Research Article

Robustness of the Storage in Cloud Data Centers Based on Simple Swarm Optimization Algorithm

Alireza Chamkoori ¹ and Serajdean Katebi ²

¹Department of Computer Engineering, Yasooj Branch, Islamic Azad University, Yasooj, Iran

²Department of Electrical and Computer Engineering, Shiraz University, Shiraz, Iran

Correspondence should be addressed to Serajdean Katebi; katebi.serajdean@gmail.com

Received 8 December 2020; Accepted 4 August 2021; Published 24 August 2021

Academic Editor: Mamoun Alazab

Copyright © 2021 Alireza Chamkoori and Serajdean Katebi. 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.

Storing extensive data in cloud environments affects service quality, transmission speed, and access to information in systems, which is becoming a growing challenge. In storage improvement, reducing various costs and reducing the shortest path in the storage of distributed cloud data centers are among the important issues in the field of cloud computing. In this paper, particle swarm optimization (PSO) algorithm and learning automaton (LA) are used to minimize the cost of a data center, which includes communication, data transfer, and storage and optimization of communication between data centers. To improve storage in distributed data centers, a new model called LAPSO is proposed by combining LA and PSO, in which LA improves particle control by searching for particle speed and position. In this method, LA moves each particle in the direction where it has the best individual and group experiences. In multipeak problems, it does not fall into local optimums. Results of the experiments are shown on the dataset of spatial information and cadastre of country lands, which includes 13 data centers. The proposed method evaluates and improves the optimal position parameters, minimum route cost, distance, data transfer cost, storage cost, data communication cost, load balance, and access performance better than other methods.

1. Introduction

With the increasing growth of cloud computing and the ability to use data storage centers and computing servers in cloud computing, nowadays, many applications require infrastructure to store and process data outside the client and systems. Data storage on cloud storage centers has advantages in cloud computing, which include increased data storage and processing capacity and increased reliability since data center servers store multiple versions of data as backups. However, due to the volume of data, the optimal solution for storing this information is one of the important issues in this field [1]. In some articles in the field of storage improvement in data centers, due to the lack of a multi-objective search model, the parameters of data transfer cost, data communication, distance between data centers, location, and so on are optimized separately. This makes the relationship between the parameters in the search space to be

ignored, and multiobjective models do not optimize the search models. But in this method, in addition to optimizing the search model, the relationship between the parameters is considered. Research in this field has generally discussed the issue of improving cloud storage in the distributed data center in order to minimize costs and the shortest route, since each resource has its own data center and stores the data in another data center to ensure safety and security. Strategies for finding the minimum distance between data centers and minimizing the cost of creation, storage, transfer, and communication between data centers can be very effective in optimizing data storage in distributed cloud data centers. Recent progress in various areas such as networking, information, and communication technologies has greatly boosted the potential capabilities of cloud computing. Cloud computing is a promising computing paradigm that facilitates the delivery of IT infrastructure, platforms, and applications of any kind to consumers as services over

the Internet. Although cloud computing systems nowadays provide better ways to accomplish the job requests in terms of responsiveness and scalability under various workloads, scheduling the jobs or tasks in cloud environment is still complex in nature due to the dynamicity of resources and on-demand user application requirements [2]. Optimization algorithms are very popular swarm intelligence (SI) optimization algorithms in recent years and have been used in many fields such as machine learning, engineering, and environment modelling [3]. They have the advantages of simplicity, flexibility, derivation independency, and escaping from local optimum. Metaheuristic algorithms create randomly initialized population and improve the population during iterations to search for global optimum in the search space. The typical search process of optimization algorithm can be divided into two phases: exploration and exploitation [3, 4]. In the exploration phase, search agents investigate the search space as widely as possible to obtain the promising region, while in the exploitation phase, search is carried out in the local region obtained by exploration phase to find the global optimum solution. It is very important for an algorithm to strike a proper balance between exploration and exploitation in order to avoid local optimum and find global optimum solution quickly. Some of the most popular metaheuristic algorithms are genetic algorithm (GA), particle swarm optimization (PSO), artificial fish swarm algorithm (AFSA), and grey wolf optimization (GWO) [4]. Whale optimization algorithm (WOA) is a new metaheuristic optimization algorithm which was proposed by Seyedali Mirjalili and Andrew Lewis in 2016. WOA mimics the hunting behavior of humpback whales called bubble-net feeding method. Humpback whales prefer to hunt school of krill or small fishes close to the surface by creating distinctive bubbles along a circle or “9-shaped path” [4, 5]. Some studies have proposed to improve the performance of basic WOA and apply it into different fields [5–7]. CWOA which was proposed in [8] blends WOA with chaos theory and applies CWOA to solving transient stability constrained OPF problem. The authors in [9] proposed another CWOA to improve the diversity and egocentricity of search agents and apply the proposed CWOA to optimizing the Elman neural network. LWOA was proposed in [3] to improve the performance of WOA based on Levy flight trajectory and was applied to solving infinite impulse response model identification. To improve the exploration and exploitation abilities as well as reducing the possibility of falling into local optimum of WOA, we optimize WOA in three aspects and apply the improved WOA to the multiresource allocation problem. The main contributions of this work are described as follows:

- (i) Introducing nonlinearly changed convergence factor to preferably adjust the exploration and exploitation processes of WOA. The linearly changed convergence factor limits the exploration and exploitation abilities of WOA, which can be improved by using nonlinearly changed convergence factor.
- (ii) Adding a new inertia weight factor to adjust the influence of current best search agent on the

movements of other search agents. Consequently, the exploration and exploitation abilities and convergence speed of WOA are greatly improved.

- (iii) Conducting random variation of current best search agent during the exploitation process. In the exploitation process of WOA, all the search agents move towards the best search agent which may be a local optimum. To reduce the possibility of falling into local optimum, we conduct a certain number of variations during each iteration of the exploitation process.
- (iv) Testing IWOA with 29 benchmark functions and comparing it with other well-known metaheuristic algorithms. Four types of benchmark functions are employed to test the performance of algorithms from different perspectives, including exploration ability, exploitation ability, ability to escape from local minima, and convergence speed.
- (v) Applying IWOA to the multiresource allocation problem to evaluate the ability of solving the engineering problem. Multiresource allocation is a general engineering problem, and various kinds of metaheuristics have been proposed to solve it. IWOA and the well-known metaheuristics are benchmarked with the system utility model to evaluate the performance of solving the multi-resource problem.

In the proposed method, to improve storage in data centers, a new model called LAPSO is proposed, which combines LA and PSO. This method improves the performance of the multiobjective PSO algorithm. In the proposed model, the LA is used to regulate the behavior of the particles and, in each step, determines whether the particles continue in their current route or follow the best particles found so far. In this model, the task of establishing a balance between global search and local search is assigned to the median coefficient, which is performed by the LA. Using LA has two main advantages: first, the existing knowledge can be employed to determine the process of moderate weight changes, and second, this process can be corrected by obtaining feedback from the algorithm implementation. In the objective function, the multiobjective PSO algorithm calculates the best particles by minimizing the optimal position parameters, minimum route cost, distance, data transfer cost, storage cost, data communication cost, load balance, access, and year. The advantages of the proposed method are as follows:

- (i) In the PSO algorithm, due to the use of LA, each particle tries to move in the direction at which it has the best individual and group experiences. In the case of multipeak problems, it is updated and prevented from falling into local optimums.
- (ii) Better access to the shortest distance among the distributed data centers with minimal communication costs between data centers, data transfer, storage, and data center construction using the provided LAPSO model.

