Hindawi
Complexity
Volume 2021, Article ID 9527430, 12 pages
https://doi.org/10.1155/2021/9527430

WILEY | Hindawi

Research Article

Use Linear Weighted Genetic Algorithm to Optimize the Scheduling of Fog Computing Resources

Ruisheng Li

College of Public Security Technology, Gansu University of Political Science and Law, Lanzhou, Gansu 730070, China

Correspondence should be addressed to Ruisheng Li; lrs6201@gsli.edu.cn

Received 12 April 2021; Revised 17 May 2021; Accepted 29 May 2021; Published 9 June 2021

Academic Editor: Zhihan Lv

This paper establishes a mathematical model for the resource management and scheduling of the fog node cluster and establishes the optimization goals of delay, communication load, and service cost. According to the idea of genetic algorithm for single-objective optimization, this paper proposes a linear weighted genetic algorithm based on linear weighting. The optimization weight is established according to the user's preference for the target. We normalize the optimization objective function and merge it into one target, and then we proceed with genetic manipulation to get a better solution. The experimental results show that when the user specifies the preference weight, the optimal solution can be obtained by the genetic algorithm based on linear weighting, and the algorithm execution efficiency is high. With the increase of the single-objective weight, the optimization effect of this objective is better. When the preference weight tends to be average, its overall optimization effect is not ideal. When the user does not specify the preference weight, a set of optimal solutions can be obtained through the improved nondominated sorting genetic algorithm with elite strategy. Compared with the traditional algorithm, in addition to the overall optimization effect of the target being better, the algorithm itself also has higher efficiency.

1. Introduction

With the rapid development of the Internet of Things technology, hundreds of millions of smart devices are interconnected with each other and exchange data and information through the Internet [1]. Intelligent services for the Internet of Things have been widely used in many areas of people's daily lives. As a new type of centralized computing model, cloud computing uses virtualization technology as the core technology. It has the characteristics of large scale, virtualization, safety, and reliability [2]. It can optimize the storage and processing of massive data generated by the Internet of Things and provide end users with high efficiency service. However, with the rapid increase of mobile devices and sensing devices, the number of user visits to cloud data centers has increased, consuming a large amount of network bandwidth, causing a huge burden on cloud data centers. In addition, the mobility, location awareness, and other functions required by a large number of IoT applications at the edge of the network cannot be met [3]. In this context, Cisco has proposed a new service computing model called fog computing, which extends the traditional cloud computing paradigm to the edge of the network [4].

In the era of big data processing, data types are diversified, and real-time data processing requirements are high. Cloud computing data centers are far away from users and cannot meet real-time requirements. Fog computing is close to the edge of the network to ensure real-time data processing. Moreover, cloud computing has shortcomings such as too complex technology and high cost, which cannot meet the current complex market needs, while fog computing can make full use of some of the advantages of cloud computing, such as scalability and technical transparency, to meet the increasing complexity of users [5–7]. In addition, the functions of cloud and fog are complementary and mutually reinforcing, which can meet the needs of low-latency IoT applications at the edge of the network, and support long-term data storage and complex analysis [8]. All in all, as a new generation of Internet of Things technology, fog

computing has improved the ecological environment of the Internet of Things, opened up new business models, and has broad market development prospects, which will have a profound impact on individuals, enterprises, and society [9, 10]. Therefore, this article is of great significance to the development of fog computing.

This paper analyzes the limitation of easily falling into the local optimal solution, expounds common improved hybrid strategies such as virus linear weighted genetic algorithm, DNA mutation genetic algorithm, and immune genetic algorithm, and summarizes the improved methods of genetic algorithm hybrid. Specifically, the technical contributions of this article can be summarized as follows:

First, based on the hill climbing algorithm and the adaptive sharing strategy, the improvement strategy of the standard genetic algorithm is discussed, and the improvement method is described in detail.

Second, we transform the multiobjective optimization problem into single-objective optimization, set preference weights for each objective, and simplify the constraint relationship between each objective. A genetic algorithm based on linear weighting is used to obtain a resource scheduling scheme that satisfies the user's service preference requirements.

Third, we perform experimental simulation. The results show that the linear weighted genetic algorithm has a better optimization effect than the multi-GA and the RAS-IN algorithm. We compare the performance of the improved algorithm in this paper with the traditional algorithm. It can be seen that the algorithm proposed in this paper not only improves the search efficiency, but also ensures the uniformity of the final optimal solution set distribution.

2. Related Work

At present, domestic research on fog computing mainly focuses on definition, system architecture, and application scenarios. Related scholars introduced the definition, characteristics, and basic architecture of fog computing in detail, then analyzed the application fields of fog computing and related security issues, briefly discussed several technologies similar to fog computing technology, and finally pointed out fog computing and cloud computing [11]. Related scholars described the background of fog computing in detail, highlighted the advantages of fog computing through comparison with other similar computing models, and analyzed the functions of each layer of the architecture in detail [12]. At the same time, the article discussed two aspects of network management and resource scheduling. Researchers have also conducted research on related applications of fog computing in the Internet of Things [13]. Based on fog computing and reinforcement learning theory, related scholars proposed a Fog Reinforcement Traffic Light (FRTL) control model [14]. In this model, the fog node uploads the collected traffic information to the fog server for processing, thereby formulating traffic lights. Simulation experiments show that this control method can reasonably adjust and control the time of traffic lights and effectively alleviate traffic congestion. Researchers proposed a new fog computing framework based on the idea of intelligent front-end, used this framework to develop a hospital service system, and finally verified the usability of the system through test experiments [15]. In addition, some scholars have conducted related research on privacy protection in fog computing. Related scholars have studied the authentication and privacy issues in fog computing, summarized the latest privacy and authentication results, and pointed out possible future research directions [16].

Aiming at the problem of realizing the reasonable allocation and scheduling of mobile user task requests under cloud and fog collaboration, a task allocation algorithm based on the cloud and fog collaboration model is proposed [17]. This algorithm improves the simple genetic algorithm, takes the cost of the service provider as the objective function, selects the optimal solution through basic genetic operations, and achieves obvious optimization in delay and cost. Abbasi et al. introduced an improved genetic algorithm, which introduces fitness judgment into the parental mutation operation and overcomes the blindness of the basic genetic algorithm in the mutation operation [18]. When using this algorithm to schedule tasks under the fog computing architecture, the response time constraints in the service level goals are considered. Both of these two articles use single-objective genetic algorithms and do not consider various resource requirements in resource scheduling. Hussein et al. first formulated the task scheduling problem in the cloud and fog environment and then proposed a non-heuristic task scheduling algorithm, which can obtain better comprehensive benefits between the maximum execution time and resource cost consumption [19]. They studied the perceptual service allocation problem of the cloud-fog joint architecture as an integer optimization problem, and their solution can minimize service delay while meeting capacity requirements.

