QoS and Energy-aware Resource Allocation in Cloud Computing Data Centers using Particle Swarm Optimization Algorithm and Fuzzy Logic System

Yu Wang[1*], Lin Zhu[2]
School of Computer Science, Xi'an Aeronautical Institute, Xi'an, Shaanxi, 710077, China[1]
Motorcycle Test Technology Institute of China South Industries Group Corporation[2]
Xi'an, Shaanxi, 710032, China[2]

***Abstract*—Cloud computing has become a viable option for many organizations due to its flexibility and scalability in providing virtualized resources via the Internet. It offers the possibility of hosting pervasive applications in the consumer, scientific, and business domains utilizing a pay-as-you-go model. This makes cloud computing a cost-effective solution for businesses as it eliminates the need for large investments in hardware and software infrastructure. Furthermore, cloud computing enables organizations to quickly and easily scale their services to meet the demands of their customers. Resource allocation is a major challenge in cloud computing. It is known as the NP-hard problem and can be solved using meth-heuristic algorithms. This study optimizes resource allocation using the Particle Swarm Optimization (PSO) algorithm and fuzzy logic system developed under the proposed time and cost models in the cloud computing environment. Receiving, processing, and waiting time are included in the time model. The cost model incorporates processing and receiving costs. Two experiments demonstrate the performance of the proposed algorithm. The simulation results demonstrate the potential of our mechanism, demonstrating improved performance over previous approaches in aspects such as providers' total income, users' total revenue, resource utilization, and energy consumption.**

Keywords*—*Cloud computing; resource allocation; scheduling; PSO; fuzzy logic

I. Introduction

Cloud computing offers on-demand access to various computing resources, including software, platforms, and storage [1]. Coupled with IoT and big data applications, it has revolutionized information technology operations, becoming the enabling technology for next-generation communications [2]. Energy consumption is a significant challenge for cloud data centers, given the substantial and constantly evolving size of cloud computing infrastructures and the rapidly growing number of users [3]. From 2005 to 2010, there has been an average annual increase of 12% in energy consumption, which has intensified over the past few years [4]. Excessive energy consumption produces excessive heat emissions and increases costs, resulting in a degradation of system reliability and performance [5]. As energy costs rise and availability diminishes, data center resource management should be optimized for energy efficiency while ensuring high service levels [6]. Consequently, cloud service providers must guarantee that rising energy costs do not adversely affect their profit margins. Rising energy costs seriously threaten cloud infrastructures as they increase the Total Cost of Ownership (TCO) and reduce the Return on Investment (ROI) [7].

Energy efficiency in data centers is a complex problem since computing applications and data grow rapidly, and larger servers and disks are required to meet the processing times [8]. Green cloud computing aims to optimize the processing and management of computing infrastructure while reducing energy consumption [9]. The success of cloud computing depends on the sustainability of its future growth. The advent of cloud computing with increasingly pervasive frontend client devices interacting with backend data centers could cause energy consumption to skyrocket [10]. To promote green cloud computing, data centers should be operated efficiently. Cloud resources should be allocated according to user-specified Quality of Service (QoS) criteria via Service Level Agreements (SLAs) and minimize energy consumption. Multiple subscribers are served by combining the resources in the cloud [11]. Using a multi-tenancy model, the provider dynamically multiplexes the resources (physical and virtual) according to the requirements of each tenant [12]. Based on the lease and SLA agreement, the number of virtual resources will be assigned based on the needs of each client. As a result, as cloud service demand has grown, providers have had to scale up the number of resources and capabilities of cloud-based services to handle the increasing resource demands [13]. Fig. 1 shows a process for allocating resources in a cloud environment.

Integrating Internet of Things (IoT), machine learning, deep learning, and neural networks within cloud resource allocation marks a transformative shift in addressing the complexities of modern computing landscapes [14]. The IoT introduces a vast network of interconnected devices and sensors, generating copious data streams requiring efficient processing and resource allocation [15, 16]. Machine learning, especially when combined with deep learning and neural networks, enables cloud systems to learn, adapt, and make data-driven decisions, facilitating predictive analytics for demand forecasting and user behavior analysis [17, 18]. These technologies empower cloud resource allocation mechanisms by automating decision-making processes, optimizing resource distribution, and enhancing the scalability of computing systems [19]. Neural networks, a subset of deep learning, allow for pattern recognition, predictive modeling, and intelligent decision-making, ensuring more accurate and adaptive resource allocation strategies [20, 21]. Leveraging IoT data and machine learning capabilities within cloud resource allocation not only enhances system efficiency but also allows for dynamic adjustments, adaptive resource scaling, and predictive provisioning, ultimately leading to improved QoS and streamlined cloud operations in a rapidly evolving technological landscape [22].

The significance of meta-heuristic algorithms in cloud resource allocation lies in their capacity to efficiently navigate cloud environments' complex, dynamic, and constantly evolving landscape, offering optimized solutions amidst varying user demands and operational challenges. This paper introduces a hybrid optimization algorithm to address the issues in multi-cloud resource allocation. Particle Swarm Optimization (PSO) algorithm and fuzzy logic system are combined as a hybrid approach to reduce the problems in multi-cloud resource allocation. The selected optimization algorithms are known for their optimal global solutions and rapid convergence characteristics. While PSO exhibits robust optimization capabilities, the integration of fuzzy logic manages the uncertainties and imprecisions inherent in the dynamic nature of cloud environments. Fuzzy logic enhances the adaptability and robustness of decision-making, particularly in scenarios involving vague or uncertain data, thereby augmenting resource allocation's overall accuracy and effectiveness. This combined approach prioritizes QoS criteria and energy efficiency in cloud data center resource allocation. The principal contributions of this paper can be summarized as follows:

- Combining the PSO algorithm and fuzzy logic system for solving the resource allocation problem in cloud computing.
- Enhancing resource utilization and reducing the execution time of the resource allocation problem.
- Increasing user and provider utility and reducing the generational distance of the resource allocation problem.

This paper presents an efficient resource allocation model that takes advantage of the benefits of optimization algorithms. The remainder of the paper is organized in the following manner. Section II reviews the previous cloud resource allocation approaches. Section III describes the proposed cloud resource allocation mechanism. Experimental results are reported in Section IV. Section V concludes the paper.