- (iii) To improve storage, this new LAPSO model is presented to improve performance in the PSO multipurpose algorithm, which is the PSO objective function. It minimizes the cost of creating the data center as well as storing, transferring, and communicating between data centers. Due to its multipurpose nature, the standard PSO algorithm does not offer a solution to leave this local optimum. But the LA prevents the model from falling into the local optimum.
- (iv) Our proposed model aims to distribute data loading to different cloud servers without creating overhead and long delays.

The rest of this paper is organized as follows. Section 2 outlines the basic WOA. Section 3 presents the improved WOA (IWOA). Experimental results and performance analysis of the proposed IWOA are provided in Section 4. Section 5 applies the proposed IWOA to solving the multi-resource allocation problem. Finally, in Section 6, conclusions and possible future works are given [10].

There are two fundamental limitations with the safe frameworks: first, the scope of companies that can be certified as a part of the program is very limited. Companies that have an interest in ineligible cloud computing include banks, telecommunication carriers, and nonprofit organizations. Second, there are extenuated conflicting legislative concerns. The scope of the safe frameworks is quite limited; this is an agreement between three jurisdictions only. Although a step in the right direction, the complicated legal environment of numerous regulating bodies and legislations still point to per-jurisdiction data centers as an attractive alternative.

2. Previous Works

In recent articles in the field of data storage in the cloud, various actions have been performed to reduce costs as well as increase the shortest path and distance by methods such as fuzzy, game theory, and evolutionary algorithms [11]. In the computational science, PSO is a computational method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. In 1995, Eberhart and Kennedy first introduced the PSO algorithm as an uncertain search method for functional

optimization. This algorithm is inspired by the mass movement of birds looking for food [1]. With the advent of cloud computing and big data, many companies are increasingly storing and retrieving their data on the cloud due to the increasing scale of data and the performance provided by cloud storage services. Data recovery duration is specifically important for cloud users. However, current data management systems have not yet been optimized to minimize this duration. Some researchers have tried to add noise to particle motions in the PSO standard algorithm, causing more varied motions to solve local optimization problems [12, 13]. Some other solutions to solving local optimization problems combine the PSO algorithm with other algorithms such as GA and gradient annealing algorithm [14, 15]. LA is a machine that can perform a variety of operations. Each action selected is evaluated by a potential environment; the evaluation result is given to the LA in the form of a positive or negative signal and the response is automatically used to select the next action. In this way, it tends to choose the action that receives the most rewards from the environment. In other words, the LA performs an action that receives the most rewards from the environment. LA has been used to improve the learning ability of many algorithms, including neural networks [16, 17], genetic algorithms [18, 19], and the mass movement of binary particles [20]. In [20], the binary PSO model was presented based on the LA. In this method, an LA is used in each particle dimension. Each of these learning automata has two operations, i.e., 0 and 1, and acts as the mastermind of the particle, controlling and directing its motion in the state space limited to 0 and 1. This model has better results than the standard binary model [21] in solving sample problems. The previous work for better observation is summarized in Table 1.

2.1. Particle Swarm Optimization Algorithm. In the PSO algorithm, a group of particles is created randomly which tries to find the optimal solution by updating the generations. In each step, each particle is updated using the two best values. The first is the best position ever reached by a particle known as p_{best} . The best position ever obtained by the population of particles is represented by g_{best} . After finding the best values, the velocity and location of each particle are updated using the following equations:

$$v = v + c1 * rand * (p_{best} - position) + c2 * rand * (g_{best} - position), \quad (1)$$

$$position = position + v \quad (2)$$

The right side of (1) consists of three parts: the first part is the current velocity of the particle, and the second and third parts are responsible for changing the velocity of the particle and rotating it towards the best individual experience and the best group experience. If the first part is neglected in this equation, then the velocity of the particles is determined only

according to the current position, the best particle experience, and the best group experience. In this way, the best particle remains in place and the others move toward that particle. In fact, the mass movement of particles ignoring the first part of (1) is a process during which the search space gradually shrinks and local search is formed around the best

TABLE 1: Summary of previous work.

Used algorithm	Paper
Fuzzy, game theory, and evolutionary algorithms	Christian et al. [11]
PSO algorithm	Kennedy and Eberhart [1]
Tried to add noise to particle motions in the PSO standard algorithm, causing more varied motions to solve local optimization problems	Riget and Vesterstroem and Kiink et al. [12, 13]
Combined the PSO algorithm with other algorithms such as genetic algorithms and the gradient annealing algorithm	Wang and Li and Afzal et al. [14, 15]
LA has been used to improve the learning ability of many algorithms, including neural networks	Meybodi and Beigy [16, 17]
Genetic algorithms	Munetomi et al. and Howell et al. [18, 19]
The mass movement of binary particles	Chan and Seung [20]
The standard binary model [21] in solving sample problems	Kennedy and Eberhart [21]

particle. Conversely, if only the first part of (1) is considered, the particles take their normal path to reach the perimeter wall and perform some sort of global search. In the model introduced in 1997 by standard algorithm providers [21], the position of each particle in each dimension is determined by one of the two values, i.e., 0 or 1. In this way, the particle moves in a space limited to 0 and 1, and the particle velocity in each dimension is equal to the probability that the

position of the particle is one in that dimension. The speed update is still performed according to (1). Then, firstly, the velocity obtained in each dimension is transferred to the interval [1,0] using the sigmoid function; then, the new position of the particle in each dimension is calculated according to equation (3), where (rand) is a random value in the range [1,0]. Also, different block diagrams for implementing RDA are shown in Algorithm 1.

$$\text{If } (\text{rand}) < \text{sigmoid}(\text{vid}(t+1)) \text{ then, } \text{xid}(t+1) = 1, \text{ else } \text{xid}(t+1) = 0. \quad (3)$$

2.2. Learning Automaton Algorithm. LA is a factor designed to be placed in a possible and uncertain environment where the machine can perform a number of finite operations. Each LA has a vector of probabilities that shows how likely each action is performed. In each operation performed by automata, the sum of the vector derivatives is equal and each of the selected operations is evaluated by a possible environment. The evaluation result is given to automata in the form of a positive or negative signal, and the automata are influenced by this response while choosing the next action. The goal for automata is to choose the best practice among all the practices. The best practice is to maximize the chances of receiving a reward from the environment [22]. Figure 1 shows the relationship between LA and the environment. The environment can be represented by $E \equiv \{\alpha, \beta, c\}$ ternary, in which $a = \{a_1, a_2, \dots, a_m\}$ is the set of inputs, $\beta = \{\beta_1, \beta_2, \dots, \beta_m\}$ is the set of outputs, and $C = \{C_1, C_2, \dots, C_m\}$ is the sum of the penalty probabilities. If β is a two-member set, the environment is of type P; in such an environment, $\beta_1 = 1$ is considered as a penalty and $\beta_2 = 0$ is considered as a reward. In the Q-type environment, $\beta(n)$ can be a discrete value of finite values at the distance of [0.1]; in an S-type environment, $\beta(n)$ is a random variable at the distance of [0.1]. C_i is the probability, by which a_i action will have an adverse effect. In the static environment, C_i values remain unchanged, while in the nonstatic environment, these values change over time. LA with a variable structure can be represented by the quadratic $\{\alpha, \beta, P, T\}$, where $a = \{a_1, a_2, \dots, a_m\}$ is the set of automata operations, $\beta = \{\beta_1, \dots, \beta_m\}$ is the set of automata inputs, $P = \{P_1, \dots, P_m\}$ is the probability vector of selecting each of the operations, and $p(n+1) = T[\alpha(n), \beta(n), P(n)]$ is the learning algorithm.