Matrouk et al. have studied the trade-off between power consumption and transmission delay in cloud computing systems, modeled the load distribution problem, and proposed the best workload distribution scheme between fog and cloud, and the assignment of tasks is constrained by service delay. The authors resolve the problem by decomposing the original problem into three subproblems and using approximate methods. Based on the results of simulation and numerical calculation, the article saves communication bandwidth and reduces transmission delay by sacrificing appropriate computing resources, which greatly improves the comprehensive performance of cloud computing. The researchers described the resource allocation problem as a bilateral matching optimization problem. By analyzing the utility and cost of the fog computing network, they proposed a double matching strategy based on the cost efficiency of the fog computing network resource allocation problem. The proposed double matching strategy is to delay the acceptance algorithm. Numerical results show that using this strategy can achieve higher cost-effective performance. Related scholars introduced a fog cloud allocation mechanism based on Gaussian process regression, which is used for

resource allocation of infrastructure composed of collaborative fog and cloud [20]. The fog cloud allocation mechanism uses Gaussian process regression to predict future demand to avoid blocking requests, especially delay-sensitive requests. The article proposes a fog computing resource allocation strategy based on the pricing time nonferrous network. Users can autonomously select satisfactory resources from a set of preallocated resources. In addition, an algorithm for predicting task completion time and a dynamic allocation algorithm for fog resources are proposed to achieve higher efficiency in both task completion time and cost.

Some foreign scholars have carried out research on the basic definition, framework, and related applications of fog computing. Related scholars expounded the concept of fog computing paradigm, defined some characteristics of the paradigm, and explained some of its applications in real life [21]. The authors also emphasized the importance of fog cloud interaction and the role of fog computing in the context of the Internet of Things. The researchers defined the various components of fog computing, described the theoretical model of fog computing from a mathematical perspective, and compared it with traditional cloud computing models from the perspective of service delay and energy consumption. Related scholars put forward the integration problem of cloud and Internet of Things and made a simple comparison between fog computing and cloud computing, evaluated the performance of fog computing according to different performance metrics, and finally pointed out the future research direction of fog computing. Related scholars described fog computing technology in detail and, based on the research and challenges faced by the institute, proposed a software architecture that can flexibly combine different design options and user-specified strategies [22]. In addition, academia has also conducted a series of researches on resource scheduling and allocation, resource management, and load transfer in fog computing.

In recent years, with the development of heuristic algorithms, modern heuristic algorithms such as particle swarm algorithm, ant colony algorithm, simulated annealing algorithm, neural network algorithm, standard GA, hill climbing algorithm, and tabu search algorithm have been gradually applied to the solution of VRPTW problems. Due to the global optimization and versatility of modern heuristic algorithms, the application of modern heuristic algorithms to solve the NP-Hard problems of VRP and VRPTW has gradually become the main direction of current research. Zhang et al. apply the tabu search algorithm to the VRP problem, design a related logistics center, convert the VRP problem into a TSP (Traveling Salesman Problem), and use the 3-opt and 2-opt methods to find a reasonable driving scheduling plan [23]. Wang et al. have extended the path optimization problem with time window by using genetic algorithm through the limitation of soft driving time and soft time window [24]. Lu et al. have proposed a hybrid genetic algorithm based on genetic algorithm and tabu search algorithm, which uses heuristic crossover operation and adaptive mutation mechanism [25]. The simulation results are better than the simulation results of each single algorithm.

3. Fog Computing Architecture

3.1. Overall Architecture. The fog computing architecture proposed by Cisco is a general architecture for all kinds of Internet of Things [26, 27]. It does not describe in detail the architecture. In response to the needs of intelligent manufacturing service applications, this section has been expanded and improved based on the general architecture proposed by Cisco. The fog unit node, fog management node, fog group, and other network element entities are introduced, and the IoT layer equipment and fog are analyzed in detail. The fog computing architecture is shown in Figure 1, which can be divided into three layers: a comprehensive perception layer, a fog computing layer, and a cloud computing layer. The lowest layer is the comprehensive perception layer, which connects most of the smart terminal devices in the industrial Internet of Things. The top layer is the cloud computing layer, located in the data center network. In the middle is the fog computing layer, which connects the comprehensive perception layer and the cloud computing layer and is located in the edge network.

3.1.1. Comprehensive Perception Layer. These devices use short-distance communication methods such as WIFI technology, Bluetooth technology, ZigBee technology, and mobile ad hoc network technology and usually access the fog gateway in the fog computing layer through a single-hop low-latency wireless network. This layer will continuously generate a large number of data streams, and the operation of these data needs to consider the delay sensitivity and the communication overhead brought by the data transmission [28]. Technically, it is necessary to consider the integration of communication interfaces, network technology, computing, and data storage units.

In the comprehensive perception layer, sensor nodes, wireless sensor network devices, and IoT devices are used to collect geographically distributed industrial data and transmit the data to the fog computing layer. The intelligent terminal device is used to send task requests to the fog computing layer and receive the result information of the fog computing layer executing the task requests. The controlled terminal is used to receive control instructions and information returned by the fog computing layer. This layer realizes the deep integration of physical objects and information entities.

3.1.2. Fog Computing Layer. The fog computing layer is composed of interconnected edge fog network devices with certain computing and storage capabilities. Among them, fog network devices include traditional hardware devices such as switches and routers, as well as virtual network functions that use network function virtualization technology to run on general-purpose servers. The fog gateway is the gateway of the fog computing layer. The comprehensive perception layer device communicates with the fog gateway