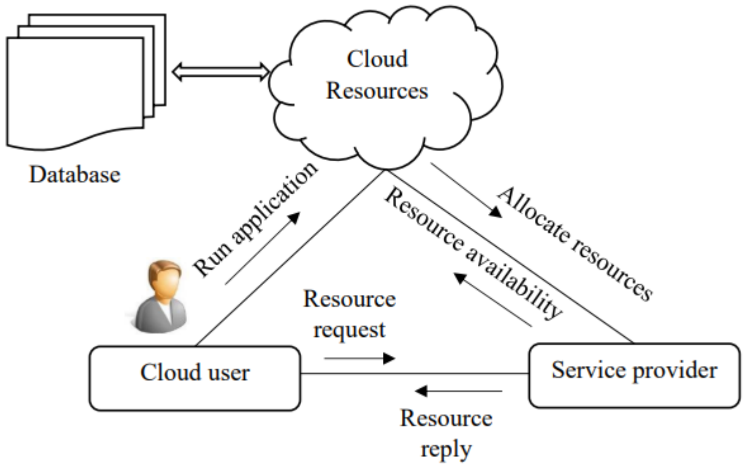

Fig. 1. Cloud resource allocation process.

II. Related Work

Wang and Su [23] developed an algorithm for dynamically allocating resources among numerous cloud nodes operating in a big data context. Based on computing power and storage factors, this algorithm uses fuzzy pattern recognition to divide nodes and tasks into distinct levels. Therefore, a dynamic mapping between tasks and nodes is generated. Upon the arrival of a new task, only the nodes corresponding to the task level will join the bid. The algorithm uses a hierarchical approach to minimize communication traffic during resource allocation. Based on the results of experiments, the presented algorithm is more efficient regarding makes span and communication traffic than the Min-Min algorithm.

The cloud-based disassembly proposed by Jiang, et al. [24] abstracts the disassembly factory as a disassembly resource, allowing it to be allocated to disassembly tasks. Based on this model, a cloud-based disassembly solution is developed that offers users a disassembly service tailored to their needs. Disassembly services are execution plans for tasks derived from scheduling and allocating disassembly tasks. The paper uses a mathematical model to describe the disassembly service formally by taking into account the uncertainty associated with disassembly processes and the precedence relationships between tasks involved.

Mousavi, et al. [25] presented a hybrid approach to load balancing that integrates Grey Wolves Optimization (GWO) and Teaching-Learning-Based Optimization (TLBO) algorithms, aiming to maximize throughput by balancing virtual machine loads and avoiding a local optimum trap. The algorithm is evaluated on eleven benchmark functions, and comparisons are made with particle swarm optimization (PSO), biogeography-based optimization (BBO), and GWO.

Cloud computing is characterized by elasticity, distinguishing it from other paradigms, such as cluster and grid computing. Based on the bio-inspired coral-reef optimization paradigm, Ficco, et al. [26] developed a meta-heuristic approach to cloud resource allocation. The resource reallocation schema was optimized using classic Game Theory based on cloud provider optimization objectives and customer requirements expressed through fuzzy linguistic SLAs.

Chen, et al. [27] presented a self-adapting resource allocation methodology that consists of several feedback loops, each involving a PSO-based runtime decision algorithm and an iterative QoS prediction model. Each iteration of the algorithm improves QoS values. Future resource allocation operations are determined based on the predicted QoS value and the PSO-based runtime decision algorithm. As the PSO-based algorithm iterates, no further improvements are suggested compared to current resource allocations. The proposed method is evaluated on the RUBiS benchmark, highlighting a 20% improvement in QoS prediction accuracy compared to the current state of the art based on the same historical data.

Singhal and Singhal [28] developed a Feedback-based Combinatorial Fair Economical Double Auction Resource Allocation Model (FCFEDARA) to determine provider genuineness based on the prices offered and feedback from customers. The proposed framework enables customers to access resources from different providers at the best prices and prioritizes genuine providers with good feedback over non-genuine providers with bad reviews. Providers and customers submit bundle bids and resource lists in the combinatorial double auction model. By assessing provider truthfulness, penalizing market spoilers, and giving preference to providers with positive feedback from customers, the proposed model takes care of the truthfulness of providers.

Thakur and Goraya [29] introduced a novel metaheuristic-based resource allocation approach for load balancing in cloud environments. The goal is to effectively reduce the uneven distribution of workloads between physical machines in addition to their resource capabilities. Consequently, the over- or under-loading of active physical machines is prevented. To develop a suitable resource allocation strategy for load balancing, dragonfly and PSO algorithms are combined. The proposed algorithm is superior to PSO, dragonfly algorithm, and comprehensive learning PSO in determining optimal resource allocation.

III. Proposed Method

A new PSO algorithm is used in this paper to select the best member of the population. This algorithm outperforms existing multi-objective optimization techniques regarding calculation time, reasonable undefeated solutions distribution, and Pareto front convergence. Moreover, the Fuzzy set theory is used in this paper to select the best adaptive solution.

A. *Mathematical Formulation of the Problem*

Each user requests a combination of the requested resources with different attributes, the required number of the resources, and a proposed cost to buy all the resources in a bundle form. Each provider presents a combination of the resources with different attributes, the number of the presented resources, and a proposed cost to sell all the resources in a bundle form. The bundle is a request to buy with all bought products and a sell request with all products. Moreover, the meaning of a specific attribute of the requested items is the number showing the processor processing power, the accumulator capacity, bandwidth, and so on. The total number of requested resources should equal or less than the total number of all the presented resources. All the users requested resources' attributes should be equal to or less than all the presented attributes by the cloud presenter to assign the resources. After determining which provider can meet the user's requests, the cost that the user should pay to the provider is determined by a costing model. The costing model should be fair and beneficial for the provider and the user. The costing model used in this paper is the presented method in [30] and [31]. Table I lists the symbols and variables used in the equations.

TABLE I. SYMBOLS AND VARIABLES

Symbol	Definition
N	Population size
n	Number of iterations
D	Number of dimensions
Rq	Users' requests
Sq	Providers' services
Rc	Average cost per user
Sc	The average cost for each provider
RSc	Average business cost
C_u^p	The paid cost by the user u to the provider p

Eq. (1) and Eq. (2) show the total number of a user's requested and presented items by the provider. Eq. (3) allows for dividing the user's suggested cost by the total number of requested products to determine the average cost per user. The average cost for each provider is the division of the proposed cost by the provider on the total number of provider items shown in Eq. (4). The average business cost of provider p and user u is determined based on the users and the provider's average costs using Eq. (5). The paid cost by the user u to the provider p is estimated through the number of assigned resources using Eq. (6). The earnings of each provider are presented in Eq. (7), which is equal to the paid cost minus the proposed cost in Eq. (6). The higher cost received by the provider for its resources than the expected cost leads to higher earnings by the provider.

Moreover, in each user's income, Eq. (8) equals the proposed cost of the user minus its paid in Eq. (6). It is also clear that more paid cost by the user to rent the requested resources to the provider than its proposed cost makes more earning for the user. The resource utilization rate in Eq. (9) is equal to the ratio of the total number of requested items by a user to the total items presented by the provider. Objective functions in Eq. (10), Eq. (11), and Eq. (12) show the total earnings of the providers, the total income of the users, and total resource utilization, respectively. The objective area in the proposed algorithm is three-dimensional, as shown in Eq. (13).

