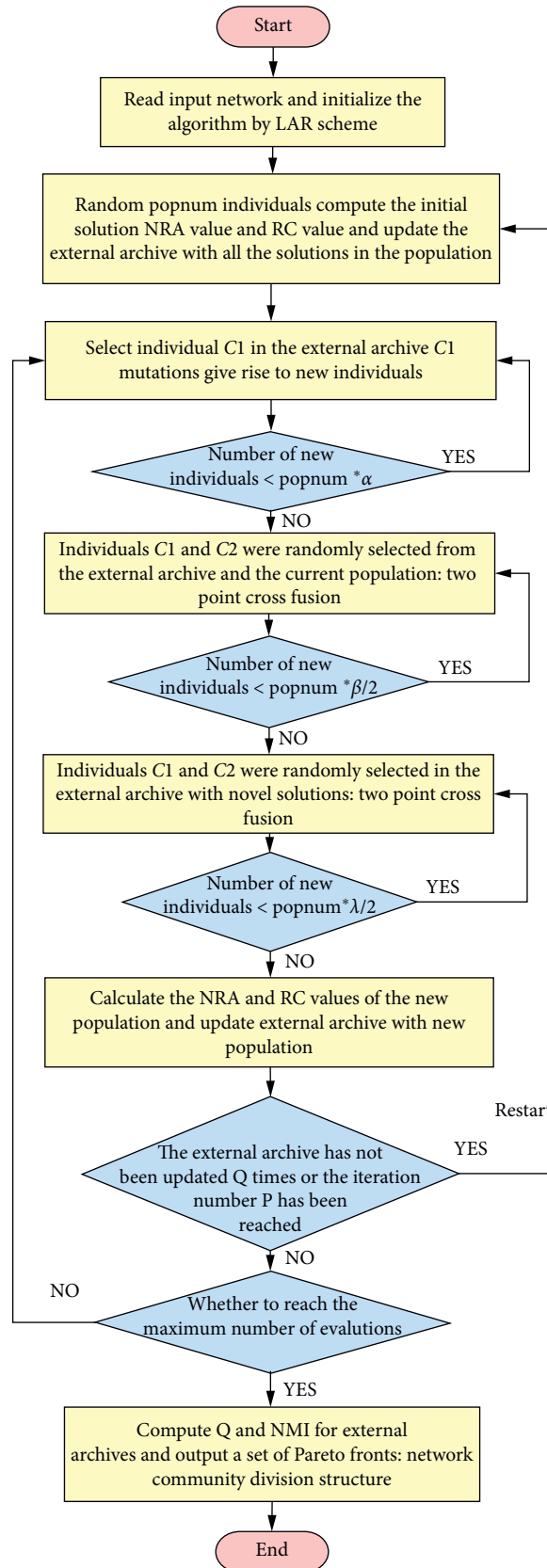


FIGURE 3: Flowchart of complex network community detection.

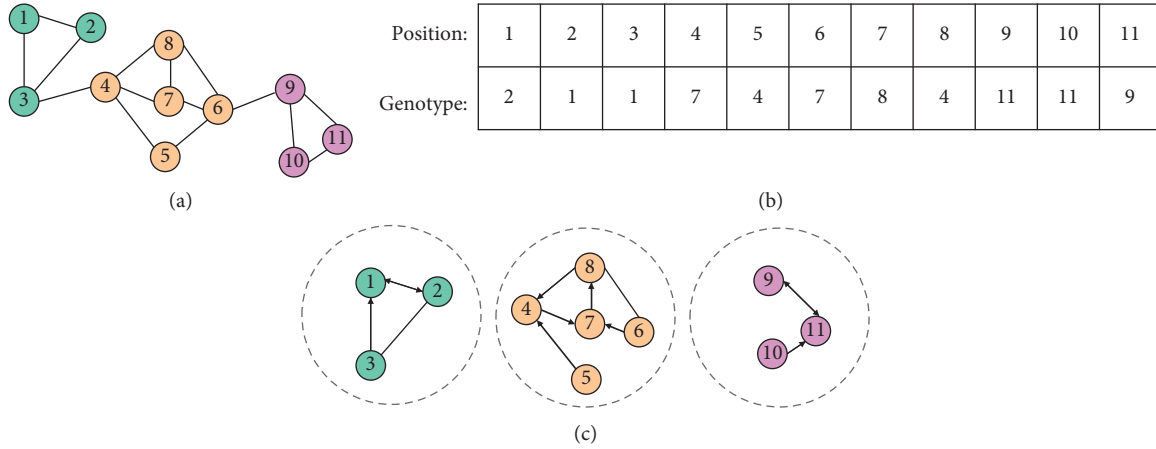


FIGURE 4: Encoding and decoding modes based on neighbor nodes. (a) A network with 11 nodes. (b) Encoding method based on neighbor nodes. (c) Decoded connected subgraph.

position, and genotype means the gene bit formed by randomly selected neighbors of each node.

3.3.4. *Decoding Process.* Decoding is the process of converting the encoded list into the community structure corresponding to the figure. All community divisions need to be identified during the decoding process, as shown in Figure 4(c). The network is divided into three communities, $\{1, 2, 3\}$, $\{3, 4, 6, 7, 8\}$, and $\{9, 10, 11\}$.