Algorithm 2 shows the pseudocode for LA with a variable structure. Algorithm 2 is an example of linear learning algorithms. Suppose a_i operation is selected in the n -th step. Optimal response and adverse response are defined according to equations (4) and (5), respectively, as

$$P_i(n+1) = P_i(n) + \alpha[1 + P_i(n)], \quad (4)$$

$$P_i(n+1) = (1-b)P_i(n), \quad (5)$$

$$P_i(n+1) = \left(\frac{b}{r-1}\right) + (1-b)P_j(n) \quad \forall j \neq i,$$

where a is the reward parameter and b is the penalty parameter. According to the values of a and b , the following three states can be considered. When a and b are equal, we call the algorithm LRP; when b is much smaller than a , we call the algorithm LR*P; and when b is equal to zero, we call the algorithm LRI. This probability update is in such a way that the sum of the probabilities remains equal to one.

2.3. A Systematic Review on Clone Node Detection in Static Wireless Sensor Networks. The recent state-of-the-art innovations in technology enable the development of low-cost sensor nodes with processing and communication capabilities. The unique characteristics of these low-cost sensor nodes such as limited resources in terms of processing, memory, battery, and lack of tamper resistance hardware make them susceptible to clone node or node replication attack. The deployment of WSNs in the remote and harsh environment helps the adversary to capture the legitimate node and extract the stored credential information such as

```

(1) for each particle do
(2)   Initialize particle
(3) end For
(4) do
(5)   for each particle do
(6)     Calculate fitness value of the particle  $f_p$ 
(7)     updating particle's best fitness value so far
(9)     If  $f_p$  is better than  $p\_Best$ 
(10)    set current value as the new  $p\_Best$ 
(11)   end for
(12)   Updating population's best fitness value so far
(13)   Set  $g\_Best$  to the best fitness value of all particles
(14)   for each particle
(15)     Calculate particle velocity according to equation (1)
(16)     Update particle position according to equation (2)
(17)   end for
(18)   while maximum iterations OR
(19)     minimum error criteria are not attained.

```

ALGORITHM 1: The different block diagrams for implementing RDA [1].

```

(1) Initialize  $p$  to  $[1/s, 1/s, \dots, 1/s]/*s$  is the number of actions*/
(2)   while not done
(3)     Select an action  $i$  based on the probability vector  $p$ 
(4)     Evaluate action and return a reinforcement signal  $\beta$ 
(5)     Update probability vector using learning rule
(6)   end while

```

ALGORITHM 2: Pseudocode for LA with variable structure [22].

```

(1) for each particle do
(2)   Initialize particle
(3) end For
(4) do
(5)   for each particle do
(6)     Calculate fitness value of the particle  $f_p$ 
(7)     Updating particle's best fitness value so far
(9)     If  $f_p$  is better than  $p\_Best$ 
(10)    set current value as the new  $p\_Best$ 
(11)   end for
(12)   Updating population's best fitness value so far
(13)   Set  $g\_Best$  to the best fitness value of all particles
(14)   for each particle
(15)     Calculate particle velocity according to equation (1)
(16)     Update particle position according to equation (2)
(17)   end for
(18)   while maximum iterations OR
(19)     minimum error criteria is not attained

```

ALGORITHM 3: Flowchart of the proposed method.

ID which can be easily reprogrammed and replicated. Thus, the adversary would be able to control the whole network internally and carry out the same functions as that of the legitimate nodes.

2.3.1. Clone Node Attack. WSNs are primarily categorized into two types i.e., static and mobile WSNs. In static WSN, once the sensor nodes are deployed, their position remains fixed compared to mobile WSN where nodes can move

FIGURE 1: Relationship between LA and the environment [22].

freely after deployment. In other words, we can say static WSNs use fixed flooding/routing for data distribution, whereas mobile WSNs use dynamic routing. Both of these categories of WSN are prone to clone node attacks.

This is the main motivation of researchers to design enhanced detection protocols for clone attacks. Clone node attack is regarded as one of the most hazardous attacks on WSNs. In a clone attack, the attacker initially targets and captures a legal node, extracts the stored credentials using some specialized tools in less than one minute [21]. The attacker then creates clones using the credentials and deploys them to several important locations of the network to carry out internal attacks like denial of service (DoS), a black hole, or even wormhole attack [23]. The attacker can then perform various actions with the help of these clones or carry out more internal attacks to the network. The entire process of launching and detection of clone attacks is depicted in a flowchart shown in Figure 2.

2.3.2. Threat Model. In WSNs, the attacker may possibly launch active and passive attacks. In this work, we consider the existence of an active attack, in which the attacker can launch a clone node attack by compromising a subclass of nodes and producing large amount of replicas for distribution all over the network. Upon compromising a node “ n ,” the attacker may produce a group of replicas $n' = n'1, n'2, n'3, \dots, n'r$ of which the IDs and secret credentials are the same as the original node n . Replicas can easily override the authenticity and integrity of existing cryptographic security mechanisms because they can sign, encrypt, and decrypt messages to execute the rule, just like original vulnerable node. Once replicas are identified as a legitimate part of the network, they can launch a variety of attacks, such as Sybil attack, selective forwarding attacks, incorrect data injection, protocol interruptions, and traffic jams.

2.3.3. Clone Detection Techniques in Static WSNs. There have been numerous techniques proposed for clone detection in static WSNs which can be categorized into centralized and distributed techniques.

(1) Centralized Clone Detection Techniques. Apart from being complex and having low overheads, these techniques mainly rely on powerful base station (BS) for information convergence and decision making, where the nodes send their position claims to the BS with the help of their neighbors. Then, the BS will check the node IDs, and if one ID is found in more than one location, an alarm is set up to

give alertness about the presence of a clone attack. These techniques are capable enough to detect clone attacks. Yet, this does not mean that private information of the sensor is secured, where the attacker can do many negative things to spy on the transmitted information between the sink and sensor node. Thus, there may still be a threat to the network. Another problem is that the lifetime of the network may decrease quickly due to the fact that the nodes which are closer to the sink node lose their energy faster. The static WSNs centralized detection techniques can be categorized into one of these six categories, i.e., base station-based, cluster head-based, key usage-based, zone-based, neighbor ID-based, and neighborhood social signature-based techniques.

(2) Distributed Clone Detection Techniques. The main difference here is that the process of clone detection is done by all the network nodes, which means that there is no central node of authority assigned to do the work. This also means that even the nodes that are located in distant positions in the network are involved in this process.