Moreover, the requested resources should not be more than the provided resources controlled by Eq. (14).

$$Rq_u = \sum_{l=1}^{k} q_k{}^u \qquad (1)$$

$$Sq_p = \sum_{l=1}^{k} q_k^p \qquad (2)$$

$$Rc_u = \frac{\cos t_u}{Rq_u} \qquad (3)$$

$$Sc_p = \frac{\cos t_p}{Sq_p} \qquad (4)$$

$$RSc_u{}^p = \left(\frac{Rc_u + Sc_p}{2} \right) \qquad (5)$$

$$c_u{}^p = RSc_u{}^p \times Rq_u \qquad (6)$$

$$Submitcost_p = c_u{}^p - cost_p \qquad (7)$$

$$Requestcost_u = cost_u - c_u{}^p \qquad (8)$$

$$quantity_p = \frac{Rq_u}{sq_p} \qquad (9)$$

$$f_1 = \sum_{p=1}^{m} (Submitcost_p) \qquad (10)$$

$$f_2 = \sum_{u=1}^{n} \left(Requestcost_u \right) \qquad (11)$$

$$f_3 = \sum_{u=1}^{n} \sum_{p=1}^{m} \left(quantity \right) \qquad (12)$$

Finess= [f_1 f_2 f_3] (13)

$$\sum_{u=1}^{n} \sum_{l=1}^{k} (q^u{}_k \le q^p{}_k \, and \;\; a^u{}_k \le a^p{}_k) \qquad (14)$$

B. The Proposed Method for Cloud Resource Assignment

The proposed multi-objective method is proposed based on the PSO algorithm. Here, the algorithm and the way of making it multi-objective are explained. Then, the steps of the proposed method are presented:

1) Particle swarm optimization algorithm: The PSO algorithm employs a population-based stochastic process. Particles move through the search space of an optimization problem. Particles' positions represent potential solutions. The particles search the search space for better positions by modifying their velocities under rules derived from behavior models of flocking birds. The PSO algorithm is known for its adeptness at approaching near-optimal solutions, a characteristic pivotal in addressing resource allocation concerns within cloud computing environments. The time complexity of the PSO algorithm typically operates at O(n*N), where n represents the number of iterations and N stands for the population size or number of particles in the swarm. In terms of space complexity, it generally stands at O(N*D), with D representing the number of dimensions in the given problem space.

2) Multi-objective particle swarm optimization algorithm: In MOPSO, a concept called the hall of fame or repository is used so that from the investigated best answers, the best of them that are undefeated answers are stored in a repository. These repository members are an approximation of the Pareto front. Each particle in MOPSO selects one repository's member as a leader when it wants to move. It is the experimented best position of that particle. However, the answers are distributed on a multi-dimensional plate. Vertical and horizontal lines should be used to tabulate the area initially. The cells in the space are then identified, including the repository members. Some cells may have repository members, but the priority is for the cells with less population because of maintaining diversity. One of the less congested cells is selected using the Roulette Wheel Selection method, and then one of the cell's members is selected as leader randomly. Each particle moves using Eq. (15) and Eq. (16).

$$v_i{}^{(it+1)} = wv_i^{(it)} + c_1 r_1 (\text{pbest}_i^{(it)} - X_i^{(it)}) + c_2 r_2 (\text{gbest}^{(it)} - X_i^{(it)}) \qquad (15)$$

$$X_i^{(it+1)} = X_i^{(it)} + v_i^{(it+1)} \qquad (16)$$

The following comparisons should be made between the new role and the top memory: 1) If the new position beats the best memory, the new position replaces the best memory. 2) Nothing is performed if the best memory beats the new position. 3) If none of them is defeated, one of the positions is considered the best memory randomly. Then, undefeated members of the current population are the elites. Now, the quality of the response is controlled. Selecting the repository's member to remove is performed using the Roulette Wheel Selection method, but the cell with fewer roles in the diversity of the answers is selected. Here, the priority is for the cells with more population. The lack of memory has a limit on the repository size. The new members should be removed using the Roulette Wheel Selection method, depending on the crowded cells if the number of repository members exceeds the calculated capacity. A multi-objective problem that requires an agent in every square inch of space is being explored, and the PSO is an algorithm with a high convergence speed. As a result, a mutation operator slows down convergence to ensure the entire space is thoroughly examined [31].

3) The proposed multi-objective particle swarm optimization algorithm through crowding distance

A new particle swarm optimization algorithm called $MOPSO-CD$ is proposed in [32]. A mutation operator is used in this algorithm, like the $MOPSO$ algorithm. Moreover, the crowding distance approach used in this algorithm is proposed in [33] and used in the $NSGA-II$ algorithm. This method is also used in $MOPSO-CD$ to find the optimal answers. The crowding distance for each answer approximates the answers' density around it. The studied problem has three objective functions. At first, the objective function values of each dimension are sorted decreasingly. Then, the previous and next points are selected rather than the problem's objective functions. The fractions of the dimension covered by the ith member of the population in the first, second, and third objective functions are obtained by Eq. (17), (18), and (19), and the crowding distance is obtained by Eq. (20).

$$d_i^1 = \frac{\left|f_1^{i+1} - f_1^{i-1}\right|}{f_1^{\max} - f_1^{\min}} \tag{17}$$

$$d_i^2 = \frac{\left|f_2^{i+1} - f_2^{i-1}\right|}{f_2^{\max} - f_2^{\min}} \tag{18}$$

$$d_i^3 = \frac{\left|f_3^{i+1} - f_3^{i-1}\right|}{f_3^{\max} - f_3^{\min}} \tag{19}$$

$$d_i = d_i^1 + d_i^2 + d_i^3 \tag{20}$$

It is beneficial if the population's ith member covers a bigger area. Hence, the higher priority is for the undefeated answers with higher crowding distance in the PSO algorithm. The undefeated answers in the external repository are sorted through the decreasing crowding distance decreasingly. Then, in each step, one of the top 10% of answers is selected randomly. If the repository is full, one of the last 10% of answers is selected randomly, and the new undefeated answer found in the last iteration replaces it. Algorithm 1 shows different steps of the PSO algorithm. Before the start of the main loop of the algorithm, the users' requests list and the providers' offerings, including the attributes, number, and proposed cost, should be received. Moreover, in the proposed algorithm, the particles' positions are defined as follows:

Algorithm 1. PSO algorithm

```
1. Initialize the population:
  1.1. Generate Xi
  1.2. Set particle velocity vi to zero (vi=0)
  1.3. Evaluate the fitness value of particle Xi
  1.4. Set the best position of each particle as Pbesti = Xi
  1.5. Update the global best position gbest with the best particle Xi
2. Initialize the number of iterations it = 0
3. Save undefeated answers of Xi in rep
4. Begin iteration:
  4.1. Calculate the crowding distance for each undefeated answer in rep
  4.2. Sort undefeated answers in rep based on their crowding distance in decreasing order
  4.3. For each particle Xi from 1 to nPop:
    4.3.1. Randomly select an optimal guide from the top 10% of the sorted rep for particle Xi and update its position in gbest
    4.3.2. Compute the new speed of each particle using equations (3-15) with c1=1 and c2=1
    4.3.3. Calculate the new position of each particle using equations (3-16)
    4.3.4. Adjust variable values of Xi to fit within the determined limits; if Xi exceeds the limits, reverse its particle speed by multiplying it by -1
    4.3.5. Implement a mutation operation on Xi
    4.3.6. Evaluate the objective function of Xi
  4.4. Update undefeated answers in rep:
    4.4.1. Calculate the crowding distance for each undefeated answer in rep
    4.4.2. Sort the undefeated answers in rep based on crowding distance in decreasing order
    4.4.3. Randomly replace one of the lower 10% of the sorted rep with the new answer Xi
  4.5. Update the best position of each particle if the new position defeats the stored position in memory
  4.6. Increment the iteration count it
5. Repeat steps 3 to 4 until the maximum number of iterations is reached
```

4) The fuzzy-based approach for the adaptive solution: Multi-objective optimization algorithms do not result in only one answer; a set of undefeated answers is obtained, the approximations of the first front. If a final answer is required, one of the answers should be selected as the resource allocation objective. To this aim, different methods are presented and used now, like the Fuzzy Set theory *[34]* . A membership function for each objective function is considered in *[35]*, given by Eq. (23).

$$\mu_o^s = \begin{cases} 1 & F_o \le F_o^{min} \\ \dfrac{F_o^{max} - f_o^s}{F_o^{max} - F_o^{min}} & F_o^{min} < f_o^s < F_o^{max} \\ 0 & F_o \ge F_o^{max} \end{cases} \tag{23}$$

A normalized membership function is obtained using Eq. (24) for each undefeated solution. The best adaptive solution is the answer with the most value for μ^s.

$$\mu^s = \frac{\sum_{o=1}^{Nobj} \mu_o^s}{\sum_{s=1}^{S} \sum_{o=1}^{Nobj} \mu_o^s} \tag{24}$$

IV. Simulation

The proposed algorithm is simulated on a Microsoft Windows system using Matlab. Experiments are divided into three categories: Small-Scale (SS), Middle-Scale (MS), and Large-Scale (LS). The total number of users and providers related to the three experiments is (20, 5), (15, 50), and (30, 100), respectively. The population size is 75, the maximum number of repetitions is 100, and the personal and collective learning coefficients are 1. The Inertia Weight is W=0.4, and the mutation rate is mu=0.5. The resource attributes are as follows:

- The power and speed of the computer processor are measured by Million Instruction per Second (MIPS) with the range of [220, 1000].
- The memory shows the amount of memory in MB with the range of [256, 512, 1024, 2048].
- The accumulator shows the amount of accumulator in MB with the range of [1500, 40000].
- Bandwidth shows the amount of bandwidth in bits per second with the range of [120, 1000].
- The proposed cost is expressed as the cost unit per million instructions with the range of [0.012-0.1046].

A. *Performance Measures*

The performance measures of this work are as follows:

- Total earnings of the provider: proportional to Eq. (10)
- Total incomes of the users: proportional to Eq. (11).
- Total resource utilization: proportional to Eq. (12).
- Generation distance: proportional to the presented model in [26].
- The distance: proportional to the model in [27].

B. *The Experimental Results*

Two distinct experiments are conducted in this study. Each experiment compares the performance of the suggested method with that of the other methods.

1) First experiment: In this experiment, the proposed method performance is compared with NSGA-II *[36]* and MOPSO *[37]* algorithms in terms of the answers' quality, generation distance, distance, and execution time. Tables II to IV show the comparison of these three algorithms comparisons through the previously explained six measures for three types of experiments. Based on three types of experiments, the results for the provider's total earnings, the total users' income, the total resource utilization, and the generation distance for each algorithm are shown in Fig. 2 to Fig. 5. The performance of the proposed algorithm is superior to those of the other three algorithms. Fig. 6 to Fig. 7 illustrate the distance and execution time results for the mentioned algorithms. The proposed algorithm had an average improvement rate of 51% in total resource utilization, 50% in generation distance, and 16.5% in execution time when compared with MOPSO.

2) Second experiment: Here, the proposed method's performance is compared with the Artificial Fish Swarm Optimization algorithm (AFSO) [38]. This experiment examines the proposed method's time, cost, and energy efficiency. Fig. 8 illustrates the performance of the proposed method. As the number of tasks increases, the execution time will also increase. Fig. 9 illustrates an analysis of the performance in terms of cost. System performance is affected by the maximum cost. Fig. 10 also illustrates performance in terms of energy consumption. An increase in energy means an increase in cost as well. The proposed algorithm improved the total execution time by 22%, cost by 9%, and energy consumption by 21% compared with AFSO.

TABLE II. Statistical Comparison for Small-Scale

Parameters	NSGA-II	MOPSO	PSO-fuzzy
Total earnings of the provider	0.0852	0.01123	0.1952
Total incomes of the users	0.2962	0.3740	0.5124
Total resource utilization	5.5175	7.0364	12.212
Generation distance	0.0385	0.022	0.00931
The distance	0.0545	0.0928	0.03012
Execution time (sec)	64.51	18.74	11.12

TABLE III. STATISTICAL COMPARISON FOR MIDDLE-SCALE

Parameters	NSGA-II	MOPSO	PSO-fuzzy
Total earnings of the provider	0.1786	0.2340	0.561
Total incomes of the users	0.5908	0.7710	2.1325
Total resource utilization	15.7246	19.2885	42.124
Generation distance	0.0067	0.0043	0.0012
The distance	0.0197	0.0394	0.00124
Execution time (sec)	68.187565	22.224956	15.324

TABLE IV. STATISTICAL COMPARISON FOR LARGE-SCALE

Parameters	NSGA-II	MOPSO	PSO-fuzzy
Total earnings of the provider	1.0241	1.3480	2.125
Total incomes of the users	1.3848	2.0113	3.163
Total resource utilization	31.2672	42.2504	65.260
Generation distance	0.0124	0.0062	0.00325
The distance	0.0421	0.0784	0.0231
Execution time (sec)	87.255300	28.799435	18.215