3.4. *Update the External Archive.* An external archive is set up in the algorithm to save the nondominant solution obtained by the algorithm after searching. After the objective value of the individual is calculated, each individual needs to be compared with other individuals in the population to determine whether it is a nondominant solution, that is, the more solutions stored in the external archive are, the more representative the Pareto frontier becomes.

In this paper, the solutions in the population and the solutions in the external archive are sorted in a nondominant way to update the external archive. In the iteration search phase, if a new nondominant solution is more dominant than the elements in the library, the elements in the library will be removed from the library, and if the nondominant solution in the newly generated population is dominated by some members of the library, the nondominant solution cannot enter the library.

3.5. *Restart Policy.* In the process of population iteration, the failure of an individual to constantly update its position means that the individual has been in the best position. If the individual is not helped to escape from this point in time, as the group evolves, more and more individuals will gather around the optimal point. If the greatest advantage, for the local optimal point, is "premature" phenomenon in a group, the ability to explore new global optimal solution will be lost. In order to overcome this defect, MOBSO-NS algorithm adopts a restart strategy to help individuals jump

out of the local optimal in time and maintain global search ability of the group.

In Algorithm 1, the restart judgment conditions are divided into the following two points:

- (1) In the iteration process, if the external nondominant population is the same for Q times continuously, the population is considered to converge to the local optimal when Q times are not renewed. In this paper, Q is referred to as the maximum number of times the external archive has not been updated, and the value of Q is determined based on different datasets.
- (2) P is the maximum number of iterations to restart. If the number of iterations reaches P , the restart strategy will be executed.

3.6. *Multiobjective Brain Storm Optimization Community Detection Method Based on Novelty Search.* The framework of multiobjective brain storm optimization community detection method based on novelty search is shown in the figure below, where $Pops$ is the initial population list, EP is the external archive, $tempRestart$ is the number of times that the external archive has not been updated, and $tempIter$ is the number of iterations. The overall algorithm framework of MOBSO-NS is shown in Algorithm 2.

Individual renewal adopts three strategies: first, elite cluster generates part of the new population through mutation; second, elite cluster and ordinary cluster are fused to generate part of the new population; and third, elite cluster and novelty cluster are fused to generate part of the new population to maintain the population diversity. The specific process is shown in Algorithm 3.

4. Experiment and Analysis

4.1. *Dataset.* Comparing with other related methods, the MBSO-OS algorithm was experimentally analyzed in four real networks including Zachary's karate club, the Bottlenose dolphins, the American college football, and the books about US Politics by comparing with other related methods.

- (1) **If** $tempRestart == Q$ or $tempIter == P$ **do**:
- (2) go to Algorithm 2 step 3

ALGORITHM 1: Restart policy.

Input: the edge set data of complex network, the maximum number of iterations of the algorithm $maxIter$, the population size $popNum$, and the proportion of the three new individuals generating the new population are different α, β, γ ($\alpha + \beta + \gamma = 1$); Q is the maximum number of times that the partial archive has not been updated, and P is the maximum number of iterations restarted.

Output: a group of network communities are divided into structures.

- (1) Initialize: $Pops = \emptyset, EP = \emptyset$
- (2) **For** $i = 1$ to $popNum$ **do**:
- (3) The LAR code is used to randomly generate an initial solution s , as shown in Figure 4, and the NRA and RC values for s are calculated according to formula (2).
- (4) Add s to $Pops$.
- (5) **End For**
- (6) Iterative search:
- (7) **For** $iter = 1$ to $maxIter$ **do**:
- (8) **For** s in $popNum$ **do**:
- (9) **If** no solution in EP can dominate s **do**
- (10) Add s to EP , and remove all solutions in EP that can be dominated by s .
- (11) **End If**
- (12) **End For**
- (13) $newPops = \emptyset$.
- (14) Individual update
- (15) Restart
- (16) **End For**
- (17) Calculate the indicator value:
- (18) **For** s in EP **do**:
- (19) Calculate the Q value and NMI value of s .
- (20) **End For**
- (21) Return the two solutions $S1$ and $S2$ a maximum value of 0 and NMI in EP

ALGORITHM 2: MOBSO-NS.

//The elite cluster generates part of the new population through mutation

- (1) **For** $i = 1$ to $popNum * \alpha$
- (2) Randomly select a solution $P1$ from EP , and then perform a mutation operation to generate a new solution $C1$.
- (3) Calculate the NRA and RC values of $C1$ and add $C1$ to $newPops$.
- (4) **End For**
- //Elite clusters and ordinary clusters are fused to generate a new population.
- (5) **For** $i = 1$ to $popNum * \beta / 2$
- (6) Randomly select a solution $P1$ and $P2$ from EP and $Pops$, and perform the fusion operation (two-point crossover) operation
- (7) Generate two new solutions $C1$ and $C2$.
- (8) Calculate the NRA and RC values of $C1$ and $C2$, and add $C1$ and $C2$ to $newPops$.
- (9) **End For**
- //Elite clusters and novelty clusters merge are fused to generate some new populations.
- (10) **For** $i = 1$ to $popNum * \lambda / 2$
- (11) Randomly select a solution $P1$ from the EP , calculate the NMI value of all solutions in $P1$ and $Pops$, and select the $P2$ corresponding to the solution with the smallest NMI value. Perform a fusion operation (two-point crossover) into two new solutions $C1$ and $C2$.
Calculate the NRA and RC values of $C1$ and $C2$, and add $C1$ and $C2$ to $newPops$.
- (12) **End For**

ALGORITHM 3: Individual update strategy.