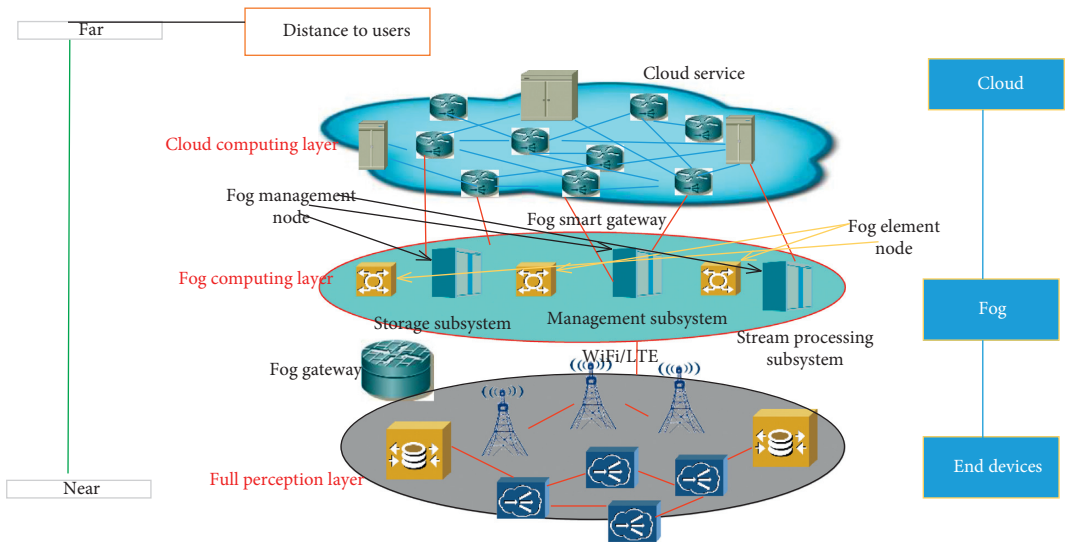


Figure 1: Fog computing architecture diagram.

through a single-hop low-latency wireless link. The fog computing layer is divided into multiple fog groups in units of fog groups. The fog group refers to a group of interconnected nodes of a part of fog units. Each fog group has a fog management node that can perceive the overall situation of the fog group [29]. The fog unit node communicates with the cloud data center of the external network through the fog management node.

3.2. Fog Resource Communication. The comprehensive perception layer includes the Message Queue Telemetry Transport (MQTT) client, that is, the publisher/subscriber. Its function is to publish topic messages to other related clients, subscribe to request to receive related topic messages, and unsubscribe to remove related topic messages. The comprehensive perception layer publishes data upwards and subscribes to controlled instructions and related information through the MQTT agent of the fog computing layer. MQTT is a lightweight communication protocol based on TCP with less protocol overhead. In terms of quality of service (QoS), in order to ensure the reliability of messages, it supports three levels of message delivery: at most once (QoS level 0), at least once (QoS level 1), and only once (QoS level 2). The three types of intelligent manufacturing services in this architecture (first-level services, fog-level services, and cloud-level services) will, respectively, use these three methods for message transmission. The information interaction process of the three types of services is shown in Figure 2.

The fog-level service uses the QoS level 1 method for message delivery. This level of service quality ensures that the message is delivered at least once, and the corresponding overhead is relatively moderate. Fog-level services have the characteristics of moderate message volume and a certain degree of reliable delivery of messages, which makes QoS level 1 meet the needs of fog-level services. The correct transmission of a message requires two interactions. For the first time, an unused message identifier is assigned to the newly sent message, and the sender sends a PUBLISH message containing the message identifier to the receiver. For the second time, the receiver sends a PUBACK message to the sender, and the PUBACK message indicates the release confirmation. At this time, the correctness of the message sent at one time is ensured, and the receiver can distribute the message to subscribers, but it is possible that the message will be sent repeatedly. When the sender receives the PUBACK message, the message identifier can be reused.

4. Linear Weighted Genetic Algorithm Construction

4.1. Genetic Algorithm. The basic idea of genetic algorithm comes from the long evolutionary process of biology from simple and low-level to complex and high-level, and it draws on natural laws such as survival of the fittest and natural selection of the fittest. The algorithm is essentially a global search method for the problem to be solved in the field of combinatorial optimization. It needs not only to consider the current basic information in the search process, but also to consider the knowledge space formed by the accumulation of past experience, and the adaptive mutation mechanism is used to control the search results to gradually approach the global optimal solution.

4.1.1. Coding Design. The coding of genetic algorithm refers to the mapping relationship between the solution space of the combinatorial optimization problem and the code space of the algorithm. During the execution of the algorithm, the calculation, evaluation, and selection of the fitness function value need to be performed in the solution space of the problem, while the chromosome genetic operation is performed in the code space of the algorithm. From this point of view, the algorithm does not directly act on the problem-related parameters during the execution process, but directly acts on the code space that the algorithm runs when it is

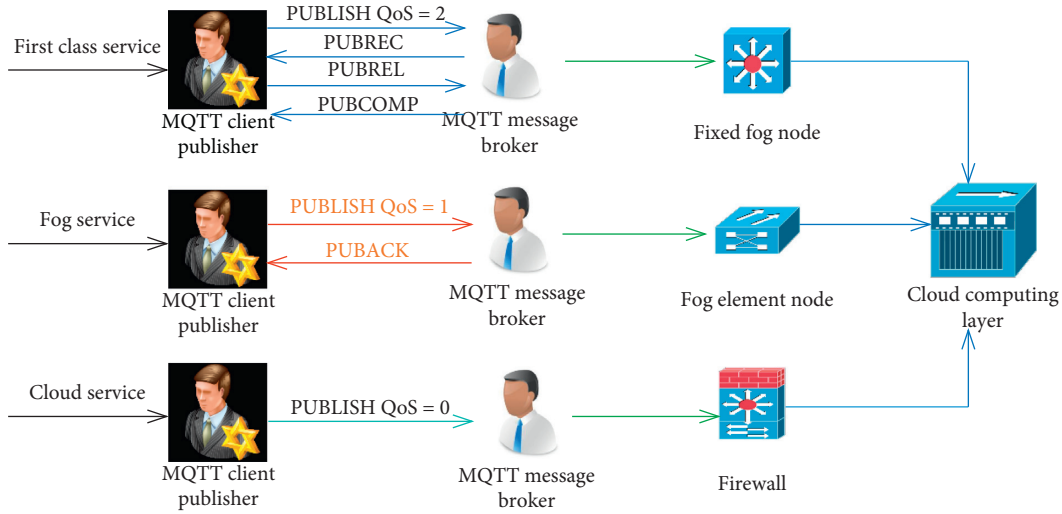


Figure 2: Information interaction process of the three types of services.

executed. Therefore, choosing a scientific and reasonable coding scheme will help improve the overall operating efficiency of the algorithm during execution. The main coding techniques of the basic genetic algorithm are gray coding, real number coding, decimal coding, and binary coding.