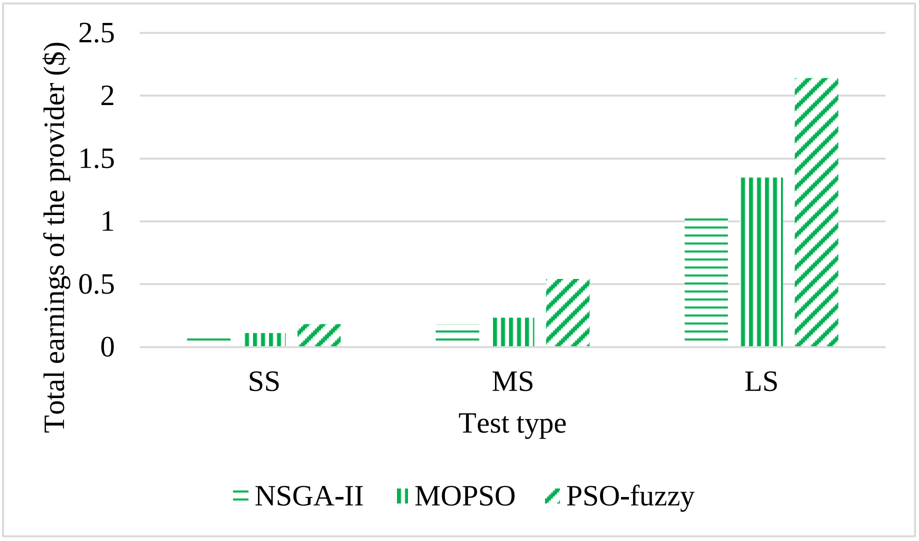

Fig. 2. Total earnings of the provider comparison.

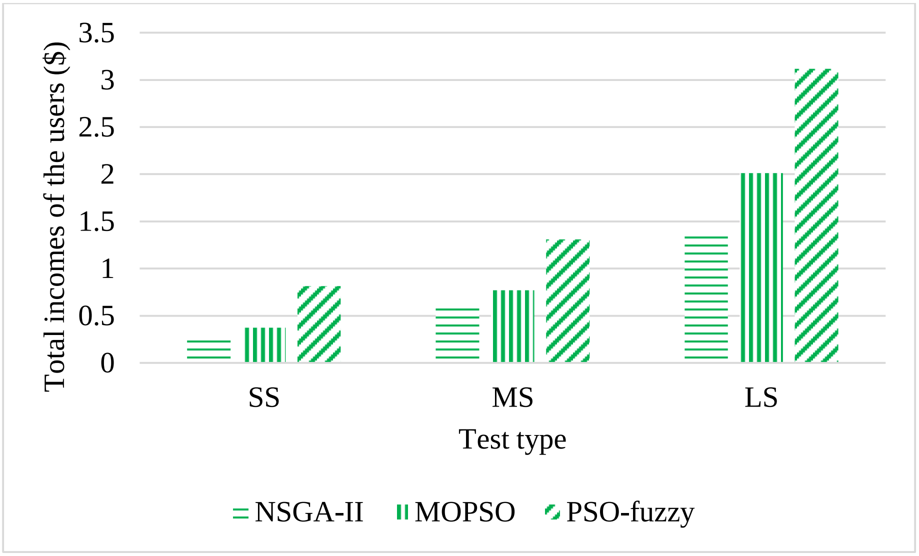

Fig. 3. Total incomes of the user's comparison.

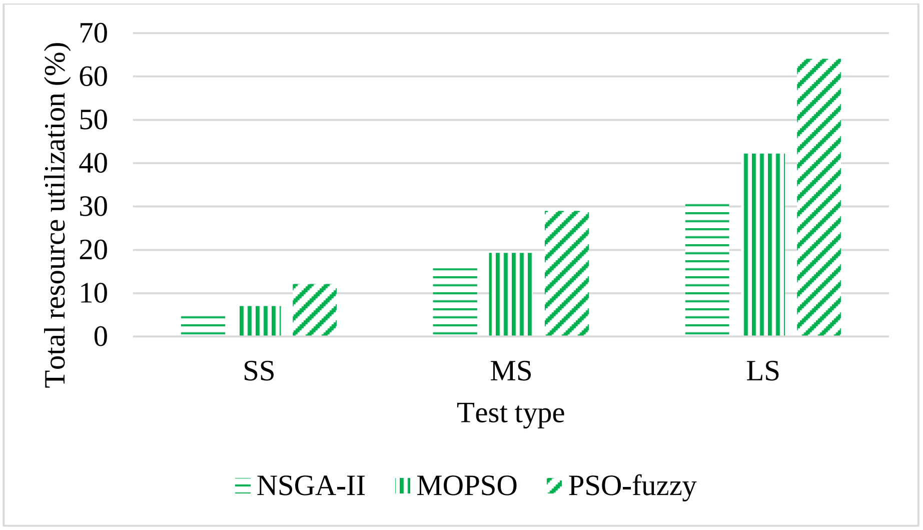

Fig. 4. Total resource utilization comparison.

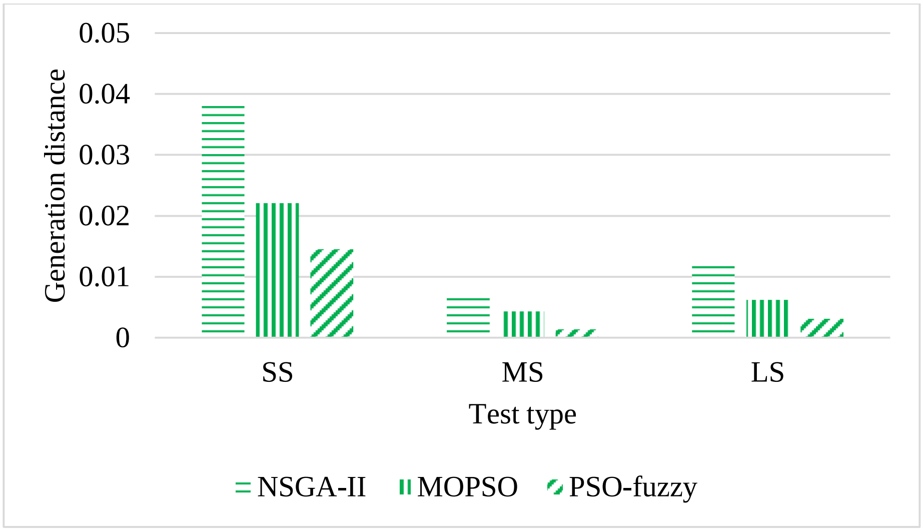

Fig. 5. Generation distance comparison.