The network of Zachary's karate club represents a social relationship between 34 members of a university karate club in the United States, where each node represents a member, and the edge between two nodes means that the corresponding two members are friends in frequent contacts. Due to the differences between the club management and the coaches, the club was finally divided into two: the coach-centered team was split from the club, forming a new club, while the manager-centered team remained at the club. The network has a total of 34 nodes and 78 links.

The dolphin network is an animal social network constructed by Lusseau by observing 62 dolphins of different genders in New Zealand's Doubtful Bay. Each node in the network represents a dolphin. If two dolphins are closely connected, there will be an edge connection between the corresponding vertices of the dolphins that are naturally divided into two communities: male and female groups.

The network of the American college football has a total of 77 nodes with 121 edges. The nodes in the network represent football teams, and the edge between the nodes indicates that a game has already been played between the two teams.

The political books' network edited by Krebs consists of 105 nodes and 441 edges. The nodes represent books on American politics from Amazon, and the edges between the two nodes indicate that the two books are frequently purchased together. Newman divided the books into different parts in light of the political views of the book, with a few exceptions.

Table 1 shows the number of nodes and edges of the four real networks.

4.2. Introduction to Evaluation Indexes. Currently, the most widely used community quality evaluation index is the Q value function developed by Newman and Girva [24]. The modularity of Q is defined as follows:

$$Q = \sum_{i=1}^K \left[\frac{e_i}{m} - \left(\frac{d_i}{2m} \right)^2 \right]. \quad (4)$$

By studying the related problems of complex network clustering based on optimization methods, the algorithm selects the two most common community quality evaluation indicators (Q , NMI) as the objective function.

The standard of modularity is a measure of the degree of goodness of the identified communities in the network. It is considered that the larger the Q value, the stronger the community structure. The modularity is defined as the score of the edges falling into the community minus the expected probability of random allocation of these edges, and the edges are randomly added to the network, independent of the community structure. k is the number of clusters found in the network, e_i is the total number of edges connecting nodes in cluster i , d_i is the sum of nodes in cluster i , and m is the total number of edges in the network. The standard of modularity is generally within the range of $[-0.5, 1]$, whereas most practical networks have a modularity value within the

range of $[0.3, 0.7]$, where a value greater than 0.3 indicates an important community structure.

Normalized mutual information (NMI) measures the similarity between the actual classification of societies and the detected societies. According to information theory, normalized mutual information $NMI(A, B)$ is defined as follows:

$$NMI(A, B) = \frac{-2 \sum_{i=1}^{C_A} \sum_{j=1}^{C_B} C_{ij} \log(C_{ij}n / C_i C_j)}{\sum_{i=1}^{C_A} C_i \log(C_i/n) + \sum_{j=1}^{C_B} C_j \log(C_j/n)}, \quad (5)$$

Set two partitions of the network as A and B , and let C represent the mixed matrix and its element C_{ij} the number of nodes that appear in communities $A_i \in A, B_j \in B$.

Among them, C_A and C_B , respectively, signify the number of communities in the partition of A and B , C_i represents the sum of the elements in matrix C , over row i , C_j indicates the sum of elements in matrix C , over column j , and n is the number of nodes in the network. The value of NMI ranges from 0 and 1. If A and B are completely consistent, the maximum value of NMI is 1, while if A and B are completely inconsistent, for example, the whole network is detected as A community, the minimum value of NMI is 0.

4.3. Parameter Setting. In order to verify the detection performance of the algorithm proposed in this paper, python3.7 was used to program the algorithm under the MAC OS X environment. The performance of the algorithm was simulated using real networks, respectively, and the test results were measured with standardized normalized mutual information. The maximum number of iterations of all datasets is set as 160, and the population number of $popNum$ is set as 100 for each dataset. The objective space is divided into elite cluster, novelty cluster, and ordinary cluster. The proportions of new population α generated by mutation of elite cluster, new population β generated by the fusion of elite cluster and ordinary cluster, and new population λ generated by the fusion of elite cluster and novelty cluster are set as 0.5, 0.4, and 0.1, respectively, and the EP (external archive has not been updated) restart parameter Q is 5, and the number of iterations restart parameter P is 20.

4.4. Test Results and Analysis

4.4.1. MOBSO-NS Test Results. For each network, the algorithm runs 30 times. After each run, the best Q value and NMI value are selected for segmentation and recording. After 30 runs, the average is taken between the best Q value and NMI each time. The results are shown in Table 2. From the results of Zachary's karate club network and Bottlenose dolphins' experiments, it can be seen that the MOBSO-NS NMI value reaches the maximum value of 1, which indicates that the results of the community detected by the algorithm are the same as the real community division. The American College Football also has an NMI value of close to 1. For the Q value, the four network division results are between 0.3 and 0.7.

TABLE 1: Network in the experiment.

Networks	Nodes	Edges
Karate	34	78
Dolphins	62	159
Football	77	121
Political books	105	441

TABLE 2: MOBSO-NS experimental results.