4.1.2. Generate the Initial Population. Before performing basic genetic operations, it is necessary to construct an initial population composed of several initial feasible solutions. The most important indicator of the initial population is the population size n. The larger the population size n, the more types of individuals for selection, crossover, and mutation, which helps to avoid falling into the local optimal solution, thereby increasing the probability of searching for the global optimal solution. However, if the population size n is too large, the calculation amount and calculation time of the calculation, evaluation, and selection of the fitness function value will increase, which will seriously affect the efficiency of the algorithm. Therefore, the population size n should not be too large or too small. In practical applications, it is generally selected between 100 and 500.

4.1.3. Fitness Function. The fitness function is the main step in evaluating the solution of the problem during the execution of the algorithm, and its design should generally be combined with the actual problem. Usually, the design is based on the cost function, objective function, etc. Among them, the maximum fitness and the average fitness are two important indexes describing the convergence of the function.

4.1.4. Selection Operator. The selection operator selects excellent individuals from the current population to form a subpopulation according to the size of the fitness function value for subsequent genetic operations. The greater the fitness function value, the greater the probability of being selected as a good individual. It is an important manifestation of the survival of the fittest in nature, and it is also an important difference between traditional search algorithms and genetic algorithms.

4.1.5. Mutation Operator. The mutation operation refers to the process in which certain genes on the coding string of the chromosome are changed according to certain rules to form a new individual. This operation directly determines the level of the global search capability of the genetic algorithm, reduces the probability of premature phenomenon to a certain extent, and increases the diversity of the population that executes the algorithm.

4.2. Improvement Strategy of Genetic Algorithm. Genetic algorithm is a general method to solve combinatorial optimization problems, but problems such as premature phenomenon and poor local search are prone to occur in the process of solving.

4.2.1. Virus Evolutionary Genetic Algorithm. Viral evolutionary genetic algorithm is in the process of establishing the initial population of genetic algorithm, using outstanding individuals in the initial population to rebuild the initial population, and introducing a heuristic algorithm based on priority rules. In the improved Virus Evolution Genetic Algorithm (IVEGA), genetic operations are performed on the excellent subpopulations of randomly initialized populations. The relatively high average fitness value accelerates the search speed of the genetic algorithm. At the same time, the loss of the optimal solution or the satisfactory solution speeds up the convergence and evolution of the algorithm.

4.2.2. Taboo-Parallel Hybrid Genetic Algorithm. Taboo-Parallel Hybrid Genetic Algorithm introduces the idea of coarse-grained parallelism in the process of genetic operations, which effectively avoids premature phenomena; it introduces a noncircular search strategy in the mutation stage to

avoid the contradiction that the mutation probability is difficult to choose; the concept of population diversity is introduced in the calculation process, which improves the calculation efficiency of the fitness function value.

4.2.3. Improved DNA Immune Genetic Algorithm. The improved DNA immune genetic algorithm mainly uses the parallel evolution of the single processor and the vaccine to update the principle of the antibody group to accelerate the spread of excellent individuals in the various subpopulations of genetic manipulation, thereby increasing the convergence speed and the diversity of the subpopulations.

A single algorithm always shows certain shortcomings when solving the problem to be solved, and the fusion of two or more algorithms, especially modern heuristic algorithms, can not only maximize the strengths and avoid weaknesses, but also take advantage of the unique advantages of different algorithms, which can greatly enhance the local search ability of the hybrid algorithm and the convergence speed of the global search. At the same time, due to the diversity of combination optimization problems and fusion algorithms to be solved, different fusion strategies are often required according to the characteristics of different problems. Generally speaking, the hybrid algorithm has achieved the purpose of solving the problem mainly by optimizing the key parameters in the basic algorithm or controlling the calling sequence between the basic algorithms.

To sum up, the basic idea of hybrid genetic algorithm is to use traditional precise algorithm or heuristic algorithm to perform local optimization in the initial population selection phase and subpopulation selection phase, crossover phase, and mutation phase of genetic operations and then use genetic algorithm for global optimization—the best exploration.

4.3. Design of Linear Weighted Genetic Algorithm Based on Hill Climbing Operator and Fitness Value Sharing

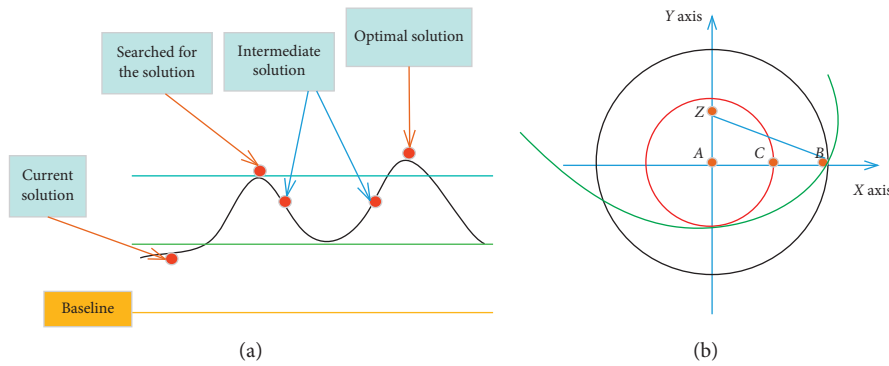
4.3.1. Hill Climbing Algorithm and Golden Section Method. Hill climbing algorithm is a heuristic algorithm with weak global search ability and strong local search ability. The algorithm uses the information obtained by positive feedback to select a better solution to replace the current solution in the adjacent space of the current solution, until a local optimal solution is obtained. In addition, because the hill climbing algorithm is a depth-first algorithm, it will be difficult to jump out of the local optimum after a local optimal solution is obtained, which greatly reduces the probability of searching for the global optimal solution. The coding implementation of the hill climbing algorithm is relatively simple, but it is easy to fall into the local optimal solution during the execution of the algorithm, thereby reducing the probability of searching for the global optimal solution. As shown in Figure 3(a), assuming that the current solution is the solution at point C, when the algorithm searches for the solution represented by point A during the execution process, no matter whether it is to the left or right, it cannot find the solution represented by point A. The solution is more optimal, and the algorithm falls into a local optimal solution at this time. In addition, the hill climbing algorithm has higher solution quality when solving small-scale NP problems, but it is difficult to find the optimal solution when solving large-scale NP problems.