2.4. Whale Optimization Algorithm with Applications to Resource Allocation in Wireless Networks. Resource allocation plays a pivotal role in improving the performance of wireless and communication networks. However, the optimization of resource allocation is typically formulated as a mixed-integer nonlinear programming (MINLP) problem, which is nonconvex and NP-hard by nature. Usually, solving such a problem is challenging and requires specific methods due to the major shortcomings of the traditional approaches, such as exponential computation complexity of global optimization, no performance optimality guarantee of heuristic schemes, and large training time and generating a standard dataset of machine learning based approaches. Whale optimization algorithm (WOA) has recently gained the attention of the research community as an efficient method to solve a variety of optimization problems. As an alternative to the existing methods, our main goal in this article is to study the applicability of WOA to solve resource allocation problems in wireless networks. First, we present the fundamental backgrounds and the binary version of the WOA as well as introduce a penalty method to handle optimization constraints. Then, we demonstrate three examples of WOA to resource allocation in wireless networks, including power allocation for energy and spectral efficiency trade-off in wireless interference networks, power allocation for secure throughput maximization, and mobile edge computation offloading. As mentioned before, the WOA has found important applications in a multitude of disciplines; however, we are not aware of any work providing the application and applicability of the WOA in wireless and communication networks [2]. The main contributions can be summarized as follows:

- (i) We first provide a brief overview of WOA, including the mathematical model and optimization algorithm. Since the original WOA is only appropriate for continuous and unconstrained optimization

FIGURE 2: Stages of node clone/replication attack in WSNs [23].

problems, we present the binary version of the WOA (BWOA) and introduce the penalty method to deal with optimization constraints. Combination of the original WOA with BWOA and penalty method allows us to solve a wide range of optimization problems and obtain a high-quality solution.

- (ii) To illustrate the applicability of WOA, we investigate three resource allocation problems in wireless networks: secure throughput maximization, energy and spectral efficiency trade-off, and mobile edge computation offloading, which are then solved by the WOA algorithm. Simulation results are conducted to show that the WOA algorithm can converge very fast and achieve almost the same performance as in the existing algorithms.
- (iii) We outline some possible applications of WOA to unmanned aerial vehicle (UAV) trajectory optimization, interference management in ultra-dense networks (UDNs), user association and scheduling,

mode selection, and computation offloading in multicarrier NOMA-enabled MEC systems. The results verify that the WOA is a highly promising algorithm to optimize resource allocation problems in wireless networks [7].

Over the last few years, many efforts have been devoted to improving the WOA, which mainly focus on the exploitation and exploration capabilities and their balance. For example, Cui et al. [2] proposed using the arcsine function to control the trade-off between exploration and exploitation, and the Levy flight trajectory was used to improve the exploration capability of the WOA in [18]. Adopting such improved versions of the WOA to optimize resource optimization problems in wireless networks is an interesting direction for the future. To conclude, the WOA algorithm can be considered as an efficient global optimizer thanks to its well balance between exploitation and exploration. Since the original WOA algorithm is for unconstrained optimization, we need to employ efficient constraint-handling techniques so as to solve constrained problems. Cui et al. in

[2] divided constraint-handling techniques into two major categories: (1) classic methods, which are still widely used in its standalone form, and (2) recent methods, which are based on the hybrid of evolutionary ideas with the classic methods. Some of the well-known constraint-handling methods are the penalty method, equality with tolerance, feasibility rules, separation of objectives and constraints, stochastic ranking, and multiobjective approach [20]. Although the applications of WOA to wireless networks remain unexplored, three examples in the previous section really justify the effectiveness of the WOA-based algorithms. In the following sections, a number of potential applications of WOA in wireless networks are discussed. From this discussion, we expect that the WOA will be an effective tool for optimizing upcoming wireless systems, such as cell-free massive multi-input multioutput (MIMO), holographic, federated learning, intelligent reflecting surface, and terahertz communications.

2.5. Employing Data Mining on Highly Secure Private Clouds for Implementing a Security-as-a-Service Framework. Cloud computing is rapidly gaining popularity. However, like any new system, cloud computing is facing some significant challenges. The most significant challenge faced by cloud adopters is related to legal compliance, security controls, privacy, and trust. Data mining has been used for IT security applications requiring inspection and approval/denial decision making. Some key examples are antimalware, antispam, Internet security servers, intrusion detection, and intrusion prevention. Data mining in these applications help in archiving data about attack-like behaviors such that malicious sessions can be separated out of genuine end-customer sessions and appropriate decision-making rules as per security strategies could be formulated and implemented [24]. The key security objectives in data mining applications for security are

- (1) Establishing validity of an identity claimed by a session requester following a process called authentication.
- (2) Authorizing the requester with an identity to gain access to resources based on context, predicates, and attributes.
- (3) Ensuring that any attempt to modify, tamper, or delete information is detected and blocked (protection of integrity).
- (4) Ensuring that information is disclosed to authorized individuals only (protecting confidentiality).
- (5) Ensuring that all attempts to disrupt availability of services are detected and blocked (protecting availability).
- (6) Ensuring that all activities are logged such that they can be used to generate and enhance knowledge about attack-like behaviors.

In practical applications, data mining is used for gathering, organizing, and using intelligence information for countering cyberterrorism, insider threats, malicious

intrusions, malware attacks, Internet-based financial frauds, identity thefts, and attacks on critical infrastructures and government facilities. Data mining helps in organized detection and analysis of vulnerabilities and threats and building their one-to-one, one-to-many, and many-to-many ontological mappings such that accurate risk analysis can be done for implementing information security [23]. In addition, system, application, user activity, and administrative/maintenance activity logs could be stored as records in data mines for monitoring the unauthorized activities on sensitive files and predict the possibility of internal or external attacks. Data cubes stored in data mines and exported in the form of XML document-modelling files help in prediction of threats and risks using an integrated activity log analysis framework [12]. Activity logs may comprise attack signatures collected from various nodes spread across the network such that a collective log analysis could be conducted using data mines comprising records of distributed logs from across the networking infrastructure. A brief review of cloud computing security is presented under “background and context.” This section is dedicated to investigating use of data mining in implementing service-oriented security applications on cloud computing. The primary systems to secure are service-oriented architecture, virtualisation, networking components and links, databases, applications, data storage, and computing resources [22]. Attack patterns can be detected by applying association and sequencing rules on the item sets captured from inbound traffic for detecting attack signatures, length of streams (comparing stream-lengths of valid traffic and known attacks’ traffic), and anomaly behaviors. They designed an improved version of Apriori algorithm for supporting length-based decisions taking help of known flow patterns and attack signatures recorded in data mines [10]. The challenge is not only mining relevant records for predictive security analysis of cloud computing but also related to processing and analyzing mass log databases employing security analytical applications. Figure 3 presents a schematic of collection of logs, mining of logs, and analysis of logs needed on cloud computing. The association rule-based mining (Apriori algorithm) combined with MapReduce is currently a valid solution to mining mass-scale logs from SaaS databases. However, applications for analysis of the logs are open for research. Currently, the key challenges in designing log analysis applications are response times, usage functions, parallel processing of queries, dynamic updating of data mines, and securing the data mines [25].