Network	Karate	Dolphin	Football	Political books
NMI_{max}	1	1	0.9367	0.6287
NMI_{avg}	1	1	0.9229	0.6101
Q_{max}	0.4198	0.5220	0.6044	0.5269
Q_{avg}	0.4198	0.5220	0.6023	0.5269

4.4.2. *Parameter Analysis.* There are three individual generation operations in the algorithm, involving three parameters α , β , and λ , among which, α is the proportion of the offspring generated by the disturbance of the elite individual, β is the proportion of the new population generated by the fusion of the elite cluster and the ordinary cluster, and λ is the proportion of new populations generated by the fusion of elite clusters and novelty clusters. In order to ensure the validity of the experimental results, a higher weight should be assigned to the elite clusters, with ordinary clusters and novel clusters as an aid to maintain the diversity of the population, which means the value of α should be above 0.5. In this paper, the parameter α starts from 0.5 and gradually increases to 0.8 in steps of 0.1, β corresponds to 0.4, 0.2, 0.2, and 0.1, and λ is 0.1, 0.2, 0.1, and 0.1. The rest parameters remain unchanged. The statistics of NMI_{max} , NMI_{avg} , Q_{max} , and Q_{avg} are performed on the test results as shown in Tables 3–5. Figures 5–8 are the NMI_{avg} line graphs of the values of the three parameters α , β , and λ for four datasets: Zachary’s karate club, the Bottlenose dolphins, the American college football, and the political books.

From the experimental results in Tables 3–5, it can be seen that although the values of Q_{max} and NMI_{max} are higher than those in Table 2 on individual datasets, the maximum Q values and maximum NMI values that appear due to the randomness of the algorithm have no reference value, and their mean values are both less than those in Table 2. Therefore, the experimental results reach the best when α is 0.5, β is 0.4, and λ is 0.1.

As can be seen from Figures 5–8, among the four sets of different values of α , β , and λ , only when α is 0.5, β is 0.4, and λ is 0.1, NMI_{max} , NMI_{avg} , Q_{max} , and Q_{avg} performed the best in four evaluation indexes. From the experimental results of the karate network, it can be found from the analysis in Figure 5 that NMI is 1 regardless of the value of α . This is because the size of the karate network is very small, α is the proportion of offspring generated by elite individual disturbance, and the changes of α , β , and λ have no effect on NMI . For the network, only when α is equal to 0.5, the result is the best, while when α is equal to 0.4, the NMI value decreases, and the scale of the network is larger than karate network, as shown in Figure 6. In Figures 7 and

TABLE 3: The value of parameter α is 0.6, β is 0.2, and λ is 0.2

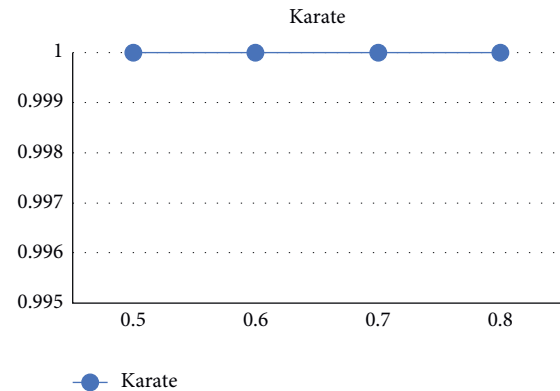
Network	Karate	Dolphins	Football	Political books
NMI_{max}	1	1	0.9269	0.6310
NMI_{avg}	1	1	0.9058	0.5985
Q_{max}	0.4198	0.5222	0.6044	0.5269
Q_{avg}	0.4198	0.5195	0.5984	0.5259

TABLE 4: The value of parameter α is 0.7, β is 0.2, and λ is 0.1

Network	Karate	Dolphins	Football	Political books
NMI_{max}	1	1	0.9269	0.6313
NMI_{avg}	1	0.9974	0.9102	0.6018
Q_{max}	0.4198	0.5238	0.6044	0.5269
Q_{avg}	0.4195	0.5200	0.6000	0.5264

TABLE 5: The value of parameter α is 0.8, β is 0.1, and λ is 0.1

Network	Karate	Dolphins	Football	Political books
NMI_{max}	1	1	0.9361	0.6287
NMI_{avg}	1	0.9963	0.9094	0.5972
Q_{max}	0.4198	0.5268	0.6046	0.5269
Q_{avg}	0.4198	0.5211	0.6001	0.5260

FIGURE 5: Line chart in karate when α , β , and λ are given different values.

8, a similar line graph appears. When α is 0.6, β is 0.2, and λ is 0.2, the NMI value is the smallest, which is caused by the large number of nodes in football network and political books’ network.

Figures 9 and 10, respectively, show the real communities detected on Zachary’s karate club network and dolphins social network through MOBSO-NS. The different colors of the nodes indicate different communities obtained by the algorithm.

Figure 11 shows the partition structure with the highest NMI value and Q value generated by the political book network through the MOBSO-NS algorithm. It can be seen that, for the partition with the highest NMI , the algorithm generates 3 communities, which are exactly equal to the number of correct communities in the political book network. That is to say, the MOBSO-NS algorithm can find a structure that is closer to the real structure.