The golden section method comes from Pythagorean school's study of regular pentagons and regular decagons. This algorithm is a minimum point search algorithm that divides a line segment according to a ratio. The ratio is 0.618, which is also called the 0.618 method. In modern optimization theories, the golden section method refers to the use of the Fibonacci sequence for the research and development of mathematical related theories and the use of basic mathematical knowledge to explain combinatorial optimization problems encountered in social practice. The graphical representation is shown in Figure 3(b). Among them, point A represents the center of the inner and outer circles, point B is an intersection point between the outer circle and the X axis, and point C is an intersection point between the inner circle and the X axis. Point Z is the intersection of the normal of the green arc at point B and the Y axis.

4.3.2. Improved Genetic Algorithm. Compared with other modern heuristic algorithms, the traditional standard genetic algorithm generally does not fall into the local optimal solution, and the probability of finding the global optimal solution is higher, while the hill climbing algorithm has the characteristics of strong local search ability and weak global search ability. So based on the basic genetic algorithm, the hill climbing algorithm can be used to improve the basic genetic algorithm to form a linear weighted genetic algorithm. The improved genetic algorithm not only has the strong local search ability of the hill climbing algorithm, but also has the strong global search ability of the traditional standard genetic algorithm.

When the genetic algorithm solves the problem, the crossover operator provides a large-span, coarse-grained search scheme, which is beneficial to improve the probability of searching for the global optimal solution, but the search performance for the local optimal solution is poor; gene mutation ensures the diversity of the population and the diversity of chromosomes during the execution of the algorithm, but the performance is poor in the later stage of the population evolution. Thus, AGA mutation strategies that can change the crossover operator and mutation operator can be introduced.

The AGA strategy means that, in the process of population evolution, when the fitness value of the individuals in the population is greater than the average value, the values of Pm and Pc are reduced, so as to be preserved as good individuals; when the fitness value of the individuals in the population is less than the average value, you increase the value of Pc and Pm, thereby increasing the production rate of individuals in the subpopulation. In this way, in the later stage of the algorithm execution, when the population falls into the local optimal solution and the function value is relatively close, changing the values of Pm and Pc can effectively jump out of the local optimal solution, thereby increasing the probability of searching for the global optimal solution.



(a) (b)

FIGURE 3: Principle diagram of hill climbing algorithm and golden section. (a) Schematic diagram of mountain climbing algorithm. (b) Schematic diagram of golden section.

One way to improve is

$$P_c = \begin{cases} P_{c1} - \dfrac{|P_{c1} - P_{c2}| \cdot |f_{avg} - f'|}{f_{\max} - f_{\min}}, & f_{\text{avg}} > f', \\ P_{c1}, \quad f_{avg} = f', \\ P_{c1} + \dfrac{|P_{c1} - P_{c2}| \cdot |f_{avg} - f'|}{f_{\max} - f_{\min}}, & f_{avg} > f', \end{cases} \tag{1}$$

$$P_m = \begin{cases} P_{m1} - \dfrac{|P_{m1} - P_{m2}| \cdot |f_{\text{avg}} - f|}{f_{\max} - f_{\text{avg}}}, & f_{\text{avg}} < f, \\ P_{m1}, \quad f_{\text{avg}} = f, \\ P_{m1} - \dfrac{|P_{m1} - P_{m2}| \cdot |f_{\text{avg}} - f|}{f_{\max} - f_{\text{avg}}}, & f_{\text{avg}} > f. \end{cases} \tag{2}$$

Among them, P_c and P_m represent crossover probability and mutation probability, respectively; P_{c1}, P_{c2}, P_{m1}, and P_{m2}, respectively, represent the value of crossover probability and mutation probability before and after the change. In addition, considering that the optimal individual is not destroyed, the above two formulas can be changed to

$$P_c = \begin{cases} P_{c\max} - e^{-0.62} \dfrac{f_{\text{avg}} - f'}{f_{\text{avg}} - f_{\max}} f_{\text{avg}} < f', \\ P_{c\max}, \quad f_{\text{avg}} = f', \\ P_{c\max} + e^{-0.62} \dfrac{f_{\text{avg}} - f'}{f_{\text{avg}} - f_{\max}}, & f_{\text{avg}} > f', \end{cases} \tag{3}$$

$$P_m = \begin{cases} P_{m\max} - e^{-0.38} \dfrac{f_{\text{avg}} - f}{f_{\max} - f_{\text{avg}}}, & f_{\text{avg}} < f, \\ f_{\text{avg}} \cdot P_{m\max}, f_{\text{avg}} = f, \\ P_{m\max} + e^{-0.38} \dfrac{f_{\text{avg}} - f}{f_{\max} - f_{\text{avg}}}, & f_{\text{avg}} > f. \end{cases} \tag{4}$$

In formulas (3) and (4), the golden ratio is introduced, and the optimal solution is approached successively in accordance with the principle of "equal ratio, symmetrical shrinkage, and principle of selection," which improves the speed of searching for the global optimal solution.

The hill climbing operator is to perform a hill climbing algorithm on individuals in the subpopulation according to a certain probability before the genetic algorithm is executed in the subpopulation. At the same time, the golden section method based on the contraction interval is introduced in the execution of the mountain climbing algorithm. This article assumes that the search interval of the initial population is $[ai, bi]$, X represents the population, x_i represents the individual with dimension i in the population, e is the shrinking interval ratio, n is the population size, and m represents the initial population dimension. The new search interval is solved as follows:

$$e = \frac{m+1}{2n-1} \cdot \lim_{n \longrightarrow \infty} \sum_{i=1}^{n} |a_i - b_i|, \tag{5}$$

$$a_{i+1} = \max(x_i, a_i, e_{i-1}), \tag{6}$$

$$b_{i+1} = \min(x_i, b_i, e_{i+1}). \tag{7}$$

The improved algorithm flow is shown in Figure 4.

In the later stage of algorithm execution, adaptive crossover probability and mutation probability are used for genetic operation. In this way, the fitness function values can be sorted into high fitness populations and low fitness populations. At the same time, related algorithms can be alternately performed in the cross mutation stage, which can effectively improve the diversity of the population. This helps to increase the probability that the algorithm searches for the optimal solution.