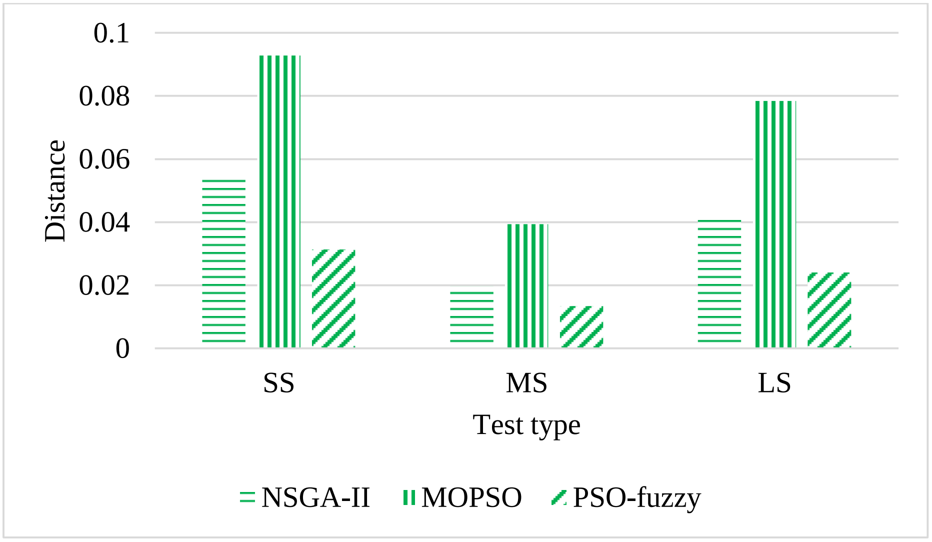

Fig. 6. The distance comparison.

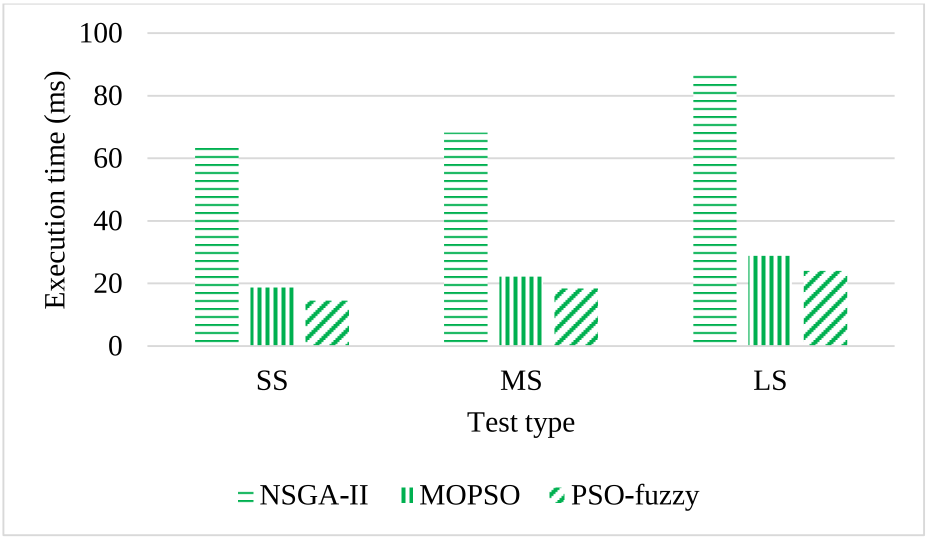

Fig. 7. Execution time comparison.

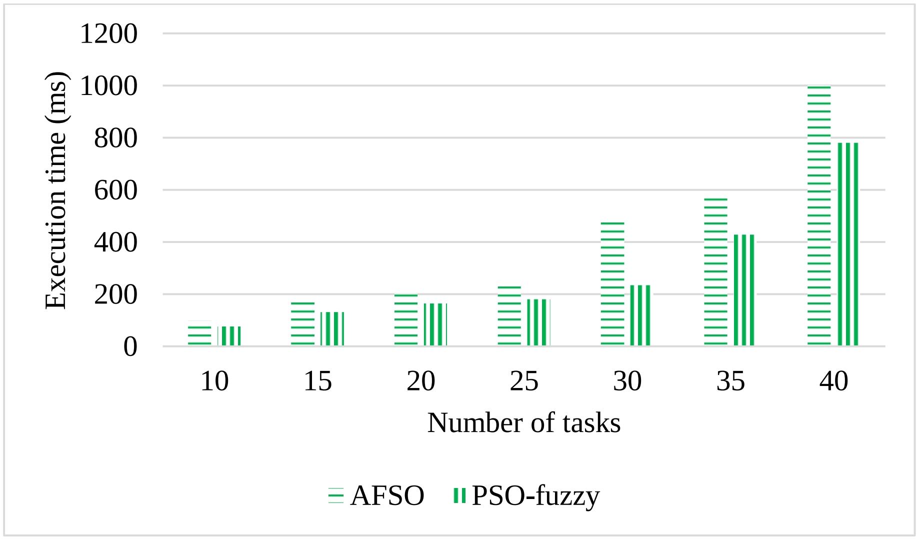

Fig. 8. Execution time comparison.

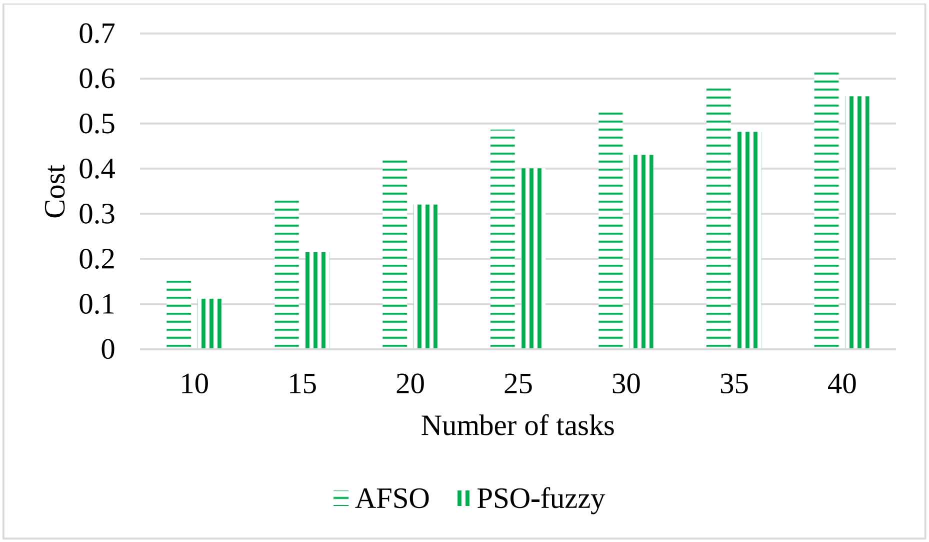

Fig. 9. Cost comparison.