Data mining has been an innovative and productive system in self-hosted infrastructures. However, its growth and popularity are limited by its neverending demands for significant computing and storage resources. Cloud computing offers a new lease for data mining in a variety of applications. Given the high usefulness of data mining in intrusion detection, anomaly detection, and user activity monitoring, an innovative era of cloud security using data mines is emerging. It will be more powerful and sustainable given the unlimited scalability, massive parallel processing capability, and resources elasticity on demand offered by the clouds. The literatures reveal usefulness of integrating

FIGURE 3: Processing and analysis of mass log data on cloud computing [9].

traditional data mining algorithms with Hadoop MapReduce framework, although research studies are still ongoing. This research will focus on the fundamentals of data mining for security applications and positioning them in the context of cloud computing with the help of existing studies. In addition, this research will present a novel data mining-based security application on cloud computing by designing, modelling, and simulating a security architecture employing applications for attack detection using data mining on the cloud. The results will be analyzed, and the architecture and its simulation results will be ratified by IT security experts working in India. The framework is expected to add value to the existing body of knowledge on data mining security and security applications using data mining on cloud computing.

3. Proposed Method for Improving Storage in Cloud Data Centers

Storage service providers have to provide services to such an extent to satisfy and meet different needs of users. Providers should be aware of the cost and reliability of their storage

service; they should first assess the storage cost of data centers and then minimize storage costs by maintaining the reliability in a safe manner, so that storage service providers do their best to reduce storage costs and maintain storage reliability. In order to solve this problem, in the proposed method, using the PSO algorithm, a multiobjective optimization model for data storage in cloud data centers is presented considering the cost of transmission, communication, storage, and construction of data centers and reliability.

3.1. Proposed LAPSO Method. In this section, a particle swarm optimization algorithm based on LA called LAPSO is proposed. Like the particle swarm optimization algorithm, there is a population of particles in this algorithm; each particle has an initial position and velocity. Figure 4 shows the proposed model for minimizing storage and reliability costs. The difference between the proposed and standard algorithms is that the proposed algorithm uses the LA to control the behavior of the particles. This LA has two actions, “following” and “continuing the current path.” Initially, the position and velocity of the particles, as well as the probability vector for the selection of LA operations, are initialized. Then, as long as the maximum number of steps or the desired goal is achieved, the following steps are repeated:

- (i) The LA chooses one of its actions based on the probability vector of the operations.
- (ii) According to the selected operation, the method for updating the particle velocity is determined and then the particles update their velocity and position.
- (iii) Based on the results of the particles’ positioning update, the LA operation is evaluated and the probability vector for the selection of LA operations is corrected.

The action that the LA takes in each step determines how the particle velocity is updated in that step. If the “following” operation is selected, only the best individual experience and the best group experience will be considered in updating the particle velocity, and the current velocity of the particles will be excluded, in which case the velocity of the particles will be updated according to (7). If the “continuing the current path” operation is selected, the new velocity of the particles will be equal to their current velocity and the particle will continue the current path.

$$V = c1 * rand * (pbest - position) + c2 * rand * (gbest - position). \quad (6)$$

In fact, selecting the “following” operation will result in a local search and selecting the “continuing the current path” operation will result in a global search and the discovery of unknown parts of the search space. The task of the LA is to learn to remove the desired probability and to create a balance between global search and local search in the search process. The method for evaluating the selected operation is that the current position of each particle is compared with its previous position. If C_{imp} improves from the population

FIGURE 4: Proposed model for minimizing storage and reliability costs.

position, the selected operation is positive; otherwise, negative. C_{imp} is one of the proposed algorithm parameters that should be adjusted according to the type of problem and the LA used. Figure 5 shows the pseudocode of the proposed algorithm.

3.2. Definition of Variables. Each server in the data center of a cloud storage system can appear as a node and the connection between the servers and edges [26]. So, we can have a complete graph to show server networks. Data files with different security levels have different storage costs. In addition to the cost of cryptographic processing and use of a special device, another problem is that high-security data files may have more backups around different locations. Therefore, the monitoring system needs to communicate with servers located in different locations. Flowchart of the proposed method is presented in Algorithm 3. These processes not only cost a lot but also make storage unreliable.

$$S = \{s_1, s_2, \dots, s_N\}, \quad (7)$$

where S is the set of servers in the database of a cloud storage system. These servers are located in different places and the total number of servers is N . Parameter d is the distance between the servers; for example, the distance between server i and server j is as follows:

$$d(s_i, s_j), \quad (8)$$

where $\forall i \in \{1, 2, \dots, N\}$, $d(s_i, s_j) = 0$ and $\forall i, j \in \{1, 2, \dots, N\}$, $i \neq j$, $d(s_i, s_j) \geq 0$. Data files sent by users are as follows:

$$F = (f_1, f_2, \dots, f_M), \quad (9)$$

in which f_i represents a split data file and M is the number of data files; each data file has its own security level, shown as $L(f_i)$.

$$L(f_i), \quad i = 1, 2, \dots, M. \quad (10)$$

3.2.1. Total Cost. The total cost of services is for storing huge data. On each server, data files with different security levels usually show different storage costs as $C_s(s_i, L(f_i))$, which is different from the security level of the files. The number of backups for f_i data files with different security levels is often adjusted accordingly. If a data file has a high level of security, it will cost more to store. The total service cost for storing a set of F data files is divided into four sections: storage cost, transfer cost, communication cost [15, 20], and data center construction cost.

3.2.2. Cost of Building a Data Center. The cost of creating a data center depends on the size and number of data centers as well as their location. Therefore, to minimize the cost of a data center, factors including the number of data centers, the distance between the data centers, and the location of the data center are optimized using the LAPSO algorithm. The LAPSO algorithm calculates the best location for data centers to communicate between the S and S' data centers. In this method, the cost of creating a data center is minimized according to the parameters of the number of data centers, data volume, distance between the data centers, and the best position of the data centers.

3.2.3. Storage Cost. The cost of storing f_i data files per unit of time can be treated as the cost of renting data center space for a data storage space, the product of which is the rented volume for data files. The price of the storage service is related to the target server and the storage time, so the cost of storing the f_i data file can be assessed through the following formula [15]:

$$\zeta_s(f_i) = \sum_{j=1}^N C_s(s_j, L(f_i)) X(f_i, s_j), \quad (11)$$

Therefore, the cost of storing datasets at time t is the total cost of storing each data file in the cloud storage system, the formula of which is as follows:

FIGURE 5: Flowchart of the proposed method.

$$\begin{aligned}
 \zeta_s(F, t) &= \sum_{i=1}^M \zeta_s(f_i) \cdot t \\
 &= \sum_{i=1}^M \sum_{j=1}^N C_s(s_j, L(f_i)) X(f_i, s_j) \cdot t.
 \end{aligned} \tag{12}$$

3.2.4. Transfer Cost. Users can upload their data files over the Internet to a cloud system and access it almost anywhere.