5. Experiment and Analysis

5.1. Simulation Parameter Setting. In the fog node layer, each agent node manages a fog node cluster, and the agent node effectively manages the fog node resources in the cluster. The fog node layer manages the fog node cluster separately and divides the distributed network into multiple management

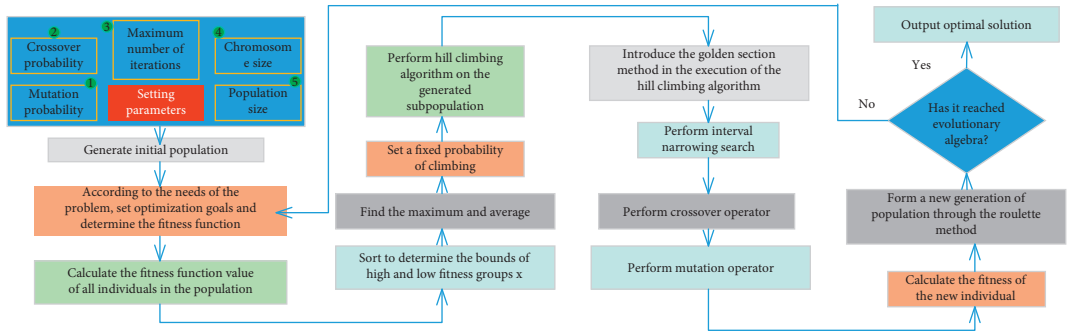


Figure 4: Improved algorithm flow.

units, which can reduce network overhead and enhance the scalability of the network. Establishing fog node clusters based on geographic location can greatly reduce service delays. At the same time, users can switch fog node clusters to provide services for them by sending requests to proxy nodes near themselves based on their geographic location. The agent node is the key to managing the resources of the fog node cluster. Its role is mainly in resource monitoring, resource evaluation and allocation, network management, payment, service quality monitoring, and fog node reputation evaluation. Resource monitoring means that the agent node monitors the idle status of the fog node resources in the cluster at any time. If a user request comes in, it can directly allocate resources. According to the needs of users, the proxy node evaluates its preference information and allocates fog computing resources that meet the needs of users in the fog node cluster. The agent nodes can communicate directly, and at the same time, the agent node is responsible for the communication function of the entire fog node cluster and the external network and at the same time controls the communication between the fog nodes in the fog node cluster. The proxy node will pay some rental fees to the fog node based on the amount of resources provided by the fog node in the cluster and its asking price. When the fog node provides resources, the proxy node will evaluate the quality of its service. If there is a violation, the proxy node can impose necessary punishment on the fog node. Reputation evaluation means that when the agent node selects the fog node in the cluster to provide services to users, it uses historical service information to evaluate the reputation of the fog node to obtain a more reliable service.

In the simulation experiment, the input is a task graph based on a directed acyclic graph, and a random network topology graph is used. The average value of the node degree is 3, and the fog nodes are connected with a probability of 0.5. The experimental parameters are set as follows: the range of the number of tasks is [20, 200], the range of task calculations is [5, 100] MB, the communication volume range of the task is [5, 100] MB, and the data volume range of nonforward nodes required by the task is [5, 50] MB. The range of the number of fog nodes is [6, 10], the processing rate range of fog nodes is [5, 20] MB/sec, the transmission rate range between fog nodes is [50, 200] Mbps, and the calculation unit price range of fog node resources is [2, 2] cent/sec. The simulation parameter setting is shown in Table 1.

5.2. Analysis of Simulation Results. In this experiment, we compare the pros and cons of the three algorithms in terms of delay, communication load, and service cost. These three algorithms are all based on improved genetic algorithms. The first one is based on adaptive genetic algorithm and is denoted as AGA; the second is RAS-IN algorithm, which is a genetic algorithm for nondominated sorting with an elite strategy. The algorithm is an improved algorithm for calculating the crowding distance; the third is the linear weighted genetic algorithm, which is recorded as the LW-GA.

In this simulation, we set the global maximum crossover rate to 1.01 and the global maximum mutation rate to 0.12. In the simulation process, we set the number of populations to 100, the maximum number of iterations to 300, and the number of fog nodes to 10. The optimization weight is set to 0.45. As shown in Figures 5–7, as the number of tasks continues to increase, the linear weighted genetic algorithm proposed in this paper has better optimization performance in terms of delay, communication load, and service cost. Through the comparison of experimental results, it can be found that the performance of the LW-GA has significant advantages.

The number of nondominated individuals in the temporary population is a criterion for judging the diversity of the population. We will count the number of nondominated individuals in the temporary population of the temporary population based on the linear weighted genetic algorithm and the other two algorithms under different iteration times. Here we set the maximum number of iterations to 400, the number of populations to 150, and the number of tasks to 30. The other settings are the same as the above parameters. As shown in Figure 8, we can see that, in the early stage of implementation, the number of nondominated individuals in the temporary population of the linearly weighted genetic algorithm grows slowly, while it grows rapidly in the midterm and tends to a final stable state earlier. This shows that the linear weighted genetic algorithm has more advantages and better adaptability.

The average value of the crowding degree distance between two solutions in the optimal solution set is an evaluation index of the quality of the distribution. The smaller the average value of the congestion distance, the more uniform the distribution of the optimal solution for fog computing resource allocation obtained by the algorithm.

Table 1: Simulation parameter settings.

Parameter	Range
Number of fog nodes	[6, 12]
Processing rate of the fog node	[5, 20] MB/sec
Amount of data of nonforward nodes required by the task	[5, 50] MB
Number of tasks	[30, 360]
Transmission rate between fog nodes	[50, 200] Mbps
Task calculation	[5, 100] MB
Task traffic	[5, 100] MB
Calculation unit price of fog node resources	[2, 2] cent/sec

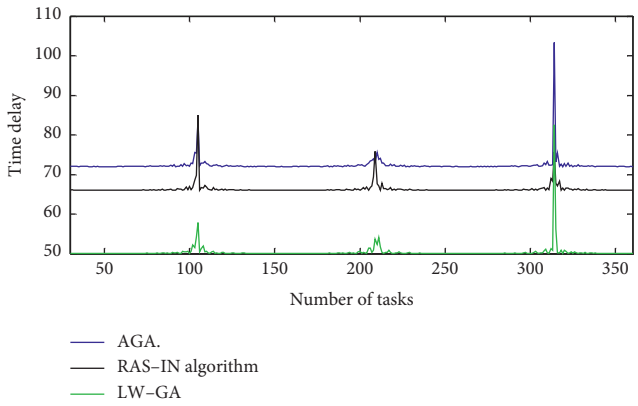


Figure 5: Delay comparison of the three algorithms.