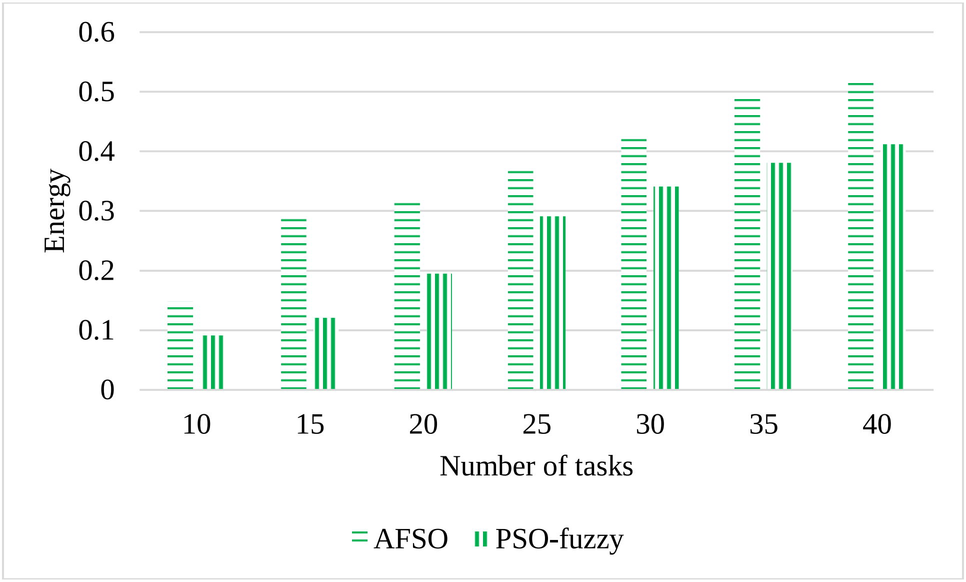

Fig. 10. Energy consumption comparison.

Performance measures expressed proportionally through specific mathematical models encompass key indicators such as total provider earnings, user incomes, resource utilization, generation distance, and execution time. In the first experiment, the PSO-fuzzy algorithm's performance is benchmarked against NSGA-II and MOPSO algorithms, showcasing superior results across various metrics. Notably, the proposed algorithm exhibits an average improvement of 51% in resource utilization, a 50% enhancement in generation distance, and a substantial 16.5% reduction in execution time compared to MOPSO. The PSO-fuzzy methodology is pitted against the Artificial Fish Swarm Optimization (AFSO) algorithm in the second experiment. The results highlight a marked improvement in execution time by 22%, cost efficiency by 9%, and a notable 21% reduction in energy consumption when compared to AFSO. These findings underscore the robustness and efficiency of the PSO-fuzzy algorithm in optimizing resource allocation and cost management across varying scales and scenarios within cloud computing environments.

V. Conclusions

This paper proposed an optimal resource allocation method combining the PSO algorithm and fuzzy logic system based on the presented time and cost models in the cloud computing environment. The time model includes receiving, processing, and waiting times. Costs associated with processing and receiving are included in the cost model. The PSO algorithm was applied to the cloud environment for optimal resource allocation. The fuzzy logic system was used to evaluate the time and cost models. The proposed algorithm's efficacy was clearly demonstrated through a series of meticulously designed experiments. In the initial experiment, comparative analysis against established algorithms, namely NSGA-II and MOPSO, revealed the superiority of our method concerning providers' total income, users' total revenue, and resource utilization. Subsequently, the second experiment showcased the algorithm's superior performance in execution time, cost-effectiveness, and energy consumption when juxtaposed with the AFSO algorithm. These results unequivocally establish the proposed algorithm's prowess, emphasizing its effectiveness in both performance and efficiency metrics. The outcomes affirm the superiority of our algorithm for scheduling within cloud computing systems, surpassing existing methodologies. The success of this approach not only underscores its potential in addressing the resource allocation challenge but also signifies a significant stride toward optimizing cloud computing operations. However, while these results are promising, future work should delve into further validation across a more extensive range of scenarios and consider real-world implementations to solidify the algorithm's robustness and applicability in diverse cloud environments.

Acknowledgments

University level Scientific Research Fund Project of Xi'an Aeronautical Institute (2022KY1204).

References

[1] B. Pourghebleh, A. A. Anvigh, A. R. Ramtin, and B. Mohammadi, "The importance of nature-inspired meta-heuristic algorithms for solving virtual machine consolidation problem in cloud environments," Cluster Computing, pp. 1-24, 2021.

[2] M. Mohseni, F. Amirghafouri, and B. Pourghebleh, "CEDAR: A cluster-based energy-aware data aggregation routing protocol in the internet of things using capuchin search algorithm and fuzzy logic," Peer-to-Peer Networking and Applications, pp. 1-21, 2022.

[3] B. Pourghebleh, N. Hekmati, Z. Davoudnia, and M. Sadeghi, "A roadmap towards energy-efficient data fusion methods in the Internet of Things," Concurrency and Computation: Practice and Experience, p. e6959, 2022.

[4] S. Bharany et al., "Energy efficient fault tolerance techniques in green cloud computing: A systematic survey and taxonomy," Sustainable Energy Technologies and Assessments, vol. 53, p. 102613, 2022.

[5] H. Feng, Y. Deng, and J. Li, "A global-energy-aware virtual machine placement strategy for cloud data centers," Journal of Systems Architecture, vol. 116, p. 102048, 2021.

[6] G. J. Ibrahim, T. A. Rashid, and M. O. Akinsolu, "An energy efficient service composition mechanism using a hybrid meta-heuristic algorithm in a mobile cloud environment," Journal of parallel and distributed computing, vol. 143, pp. 77-87, 2020.

[7] K. Karthikeyan et al., "Energy consumption analysis of Virtual Machine migration in cloud using hybrid swarm optimization (ABC–BA)," The Journal of Supercomputing, vol. 76, no. 5, pp. 3374-3390, 2020.

[8] M. Yavari, A. G. Rahbar, and M. H. Fathi, "Temperature and energy-aware consolidation algorithms in cloud computing," Journal of Cloud Computing, vol. 8, no. 1, pp. 1-16, 2019.

[9] W. Shu, K. Cai, and N. N. Xiong, "Research on strong agile response task scheduling optimization enhancement with optimal resource usage in green cloud computing," Future Generation Computer Systems, vol. 124, pp. 12-20, 2021.