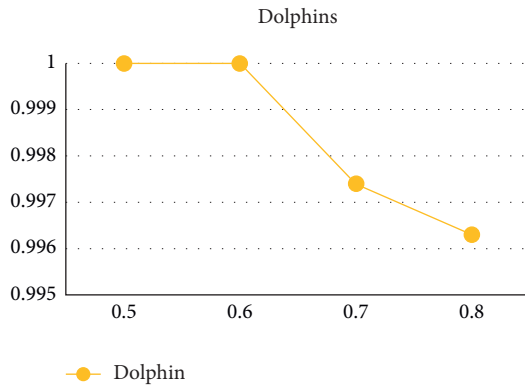


FIGURE 6: Line chart in dolphins when α , β , and λ are given different values.

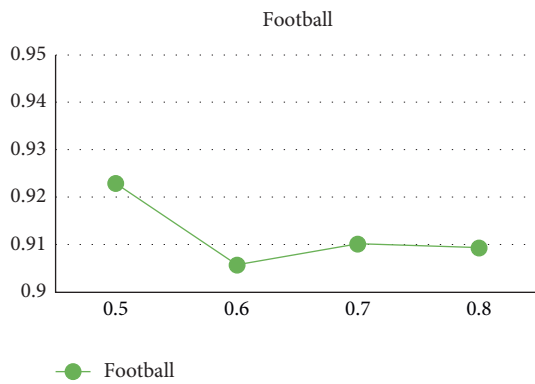


FIGURE 7: Line chart in football when α , β , and λ are given different values.

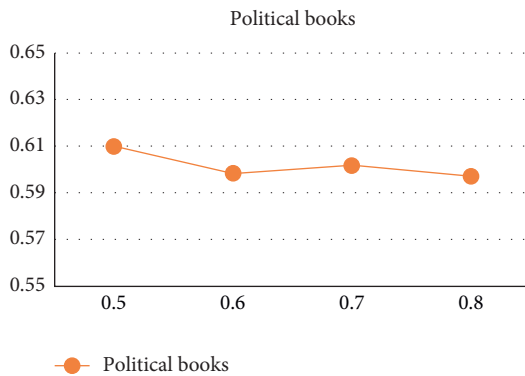


FIGURE 8: Line chart in political books when α , β , and λ are given different values.

Figure 12 shows the communities in which the MOBSONS algorithm detects the maximum NMI on the American college football network. MOBSONS produced 11 communities, but the correct number of communities in the soccer network was 12.

4.5. Comparison of Qualities of Online Communities. In order to demonstrate the advantages of MOBSONS algorithm in community detection from various aspects, the experimental

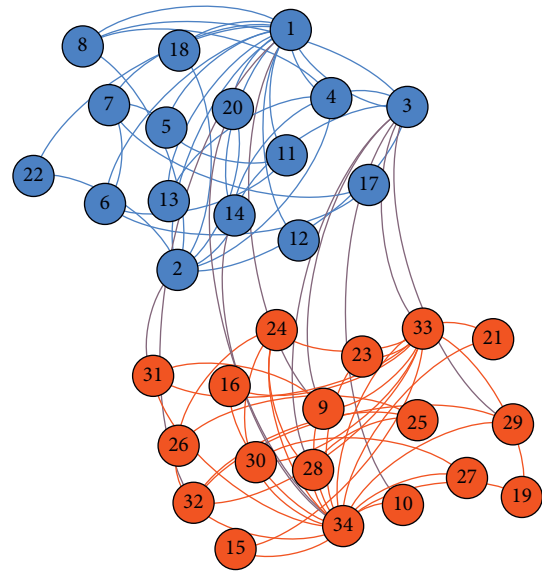


FIGURE 9: Karate.

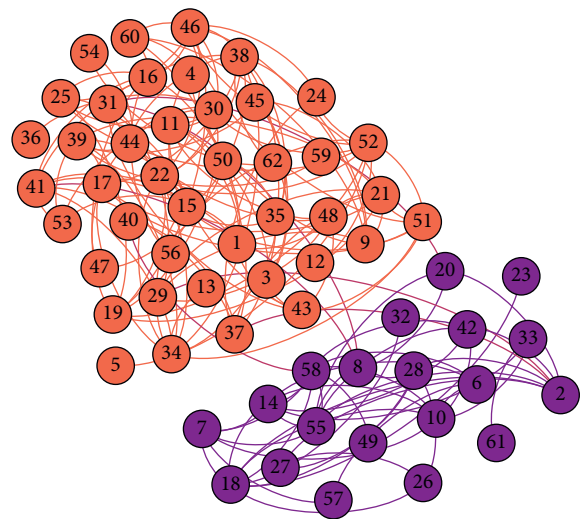


FIGURE 10: Dolphins.

data of MOBSONS algorithm, MOLS-Net, MODPSO, MOEA/D-NET, and BGLL algorithm in NMI_{max} , NMI_{avg} , Q_{max} , and Q_{avg} are compared, as shown in Table 6.