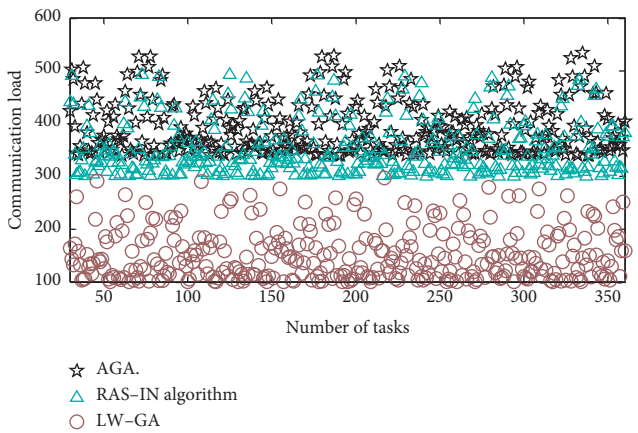


Figure 6: Comparison of the communication load of the three algorithms.

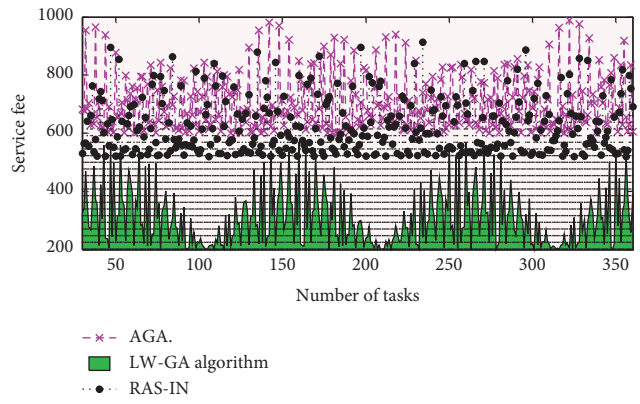


Figure 7: Comparison of service fees of the three algorithms.

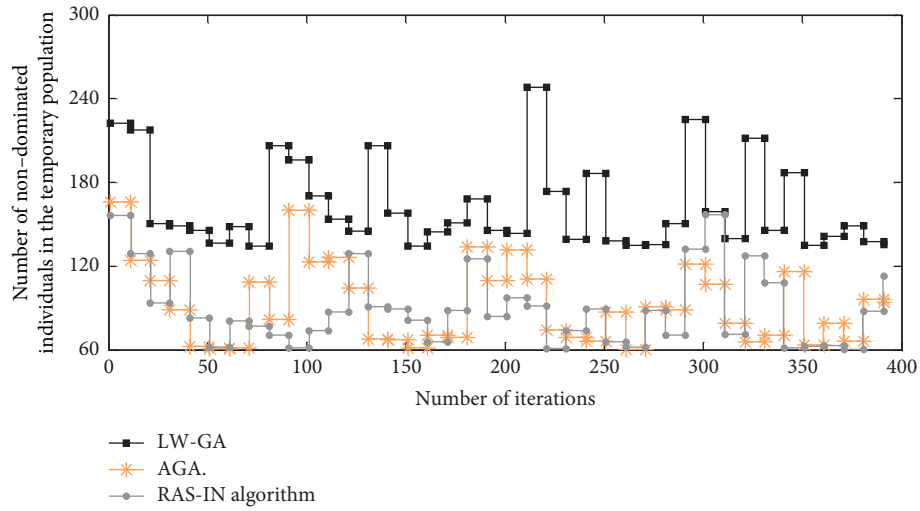


Figure 8: The number of nondominated individuals in the temporary population under different iteration times.

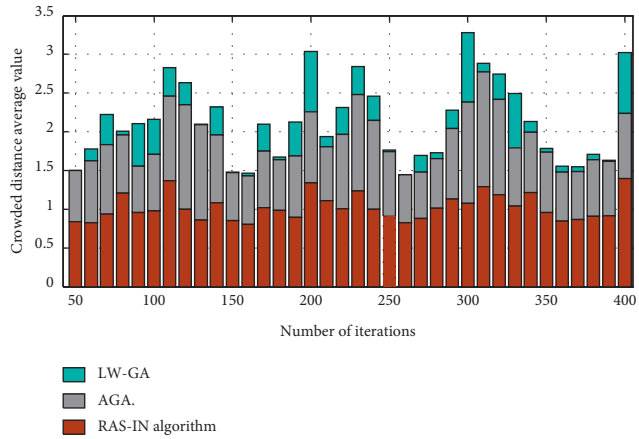


Figure 9: Comparison of the average crowded distance of the three algorithms.

The more uniform the distribution of the optimal solution set, the better the individual distribution and the more practical the significance of the experimental results.

In order to evaluate the algorithms proposed in this section, this article evaluates the AGA, the RAS-IN algorithm, and the LW-GA. In this experiment, we set the maximum number of iterations to 400, the number of populations to 150, and the number of tasks to 30. The other settings are the same as the above parameters. It can be seen from Figure 9 that the average value of the crowded distance between two adjacent solutions in the Pareto solution set obtained by the linear weighted genetic algorithm is smaller than the AGA and the RAS-IN algorithm. This shows that the nondominated solution set of the linearly weighted genetic algorithm has better distribution in the fog computing environment.

6. Conclusion

Based on the fog computing architecture proposed by Cisco, taking into account the flexibility, scalability, and resource utilization of the architecture, this paper proposes an adaptive fog computing architecture, which mainly includes a comprehensive perception layer, a fog computing layer, and a cloud computing layer. In the fog computing layer, network element entities such as fog group, fog management node, and fog unit node are introduced. This paper proposes a fog resource scheduling scheme based on linear weighted genetic algorithm, which converts the multiobjective optimization problem into a single-objective optimization problem. When applying genetic algorithms based on weighted sums, preference weights are assigned to delays, communication loads, and service costs, and they are integrated into an objective function to perform genetic operations to obtain a better solution. This algorithm can greatly reduce the nondominant relationship between solutions in multiobjective optimization problems. The simulation results show that the improved strategy proposed in this paper shows better performance in the three optimization goals of delay, communication load, and service cost and improves the search efficiency and ensures the uniformity of the optimal solution set distribution. However,

when the fog node model is established in this article, the fog node resource model is established using the fuzzy processing method, and in the resource scheduling problem, the actual resource dimension of the fog node is not considered. The next step of research can be to match nodes of different resource dimensions or resource attributes for different types of services.

Data Availability

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

Conflicts of Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Y. Sun, F. Lin, and H. Xu, "Multi-objective optimization of resource scheduling in fog computing using an improved NSGA-II," *Wireless Personal Communications*, vol. 102, no. 2, pp. 1369–1385, 2018.

[2] S. Bitam, S. Zeadally, and A. Mellouk, "Fog computing job scheduling optimization based on bees swarm," *Enterprise Information Systems*, vol. 12, no. 4, pp. 373–397, 2018.