[10] V. Hayyolalam, B. Pourghebleh, M. R. Chehrehzad, and A. A. Pourhaji Kazem, "Single-objective service composition methods in cloud manufacturing systems: Recent techniques, classification, and future trends," Concurrency and Computation: Practice and Experience, vol. 34, no. 5, p. e6698, 2022.

[11] P. Behrouz, H. Vahideh, and A. A. Aghaei, "Service discovery in the Internet of Things: review of current trends and research challenges," Wireless Networks, vol. 26, no. 7, pp. 5371-5391, 2020.

[12] T. Baker, M. Asim, H. Tawfik, B. Aldawsari, and R. Buyya, "An energy-aware service composition algorithm for multiple cloud-based IoT applications," Journal of Network and Computer Applications, vol. 89, pp. 96-108, 2017.

[13] H. Vahideh, P. Behrouz, P. K. A. Asghar, and A. Ghaffari, "Exploring the state-of-the-art service composition approaches in cloud manufacturing systems to enhance upcoming techniques," The International Journal of Advanced Manufacturing Technology, vol. 105, no. 1-4, pp. 471-498, 2019.

[14] R. Singh et al., "Analysis of Network Slicing for Management of 5G Networks Using Machine Learning Techniques," Wireless Communications and Mobile Computing, vol. 2022, 2022.

[15] B. Pourghebleh and V. Hayyolalam, "A comprehensive and systematic review of the load balancing mechanisms in the Internet of Things," Cluster Computing, pp. 1-21, 2019.

[16] P. He, N. Almasifar, A. Mehbodniya, D. Javaheri, and J. L. Webber, "Towards green smart cities using Internet of Things and optimization algorithms: A systematic and bibliometric review," Sustainable Computing: Informatics and Systems, vol. 36, p. 100822, 2022, doi: https://doi.org/10.1016/j.suscom.2022.100822.

[17] S. N. H. Bukhari, J. Webber, and A. Mehbodniya, "Decision tree based ensemble machine learning model for the prediction of Zika virus T-cell epitopes as potential vaccine candidates," Scientific Reports, vol. 12, no. 1, p. 7810, 2022.

[18] F. Kamalov, B. Pourghebleh, M. Gheisari, Y. Liu, and S. Moussa, "Internet of Medical Things Privacy and Security: Challenges, Solutions, and Future Trends from a New Perspective," Sustainability, vol. 15, no. 4, p. 3317, 2023.

[19] B. M. Jafari, M. Zhao, and A. Jafari, "Rumi: An Intelligent Agent Enhancing Learning Management Systems Using Machine Learning Techniques," Journal of Software Engineering and Applications, vol. 15, no. 9, pp. 325-343, 2022.

[20] J. Webber, A. Mehbodniya, Y. Hou, K. Yano, and T. Kumagai, "Study on idle slot availability prediction for WLAN using a probabilistic neural network," in 2017 23rd Asia-Pacific Conference on Communications (APCC), 2017: IEEE, pp. 1-6.

[21] S. Aghakhani, A. Larijani, F. Sadeghi, D. Martín, and A. A. Shahrakht, "A Novel Hybrid Artificial Bee Colony-Based Deep Convolutional Neural Network to Improve the Detection Performance of Backscatter Communication Systems," Electronics, vol. 12, no. 10, p. 2263, 2023.

[22] M. Sadi et al., "Special Session: On the Reliability of Conventional and Quantum Neural Network Hardware," in 2022 IEEE 40th VLSI Test Symposium (VTS), 2022: IEEE, pp. 1-12.

[23] Z. Wang and X. Su, "Dynamically hierarchical resource-allocation algorithm in cloud computing environment," The Journal of Supercomputing, vol. 71, no. 7, pp. 2748-2766, 2015.

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

[25] S. Mousavi, A. Mosavi, and A. R. Varkonyi-Koczy, "A load balancing algorithm for resource allocation in cloud computing," in International conference on global research and education, 2017: Springer, pp. 289-296.

[26] M. Ficco, C. Esposito, F. Palmieri, and A. Castiglione, "A coral-reefs and game theory-based approach for optimizing elastic cloud resource allocation," Future Generation Computer Systems, vol. 78, pp. 343-352, 2018.

[27] X. Chen, H. Wang, Y. Ma, X. Zheng, and L. Guo, "Self-adaptive resource allocation for cloud-based software services based on iterative QoS prediction model," Future Generation Computer Systems, vol. 105, pp. 287-296, 2020.

[28] R. Singhal and A. Singhal, "A feedback-based combinatorial fair economical double auction resource allocation model for cloud computing," Future Generation Computer Systems, vol. 115, pp. 780-797, 2021.

[29] A. Thakur and M. S. Goraya, "RAFL: A hybrid metaheuristic based resource allocation framework for load balancing in cloud computing environment," Simulation Modelling Practice and Theory, vol. 116, p. 102485, 2022.

[30] L. Li, Y.-a. LIU, K.-m. LIU, and Y. Ming, "Pricing in combinatorial double auction-based grid allocation model," The Journal of China Universities of Posts and Telecommunications, vol. 16, no. 3, pp. 59-65, 2009.

[31] P. Samimi, Y. Teimouri, and M. Mukhtar, "A combinatorial double auction resource allocation model in cloud computing," Information Sciences, 2014.

[32] C. R. Raquel and P. C. Naval Jr, "An effective use of crowding distance in multiobjective particle swarm optimization," in Proceedings of the 7th Annual conference on Genetic and Evolutionary Computation, 2005: ACM, pp. 257-264.

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

[34] K. Miettinen, "Nonlinear Multiobjective Optimization, volume 12 of International Series in Operations Research and Management Science," ed: Kluwer Academic Publishers, Dordrecht, 1999.

[35] L. Wang and C. Singh, "Environmental/economic power dispatch using a fuzzified multi-objective particle swarm optimization algorithm," Electric Power Systems Research, vol. 77, no. 12, pp. 1654-1664, 2007.

[36] L. Xu, Z. Zeng, and X. Ye, "Multi-objective optimization based virtual resource allocation strategy for cloud computing," in Computer and Information Science (ICIS), 2012 IEEE/ACIS 11th International Conference on, 2012: IEEE, pp. 56-61.

[37] M. Feng, X. Wang, Y. Zhang, and J. Li, "Multi-objective particle swarm optimization for resource allocation in cloud computing," in Cloud Computing and Intelligent Systems (CCIS), 2012 IEEE 2nd International Conference on, 2012, vol. 3: IEEE, pp. 1161-1165.

[38] P. Albert and M. Nanjappan, "An efficient kernel FCM and artificial fish swarm optimization-based optimal resource allocation in cloud," Journal of Circuits, Systems and Computers, vol. 29, no. 16, p. 2050253, 2020.