Among them, the multiobjective discrete particle swarm optimization (MODPSO) proposed by Gong et al. [15] designed a swarm optimization method specific to the problem of tag propagation, which adopted neighbor-based turbulence operators to produce different individuals and improve diversity, showing high clustering efficiency. Zhou et al. [20], proposed the multiobjective local search (MOLS-NET) algorithm to express the community detection problem as a multiobjective optimization problem and then presented a multiobjective optimization algorithm based on local search. Different target local search methods designed can optimize two targets simultaneously. The BGLL algorithm is an aggregation algorithm proposed by Blondel et al. [25] based on the concept of modularity, which can be used to analyze the

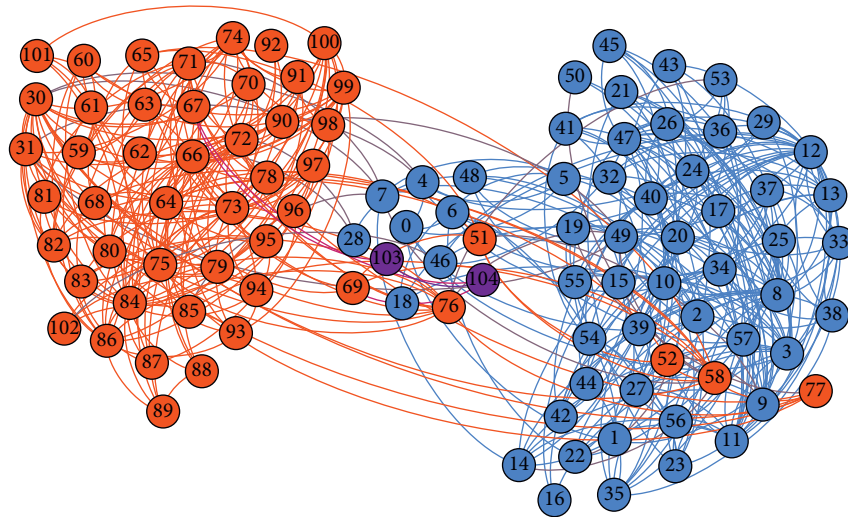


FIGURE 11: Political books.

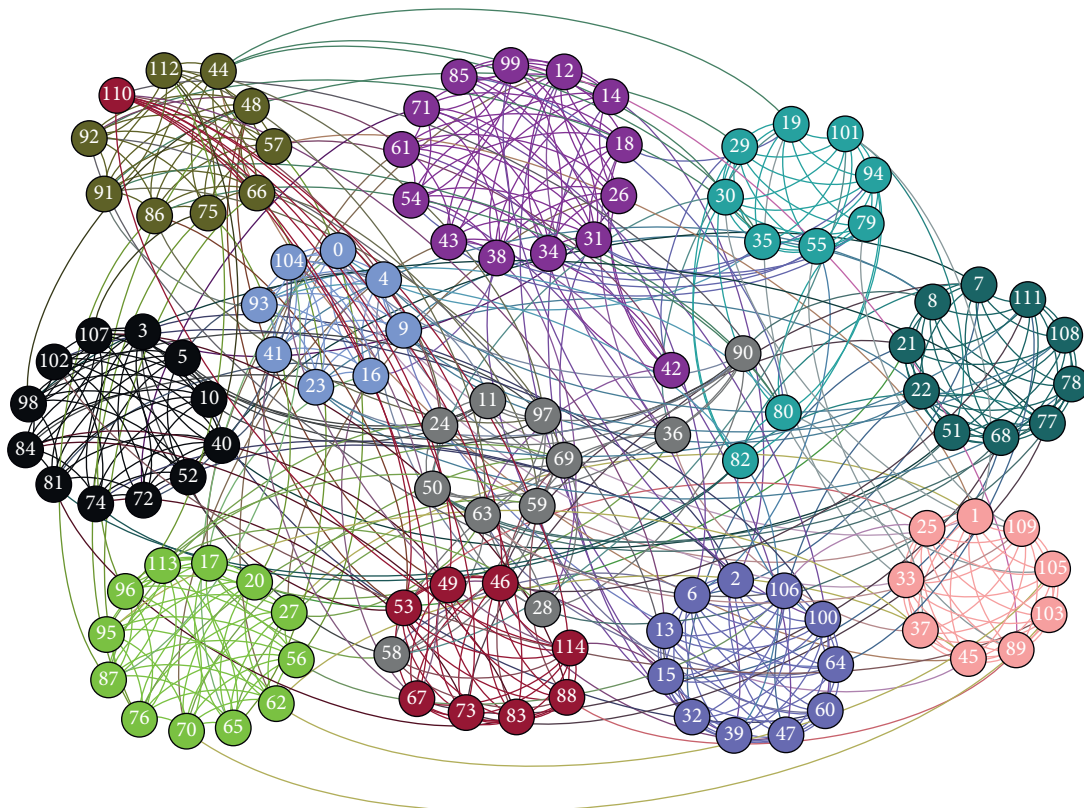


FIGURE 12: Football.

hierarchical structure of weighted networks. The MOEA/D-NET algorithm was designed by Gong et al. [14] to solve the community detection as a multiobjective optimization problem using the decomposition based multiobjective evolutionary algorithm. The proposed algorithm maximized the density of the interior degree and minimized the density of the exterior degree.

Table 6 describes the comparison results between MOBSO-NS proposed in this paper and other four algorithms on the four evaluation indexes. The network of

karate and dolphins is simple because the real network is divided into two communities. The *NMI* value given by the algorithm in this paper is the same as that of the compared algorithm, both of which are 1. This results in the community structure of the two networks being divided accurately enough. As for the *Q* value, the MOBSO-NS is not the best in the network, but the *NMI* value of 1 indicates that the divided network is exactly the same as the real network, which does not affect the experimental results.