[3] M. Yang, H. Ma, S. Wei, Y. Zeng, Y. Chen, and Y. Hu, "A multi-objective task scheduling method for fog computing in cyber-physical-social services," *IEEE Access*, vol. 8, pp. 65085–65095, 2020.

[4] O. Skarlat, M. Nardelli, S. Schulte, M. Borkowski, and P. Leitner, "Optimized IoT service placement in the fog," *Service Oriented Computing and Applications*, vol. 11, no. 4, pp. 427–443, 2017.

[5] Y. Liu, J. E. Fieldsend, and G. Min, "A framework of fog computing: architecture, challenges, and optimization," *IEEE Access*, vol. 5, pp. 25445–25454, 2017.

[6] J. Wan, B. Chen, S. Wang, M. Xia, D. Li, and C. Liu, "Fog computing for energy-aware load balancing and scheduling in smart factory," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 10, pp. 4548–4556, 2018.

[7] R. O. Aburukba, M. AliKarrar, T. Landolsi, and K. El-Fakih, "Scheduling Internet of Things requests to minimize latency in hybrid Fog-Cloud computing," *Future Generation Computer Systems*, vol. 111, pp. 539–551, 2020.

[8] C.-g. Wu, W. Li, L. Wang, and A. Y. Zomaya, "An evolutionary fuzzy scheduler for multi-objective resource allocation in fog computing," *Future Generation Computer Systems*, vol. 117, pp. 498–509, 2021.

[9] S. H. H. Madni, M. S. A. Latiff, J. Ali, and S. I. M. Abdulhamid, "Multi-objective-Oriented cuckoo search optimization-based resource scheduling algorithm for clouds," *Arabian Journal for Science and Engineering*, vol. 44, no. 4, pp. 3585–3602, 2019.

[10] C. Guerrero, I. Lera, and C. Juiz, "Evaluation and efficiency comparison of evolutionary algorithms for service placement optimization in fog architectures," *Future Generation Computer Systems*, vol. 97, pp. 131–144, 2019.

[11] D. Zhang, F. Haider, M. St-Hilaire, and C. Makaya, "Model and algorithms for the planning of fog computing networks," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 3873–3884, 2019.

[12] P. Hosseinioun, M. Kheirabadi, S. R. Kamel Tabbakh, and R. Ghaemi, "A new energy-aware tasks scheduling approach in fog computing using hybrid meta-heuristic algorithm," *Journal of Parallel and Distributed Computing*, vol. 143, pp. 88–96, 2020.

[13] S. Wang, T. Zhao, and S. Pang, "Task scheduling algorithm based on improved firework algorithm in fog computing," *IEEE Access*, vol. 8, pp. 32385–32394, 2020.

[14] X. Li, Y. Liu, H. Ji, H. Zhang, and V. C. M. Leung, "Optimizing resources allocation for fog computing-based Internet of Things networks," *IEEE Access*, vol. 7, pp. 64907–64922, 2019.

[15] C. Tang, S. Xia, Q. Li et al., "Resource pooling in vehicular fog computing," *Journal of Cloud Computing*, vol. 10, no. 1, pp. 1–14, 2021.

[16] C. Tang, S. Xia, C. Zhu, and X. Wei, "Phase timing optimization for smart traffic control based on fog computing," *IEEE Access*, vol. 7, pp. 84217–84228, 2019.

[17] S. Hu and G. Li, "Dynamic request scheduling optimization in mobile edge computing for IoT applications," *IEEE Internet of Things Journal*, vol. 7, no. 2, pp. 1426–1437, 2019.

[18] M. Abbasi, E. Mohammadi-Pasand, and M. R. Khosravi, "Intelligent workload allocation in IoT-Fog-cloud architecture towards mobile edge computing," *Computer Communications*, vol. 169, pp. 71–80, 2021.

[19] M. K. Hussein and M. H. Mousa, "Efficient task offloading for IoT-based applications in fog computing using ant colony optimization," *IEEE Access*, vol. 8, pp. 37191–37201, 2020.

[20] K. Matrouk and K. Alatoun, "Scheduling algorithms in fog computing: a survey," *International Journal of Networked and Distributed Computing*, vol. 9, no. 1, pp. 59–74, 2021.

[21] N. Bhalaji, "Delay diminished efficient task scheduling and allocation for heterogeneous cloud environment," *Journal of Trends in Computer Science and Smart Technology (TCSST)*, vol. 1, no. 01, pp. 51–62, 2019.

[22] M. Abdullahi, M. A. Ngadi, and S. I. Dishing, "An efficient symbiotic organisms search algorithm with chaotic optimization strategy for multi-objective task scheduling problems in cloud computing environment," *Journal of Network and Computer Applications*, vol. 133, pp. 60–74, 2019.

[23] Z. Zhang, M. Liu, M. Zhou, and J. Chen, "Dynamic reliability analysis of nonlinear structures using a Duffing-system-based equivalent nonlinear system method," *International Journal of Approximate Reasoning*, vol. 126, pp. 84–97, 2020.

[24] J. Wang, P. Zhu, B. He et al., "An adaptive neural sliding mode control with ESO for uncertain nonlinear systems," *International Journal of Control, Automation and Systems*, pp. 1–11, 2020.

[25] Y. Lu, Y. Qi, S. Qi et al., "Secure deduplication-based storage systems with resistance to side-channel attacks via fog computing," *IEEE Sensors Journal*, p. 1, 2021.

[26] J. Wen, J. Yang, B. Jiang, H. Song, and H. Wang, "Big data driven marine environment information forecasting: a time series prediction network," *IEEE Transactions on Fuzzy Systems*, vol. 29, no. 1, pp. 4–18, 2021.

[27] W. Wei, X. Fan, H. Song et al., "Imperfect information dynamic stackelberg game based resource allocation using hidden Markov for cloud computing," *IEEE Transactions on Services Computing*, vol. 11, no. 1, pp. 78–89, 2016.

[28] J. Wen, J. Yang, B. Jiang et al., "Big data driven marine environment information forecasting: a time series prediction network," *IEEE Transactions on Fuzzy Systems*, 2020.

[29] M. Taneja, N. Jalodia, J. Byabazaire, A. Davy, and C. Olariu, "SmartHerd management: a microservices-based fog computing-assisted IoT platform towards data-driven smart dairy farming," *Software: Practice and Experience*, vol. 49, no. 7, pp. 1055–1078, 2019.