Servers that are in the cloud management layer decide which target storage servers are selected to store user data files. Therefore, some data files may be transferred to other servers located in different locations. The cost of the required transfer is essentially proportional to the distance between the transfer and data volume. Assuming that data files are loaded on the server, the cost of migrating a data block f_k from server s_b to server s_j can be calculated as follows:

$$\zeta_{\text{mig}}(f_k, s_b, s_j) = C_m X(f_k, s_b) d(s_b, s_j), \tag{13}$$

where C_m is the transfer price. Therefore, the cost of transferring data files during the storage process can be calculated as follows [15]:

$$\begin{aligned}\zeta_m(F) &= \sum_{k=1}^M \sum_{j=1}^N \zeta_{\text{mig}}(f_k, s_b, s_j) \\ &= \sum_{k=1}^M \sum_{j=1}^N C_m X(f_k, s_b) d(s_b, s_j).\end{aligned}\quad (14)$$

3.2.5. Communication Cost. Communication cost is related to the process of transferring data files to the cloud system. The storage management system must collect the storage information of the cloud system servers to ensure that certain data files can be stored on that server. Therefore, communication is established throughout the entire storage process [26]. In the management system, when the desired server is s_i . The cost of communication is proportional to the distance of server s_m . The data file expressed by the f_k is intended for storage in communication. The cost of communication from f_k data file storage is as follows:

$$\zeta_{\text{com}}(f_k) = \sum_{i=1}^N [X(f_k, s_i) \omega(s_i, s_m) d(s_i, s_m)]. \quad (15)$$

Therefore, the total cost of communicating a set of data files provided for storage is calculated as follows [15]:

$$\zeta_{\text{com}}(f_k) = \sum_{i=1}^N [X(f_k, s_i) \omega(s_i, s_m) d(s_i, s_m)]. \quad (16)$$

3.2.6. Total Cost. The F dataset must be sent via the sb terminal server. $\zeta(F, S)$ is the total cost of storing dataset in the cloud storage system, which is the sum of the storage cost, communication cost, and cost of transferring and creating the data center.

4. Result and Experiments

In this section, we review and compare the proposed LAPSO method with other storage improvement methods, implementing the program using MATLAB software on a system with a 2.4 GH Core i3 processor, 4 GB of RAM, and Windows 8 operating system. The proposed method is implemented and evaluated with other methods in equal conditions for a dataset that includes 13 data centers. Below is a description of Iran's spatial and geographical information dataset and initial parameters for the LAPSO algorithm; then, the results are examined.

4.1. Data Collection. In the defined dataset, which has 13 data centers, belongs to Iran's Spatial and Geographical Information, and is taken from the country's Land Affairs Organization, 13 cities out of 31 provinces are located that can be optimal in terms of distance and cost of data transfer. Data centers are selected to store the location data of each

province in the existing or near data center of that province and as a backup in another data center that is optimal in terms of data center distance and cost of data transfer. The parameters of this dataset are shown in Table 2. In Table 2, the dataset parameters include the number of data centers (n), the set of data centers (S, S'), the characteristics of data centers (q ID, p name), the position of data centers (x, y), and the edge for data center communication (E). In Table 3, the characteristics of the data centers, which include the name, ID, and location of the data center, are displayed. In Table 2, longitude and latitude (x, y) of data centers related to spatial information of Iran are displayed with codes DC1 to DC13.

4.2. PSO Parameters. The parameters of the PSO algorithm for improving storage in data centers include the number of particles (m), the objective function (f), the position and velocity of the particles (X, V), and other parameters as follows:

- (i) Population number $n = 100$; number of particles $m = 78$.
- (ii) Acceleration coefficient: value of c_1, c_2 is 2 by default and $\omega = 0.9 - t/2\text{MAXgen}$, where t is the current generation and MAXgen is the maximum generation.
- (iii) Selection strategy: the next population is selected using the elitism strategy according to the unexpected fast classification approach.
- (iv) Pausing the criterion: the algorithm will stop when the evolution generation reaches n or the solutions found by the algorithm converge in 10 generations.

4.3. Cloud Storage Optimization Tests. In this section, for cloud storage optimization tests, we consider two optimizations including cloud storage in terms of distance and cloud storage for distance and cost factor, which are described below.

4.3.1. Optimization of Cloud Storage in terms of Distance. The main task of this section is to analyze and discuss the problem of cloud storage among 13 data centers according to the distance factor. The entire distance between adjacent data centers plays an important role in the transmission and efficiency of data centers. The proposed method (LAPSO) is compared with the PSO and genetic algorithms [14]. The parameters in the LAPSO algorithm and the cloud storage optimization problem should start at the beginning of the evolutionary process. The number of all particles in the LAPSO algorithm is set to 78, and the maximum number of generations is set to 100. Data center coordinates are mainly determined by latitude and longitude. There are 13 data centers, and the topology is shown among the data centers listed in Figure 6. Each particle in the LAPSO algorithm is a relationship between the data center of the S set and another data center of the S' set. The goal of the LAPSO algorithm is to find the smallest distance between the S and S' data centers for the topology of the defined data centers. So, if

TABLE 2: Parameters of Iranian spatial and geographical information dataset.

Value	Variable	Parameter name
Number of data centers	N	13
Collection knot	S, S'	Si is the existing data center and Sj is the associated data center
Set edge	E	Relationship between two data centers (1 and 0)
Characteristics of data centers	q ID, p name	Location of data centers with coordinates x, y (Table 3)

TABLE 3: Parameters of Iranian spatial and geographical information dataset.

ID(q)	Name (p)	Location UTM (x, y)	
		X	Y
DC1	Tehran center	538626.46216459	3951535.2490513
DC2	Tehran	536943.9872355	3951505.9101427
DC3	Alborz	497922.53361457	3964705.2853616
DC4	Qazvin	410482.75511827	4015920.8657207
DC5	Bushehr	483400.08400049	3205947.9365054
DC6	Fars	648527.49659749	3278582.3692991
DC7	Khorasan Razavi	730531.26932172	4022880.6637983
DC8	Esfahan	562144.37856225	3607647.7735902
DC9	Lorestan	255113.93320934	3709585.6133087
DC10	Sistan and Baluchestan	290995.76871514	3263590.6754382
DC11	Kerman	504876.22112375	3349569.562802
DC12	Tabriz	610369.82049148	4213874.051253
DC13	Gilan	375359.84915775	4125296.0895937

each data center (n) is considered a node on the graph, the number of PSO particles (m) is equal to the total number of edges of the complete graph ($m = n(n - 1)/2$). Table 4 shows the results of the proposed method and the genetic algorithm. Figure 7 shows the geographic coordinates and relationships between the data centers optimized with the PSO algorithm. In addition, the minimum fit of the objective function in 20 executions corresponding to the LAPSO algorithm is 32.4322; for the discrete PSO algorithm, it is 40.7855 as discussed in [20], and for the genetic algorithm, it is equal to 45.0838 as discussed in [14]. The proposed method has better results in the parameters of minimum distance and cost, which includes the cost of data transfer, data communication, and cost of creating data centers, compared to other methods.

4.3.2. Optimizing Cloud Storage for Distance and Cost Factor.

Depending on the distance and cost factor in the cloud storage issue, the corresponding objective function is selected. With the LAPSO algorithm, the mapping is optimized between the S node set and the S' node set. The best fit for the objective function among the 20 executions is 37,6634 for LAPSO, 48,5824 for PSO, and 52,3044 for the genetic algorithm. The issue of cloud storage among distributed data centers is identified by the LAPSO algorithm and compared with other articles. The result can be applied to the problem of cloud storage in the 13 distributed data centers, and they also provide guidance for the 8 central points of the data center, leading to low production costs. In Table 5, the parameters include optimal location, minimum route cost, distance, data transfer cost, storage cost, data connection cost, load

balance, access, and year, which are evaluated and compared with other methods.