TABLE 6: Comparison of network community quality.

Network	Index	MOBSO-NS	BSO-OS	MOLS-Net	MODPSO	MOEA/D-Net	BGLL
Karate	NMI_{max}	1	1	1	1	1	0.7071
	NMI_{avg}	1	1	1	0.9729	1	0.6280
	Q_{max}	0.4198	0.4198	0.4198	0.4198	0.3715	0.4198
	Q_{avg}	0.4198	0.4198	0.4198	0.4138	0.3715	0.4118
Dolphin	NMI_{max}	1	1	1	1	1	0.6363
	NMI_{avg}	1	1	1	1	1	0.5148
	Q_{max}	0.5220	0.5269	0.5268	0.5268	0.3735	0.5277
	Q_{avg}	0.5220	0.5269	0.5237	0.5196	0.3735	0.5210
Football	NMI_{max}	0.9367	0.9357	0.9361	0.9289	0.9367	0.8923
	NMI_{avg}	0.9229	0.9229	0.9273	0.9245	0.9334	0.8765
	Q_{max}	0.6044	0.6044	0.6046	0.6032	0.6005	0.6046
	Q_{avg}	0.6023	0.6032	0.6044	0.5995	0.5993	0.6038
Political books	NMI_{max}	0.6287	0.6254	—	0.5910	0.5895	0.4420
	NMI_{avg}	0.6101	0.5999	—	0.5910	0.5885	0.4420
	Q_{max}	0.5269	0.5259	—	0.5263	0.5259	0.5186
	Q_{avg}	0.5269	0.5259	—	0.5263	0.5259	0.5186

For Zachary's karate club network and the Bottlenose dolphins network, because the real network is relatively simple, two communities are divided. The NMI value given by the algorithm in the paper is the same as that given by the compared algorithm, both of which are 1, indicating that these two networks have been accurately divided into network community structures. As for the Q value, although the performance of MOBSO-NS is not the best performance in dolphin network, the NMI value of 1 indicates that the partitioned network is completely consistent with the real network, which has no effect on the experimental results.

The political books' network is complex, and most existing methods fail to detect the true division of the network. MOBSO-NS is the result when the actual number of communities is unknown. Compared with those of MOLS-NET algorithm with the known actual number of communities, the experimental results of MOBSO-NS algorithm are more objective and accurate. In addition, it is clear from the contents recorded in Table 6 that the experimental results in the political books' network and the community quality are much higher than those of the other four known algorithms.

In the American college football network, the real network has 12 community divisions. For the Q value, MOLS-NET is superior to all comparison algorithms. Due to the complexity of the network, no algorithm can realize the real partition structure. Other algorithms perform better than MOBSO-NS in terms of the average Q because the American college football league network is essentially a network with numerous and difficult community divisions. However, it is found in the experimental results that MOBSO-NS algorithm has the maximum NMI value compared with other algorithms, indicating that the community structure divided by the algorithm in this paper is most similar to the real network.

Compared with MOLS-NET, MODPSO, MOEA/D-NET, and BGLL, MOBSO-NS algorithm has significant advantages in terms of average and maximum value of community structure, and the similarity between the community detected by it and the actual community division is also closer. The network of Zachary's karate club and the Bottlenose dolphins,

in particular, has a strong community structure. Because the NMI values of the algorithm are uniformly distributed on each network, the algorithm has stronger robustness.

5. Conclusion

The research of complex network community detection is of great significance to Internet culture security and information personalized service. At present, most of the complex network community detection algorithms based on heuristic optimization are the strength of a single community structure quality evaluation, and the diversity of community quality evaluation indexes makes the network community structure analysis more decision-making. In this paper, a novel multiobjective brain storm community detection method (MOBSO-NS) based on novelty search is proposed to solve the problem of complex network community detection. The novelty search method can effectively avoid premature convergence and enhance the global search ability while maintaining the diversity of the population. Secondly, the restart operation is used to help individuals escape from the local optimal point. The idea of novelty search is integrated into the brainstorming optimization algorithm, and the mechanism of generating new individuals is innovated. Experimental results show that the algorithm in this paper has better optimization ability and can obtain better results of network community division.

There still remains some work related with MOBSO-NS that deserves to be further investigated. The MOBSO-NS suggested in this paper has shown that utilizing local community information is a promising idea to obtain good community partition in the static networks. In the future, we would like to combine this strategy with other frameworks of MOEA, such as NSGA-II, SPEA2, and IBEA, to further explore the local information in other kinds of complex networks, such as overlapping communities and dynamic networks. In addition, as the real data communities tend to be very large, how to improve the performance of MOBSO-NS by identifying the communities with small sizes is also an interesting work.

Data Availability

The data, models, or code generated or used during the study are available at <http://vldowiki.fmf.uni-lj.si/doku.php?id=pajek:data:urls:index>.

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 (no. 62001380) and General Special Scientific Research Program of Shaanxi Provincial Education Department (no. 20JK0910).