5. General Explanation about Proposed Method

This simulation, which is implemented in MATLAB programming language, uses the capabilities of the SSO algorithm or similar particle swarm optimization (simplified swarm optimization) to find the nearest and most optimal neighbors, so that it can send and receive information with energy consumption highly optimized in cloud computing by virtual machines. Given that the SSO algorithm is an improved version of the PSO algorithm, this section provides a complete description of the PSO and SSO algorithms. The PSO algorithm is a social search algorithm modelled on the social behavior of bird flocks. Initially, this algorithm is used to discover the patterns governing the simultaneous flight of birds and their sudden change of direction and optimal deformation of the handle. In PSO, particles flow in the search space. Displacing the particles in the search space is influenced by the experience and knowledge of themselves and their neighbors. Therefore, the other position of the particle mass affects how a particle is searched. The result of modelling this social behavior is the search process, in which particles tend to go to successful areas. Particles learn from each other and move towards their best neighbors based on the knowledge gained and determine the location of all neighbors. According to the contents presented in this section, a new algorithm can be presented, which is an improved PSO algorithm. This algorithm is called "similar particle swarm or SSO algorithm." SSO improves the search space of the PSO algorithm and selects the best neighbor, or in fact, the best particle that has outstanding properties

FIGURE 6: UTM coordinates of data centers in x , y -axes (latitude and longitude).

TABLE 4: Comparison of the proposed method with other methods in improving cloud storage.

Method	Genetic algorithm	PSO	Proposed method
The cost of creating a data center	45.0838	40.7855	32.4322
Data transfer cost	52.3044	48.5824	37.6634

FIGURE 7: Optimal cloud storage strategy among distributed data centers.

compared to other particles from the particles in the search space. From this perspective, the SSO algorithm can be used to select the best virtual machine available in cloud computing data centers in order to exchange data and thus reduce energy consumption. The SSO work is based on the principle that at any given moment, each particle adjusts its location in the search space according to the best location it has ever been in and the best location in its entire neighborhood. Figure 8 shows the improvement in selecting the

best and most optimal ninety neighbors for transmitting information.

Therefore, by simulating the SSO algorithm in cloud computing, we can significantly improve the selection of the best and most optimal ninety neighbors to transmit information and exchange messages in reducing the energy consumption of cloud data centers. Different parameters are taken here and their comparison is made as shown below. The value of the makespan of SSO is compared with the

TABLE 5: Comparison of the proposed method with other methods.

Strategy	Availability	Load balancing	Data connected cost	Storage cost	Data transfer cost	Distance	Least cost path	Optimal location
DCR2S [27]	✓	✓	✓	✓	✓			
SLDP [28]	✓	✓						
CDRM [29]	✓	✓	✓	✓				
BARRS [30]	✓	✓						
MORM [31]	✓	✓		✓				
RAPGC [32]	✓	✓		✓				
Fuzzy-FP [33]	✓	✓						
PDDRA [23]	✓	✓						
EIMORM [34]	✓	✓						
DPRS [35]	✓	✓						
PDR [36]	✓	✓		✓				
ACO [24]	✓	✓						
GASA [37]	✓	✓						
DRSACO [25]	✓	✓						
Genetic algorithm [2]	✓	✓						
EFS [38]	✓	✓						
LAPSO (proposed)	✓	✓	✓	✓	✓	✓	✓	✓

FIGURE 8: Improving the selection of the best and most optimal ninety neighbors to transmitting information.

FIGURE 9: PSO and SSO makespan time for 10 data centers.

existing PSO for 10 data centers as shown in Figure 9. The number of cloudlets that show the value of makespan is calculated for 10 data centers with each 50, 100, 150, and 200 cloudlets for both SSO and PSO approaches.

The existing PSO is compared with the proposed SWO. It is observed that the proposed technique gives an optimized makespan, rather than the existing PSO. It is observed that makespan is nearly 73.23% more optimum in SSO than the existing PSO for 50 cloudlets, 88.64% for 100 cloudlets, 92.18% for 150 cloudlets, and 94.04% for 200 cloudlets.

6. Conclusions

Robustness of cloud storage in data centers was analyzed using the proposed method. In data centers, spatial information of Iranian lands was introduced according to the existing communication network and the number of available resources to solve the storage problem between LAPSO data centers. The method used to optimize cloud storage was to minimize storage costs. Optimal location parameters, route minimum cost, distance, data transfer cost, storage cost, data connection cost, load balance, access, and year were employed for optimization. But in other methods, only a few parameters have been evaluated. On the other hand, in this method, implementing the LA algorithm increased accuracy and improved the results compared to other methods. In this paper, a simplified version of PSO algorithm was proposed to solve the job scheduling problem in cloud computing environment. To evaluate the performance of the proposed approach, this study compared the proposed SSO strategy by having it implemented on the CloudSim toolkit. The results obtained demonstrated that the presented PSO algorithm can significantly reduce the makespan of the job scheduling problem compared with the other metaheuristic algorithms evaluated in this paper. Average makespan was compared between the existing PSO and proposed SSO. The average difference thus found out was nearly constant for all the values. The result was validated as the makespan of the proposed SSO was less than that of the existing PSO. The fitness function was also improved.

Data Availability

The data used to support the findings of this study are currently under embargo while the research findings are commercialized. Requests for data, 6/12 months after publication of this article, will be considered by the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

- [1] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of IEEE International Conference on Neural Networks*, pp. 1942–1948, Piscataway, Perth, Australia, November 1995.
- [2] L. Cui, J. Zhang, L. Yue, Y. Shi, H. Li, and D. Yuan, "A genetic algorithm based data replica placement strategy for scientific applications in clouds," *IEEE Trans. on Services Computing*, vol. 11, no. 4, pp. 727–739, 2015.
- [3] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in Engineering Software*, vol. 95, pp. 51–67, 2016.
- [4] H. Faris, I. Aljarah, M. A. A. Betar et al., "Grey wolf optimizer: a review of recent variants and applications," *Neural Computing & Applications*, vol. 30, pp. 413–435, 2018.
- [5] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, no. 3, pp. 46–61, 2014.
- [6] H. Hu, Y. Bai, and T. Xu, "A whale optimization algorithm with inertia weight," *WSEAS Transactions on Computers*, vol. 15, pp. 319–326, 2016.
- [7] D. Prasad, A. Mukherjee, G. Shankar et al., "Application of chaotic whale optimisation algorithm for transient stability constrained optimal power flow," *IET Science, Measurement & Technology*, vol. 11, no. 8, pp. 1002–1013, 2017.
- [8] W. Z. Sun and J. S. Wang, "Elman neural network soft-sensor model of conversion velocity in polymerization process optimized by chaos whale optimization algorithm," *IEEE Access*, vol. 5, pp. 13062–13076, 2017.
- [9] Y. Ling, Y. Zhou, and Q. Luo, "Lévy flight trajectory-based whale optimization algorithm for global optimization," *IEEE Access*, vol. 5, pp. 6168–6186, 2017.
- [10] P. H. Venkatrao and S. S. Damodar, "HWFusion: holoentropy and SP-whale optimisation-based fusion model for magnetic resonance imaging multimodal image fusion," *IET Image Processing*, vol. 12, no. 4, pp. 572–581, 2018.
- [11] E. Christian, F. Massimo, P. Francesco, and C. Aniello, "Smart cloud storage service selection based on fuzzy logic, theory of evidence and game theory," *IEEE Transactions on Computers*, vol. 65, no. 8, pp. 2348–2362, 2017.
- [12] J. Riget and J. S. Vesterstroem, *A Diversity Guided Particle Swarm Optimizer-The ARPSO*, Computer Science, 2002.
- [13] T. Kiink, J. S. Vesterstroem, and J. Riget, "Particle swarm optimization with spatial particle extension," in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1474–1479, Honolulu, HI, USA, May 2002.
- [14] X. H. Wang and J. J. Li, "Hybrid particle swarm optimization with simulated annealing," in *Proceedings of the Third International Conference on Machine Learning and Cybernetics*, Shanghai, China, August 2004.
- [15] B. Afzal, S. Shahaboddin, G. Anwar, and C. Anthony, "Optimizing iaasprovider revenue through customer satisfaction

- and efficient resource provisioning in cloud computing,” *IET Communications*, vol. 13, no. 9, pp. 2913–2922, 2019.
- [16] M. R. Meybodi and H. Beigy, “A note on learning automata based schemes for adaptation of BP parameters,” *Journal of Neurocomputing*, vol. 48, no. 4, pp. 957–974, Oct. 2002.
 - [17] H. Beigy and M. R. Meybodi, “A learning automata based algorithm for determination of minimum number of hidden units for three layers neural networks,” *International Journal of Systems Science*, vol. 40, no. 1, pp. 101–118, 2009.
 - [18] M. Munetomi, Y. Takai, and Y. Sato, “SGA: an application of genetic algorithm to stochastic learning automata,” *Systems and Computers in Japan*, vol. 27, pp. 68–78, 1996.
 - [19] M. N. Howell, T. J. Gordon, and F. V. Brandao, “Genetic learning automata for function optimization,” *IEEE Trans. on Systems, Man, and Cybernetics-Part B: Cybernetics*, vol. 32, no. 6, 2002.
 - [20] K. P. Chan and J. B. Seung, “Blockchain of finite-lifetime blocks with applications to edge-based iot,” *IEEE Internet of Things Journal*, vol. 7, no. 9, 2020.
 - [21] J. Kennedy and R. C. Eberhart, “A Discrete Binary Version of the Particle Swarm Algorithm,” in *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, pp. 4104–4108, Computational Cybernetics and Simulation, Orlando, FL, USA, October 1997.
 - [22] K. S. Narendra and M. A. L. Thathachar, *Learning Automata: An Introduction*, Prentice-Hall, Hoboken, NJ, USA, 1989.
 - [23] N. Saadat and A. M. Rahmani, “PDDRA: a new prefetching based dynamic data replication algorithm in data grids,” *Future Generation Computer Systems*, vol. 28, no. 4, pp. 666–681, 2012.
 - [24] N. J. Navimipour and B. A. Milani, “Replica Selection in the Cloud Environments Using an Ant colony algorithm,” in *Proceedings of the Third International Conference on Digital Information Processing, Data Mining, and Wireless Communications*, pp. 105–110, DIPDMWC), Moscow, Russia, July 2016.
 - [25] L. Wang, J. Luo, J. Shen, and F. Dong, “Cost and Time Aware Ant colony Algorithm for Data Replica in Alpha Magnetic Spectrometer experiment,” in *Proceedings of the IEEE International Congress on Big Data*, pp. 247–254, Santa Clara, USA, June 2013.
 - [26] R. Kohli, A. Garg, S. Phutela, Y. Kumar, and S. Jain, *An improvised model for securing cloud-based E-healthcare systems IoT in Healthcare and Ambient Assisted Living. Studies in Computational Intelligence*, G. Marques, A. K. Bhoi, and V. H. C. Albuquerque, Eds., Springer, Singapore, 2021.
 - [27] N. K. Gill and S. Singh, “A dynamic, cost-aware, optimized data replication strategy for heterogeneous Cloud data-centers,” *Future Generation Computer Systems*, vol. 65, pp. 10–32, 2016.
 - [28] R. Xiong, J. Luo, and F. Dong, “Optimizing data placement in heterogeneous Hadoop clusters,” *Cluster Computing*, vol. 18, no. 4, pp. 1465–1480, 2015.
 - [29] Q. Wei, B. Veeravalli, B. Gong, L. Zeng, and D. Feng, “CDRM: A Cost-Effective Dynamic Replication Management Scheme for Cloud Storage cluster,” in *Proceedings of the IEEE International Conference on Cluster Computing*, pp. 188–196, Heraklion, Greece, September 2010.
 - [30] I. Casas, J. Taheri, R. Ranjan, L. Wang, and A. Y. Zomaya, “A balanced scheduler with data reuse and replication for scientific workflows in Cloud computing systems,” *Future Generation Computer Systems*, vol. 74, pp. 168–178, 2017.
 - [31] S. Q. Long, Y. L. Zhao, and W. Chen, “MORM: a multi-objective optimized replication management strategy for cloud storage cluster,” *Journal of Systems Architecture*, vol. 60, no. 2, pp. 234–244, 2014.
 - [32] B. Spinnewyn, R. Mennes, J. F. Botero, and S. Latre, “Resilient application placement for geo-distributed Cloud networks,” *Journal of Network and Computer Applications*, vol. 85, pp. 14–31, 2017.
 - [33] P. Elango and K. Kuppasamy, “Fuzzy FP-tree based data replication management system in cloud,” *International Journal of Engineering Trends and Technology*, vol. 36, no. 9, pp. 481–489, 2016.
 - [34] E. B. Edwin, P. Umamaheswari, and M. R. Thanka, “An efficient and improved multi-objective optimized replication management with dynamic and cost aware strategies in Cloud computing data center,” *Cluster Computing*, vol. 22, Article ID 11119, 2017.
 - [35] L. Zhang, Y. Deng, W. Zhu, J. Zhou, and F. Wang, “Skewly replicating hot data to construct a power-efficient storage cluster,” *Journal of Network and Computer Applications*, vol. 50, pp. 168–179, 2015.
 - [36] N. Mansouri and M. M. Javidi, “A new Prefetching-aware Data Replication to decrease access latency in Cloud environment,” *Journal of Systems and Software*, vol. 144, pp. 197–215, 2018.
 - [37] T. Junfeng and L. Weiping, “Pheromone-based genetic algorithm adaptive selection algorithm in cloud storage,” *International Journal of Grid and Distributed Computing*, vol. 9, no. 6, pp. 269–278, 2016.
 - [38] M. Bsoul, A. Al-Khasawneh, E. Abdallah, and Y. Kilani, “Enhanced fast spread replication strategy for data grid,” *Journal of Network and Computer Applications*, vol. 34, pp. 575–580, 2011.