References

- [1] A. Asikainen, G. Iñiguez, J. Ureña-Carrión et al., “Cumulative effects of triadic closure and homophily in social networks,” *Science Advances*, vol. 6, no. 19, Article ID eaax7310, 2020.
- [2] C. Esposito, “Interoperable, dynamic and privacy-preserving access control for cloud data storage when integrating heterogeneous organizations,” *Journal of Network and Computer Applications*, vol. 108, no. APR, pp. 124–136, 2018.
- [3] K. Devkota, L. J. Murphy, and L. J. Cowen, “GLIDE: combining local methods and diffusion state embeddings to predict missing interactions in biological networks,” *Bioinformatics*, vol. 36, no. Supplement_1, pp. i464–i473, 2020.
- [4] A. Castiglione, R. Pizzolante, A. De Santis, B. Carpentieri, A. Castiglione, and F. Palmieri, “Cloud-based adaptive compression and secure management services for 3D healthcare data,” *Future Generation Computer Systems*, vol. 43–44, pp. 120–134, 2015.
- [5] D. J. Watts and S. H. Strogatz, “Collective dynamics of “small-world” networks,” *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.
- [6] A.-L. Barabási and E. Bonabeau, “Scale-free networks,” *Scientific American*, vol. 288, no. 5, p. 60, 2003.
- [7] A. D. King, N. Przulj, and I. Jurisica, “Protein complex prediction via cost-based clustering,” *Bioinformatics*, vol. 20, no. 17, pp. 3013–3020, 2004.
- [8] T. Sangkaran, N. A. Abdullah, and N. Z. Jhanjhi, “Criminal community detection based on isomorphic subgraph analytics,” *Open Computer Science*, vol. 10, no. 1, pp. 164–174, 2020.
- [9] S. Fortunato, “Community detection in graphs,” *Physics Reports*, vol. 486, no. 3–5, pp. 75–174, 2010.
- [10] M. Tasgin, A. Herdagdelen, and H. Bingol, *Community Detection in Complex Networks Using Genetic Algorithms*, Corrosion -Houston Tx-, Odessa, TX, USA, 2007.
- [11] A. I. Hafez, N. I. Ghali, A. E. Hassanien et al., “Genetic algorithms for community detection in social networks,” in *Proceedings of the International Conference on Intelligent Systems Design & Applications*, IEEE, Vellore, India, September 2013.
- [12] M. Lipczak and E. Miliós, “Agglomerative genetic algorithm for clustering in social networks,” in *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*, pp. 1243–1250, New York, NY, USA, July 2009.
- [13] D. Chen, F. Zou, R. Lu, L. Yu, Z. Li, and J. Wang, “Multi-objective optimization of community detection using discrete teaching-learning-based optimization with decomposition,” *Information Sciences*, vol. 369, pp. 402–418, 2016.
- [14] M. Gong, Q. L. Ma, and L. Jiao, “Community detection in networks by using multiobjective evolutionary algorithm with decomposition,” *Physica A: Statistical Mechanics and Its Applications*, vol. 391, no. 15, pp. 4050–4060, 2012.
- [15] L. Li, L. Jiao, J. Zhao et al., “Quantum-behaved discrete multi-objective particle swarm optimization for complex network clustering,” *Pattern Recognition*, vol. 63, pp. 1–14, 2016.
- [16] M. Gong, X. Q. Cai, and L. Ma, “Complex network clustering by multiobjective discrete particle swarm optimization based on decomposition,” *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 1, pp. 82–97, 2014.
- [17] H. Jiang, C. Z. Liu, and X. Y. ZhangSu, “Community detection in complex networks with an ambiguous structure using central node based link prediction,” *Knowledge-Based Systems*, vol. 195, p. 105626, 2020.
- [18] M. Gong, T. Hou, B. Fu et al., “A non-dominated neighbor immune algorithm for Community detection in networks,” in *Proceedings of the 13th annual Conference on Genetic and Evolutionary Computation*, pp. 1627–1634, Dublin, Ireland, July 2011.
- [19] L. Zhang, J. Xia, and C. Fan, J. Qiu and X. Zhang, Multi-objective optimization of critical node detection based on cascade model in complex networks,” *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 3, 2020.
- [20] Y. Zhou, N. J. Wang, and Z. Zhang, “Multiobjective local search for community detection in networks,” *Soft Computing*, vol. 20, no. 8, pp. 3273–3282, 2016.
- [21] D. Malhotra, “Community detection in complex networks using link strength-based hybrid genetic algorithm,” *SN Computer Science*, vol. 2, no. 1, pp. 1–16, 2021.
- [22] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi, “Defining and identifying communities in networks,” *Proceedings of the National Academy of Sciences*, vol. 101, no. 9, pp. 2658–2663, 2004.
- [23] Y. Shi, *Brain storm Optimization algorithm in Objective space*, IEEE Congress on Evolutionary Computation, Sendai, China, 2015.
- [24] M. Newman and M. Girvan, “Finding and evaluating community structure in networks,” *Physical Review. E, Statistical, Nonlinear, and Soft Matter Physics*, vol. 69, no. 2, pp. 26113–26120, 2004.
- [25] V. D. Blondel, J. L. Guillaume, R. Lambiotte et al., “Fast unfolding of communities in large networks,” *Journal of Statistical Mechanics Theory & Experiment*, vol. 10, 2008.