International Journal of Electrical and Computer Engineering (IJECE)
Vol. 13, No. 4, August 2023, pp. 4743~4751
ISSN: 2088-8708, DOI: 10.11591/ijece.v13i4.pp4743-4751

Virtual machine placement in cloud using artificial bee colony and imperialist competitive algorithm

Seyyed-Mohammad Javadi-Moghaddam[1], Zahra Dehghani[2]
[1]Department of Computer, Faculty of Engineering, Bozorgmehr University of Qaenat, Qaen, Iran
[2]Department of Computer Engineering, Azad University of Birjand, Birjand, Iran

Article Info

Article history:

Received Sep 14, 2022
Revised Dec 8, 2022
Accepted Dec 11, 2022

Keywords:

Artificial bee colony algorithm
Cloud computing
Imperialist competitive algorithm
Virtual machine placement
Virtualization

ABSTRACT

Increasing resource efficiency and reducing energy consumption are significant challenges in cloud environments. Placing virtual machines is essential in improving cloud systems' performance. This paper presents a hybrid method using the artificial bee colony and imperialist competitive algorithm to reduce provider costs and decrease client expenditure. Implementation of the proposed plan in the CloudSim simulation environment indicates the proposed method performs better than the Monarch butterfly optimization and salp swarm algorithms regarding energy consumption and resource usage. Moreover, average central processing unit (CPU) and random-access memory (RAM) usage and the number of host shutdowns show better results for the proposed model.

Corresponding Author:

Seyyed-Mohammad Javadi-Moghaddam
Department of Computer, Faculty of Engineering, Bozorgmehr University of Qaenat
Abolmafakher Street, Qaen, Iran
Email: smjavadim@buqaen.ac.ir

1. INTRODUCTION

A particular distributed computing model that introduces applicable models for providing remotely measurable resources and commercial computing network commands, parallel computing, and distributed computing is known as cloud computing [1], [2]. The demand for users for cloud resources is increasing while cloud computing technology is developing [3]. The increase in the resources provided in the cloud causes an increase in essential parameters such as the amount of energy consumption, which can cause new challenges such as limited energy resources and global warming.

Technology virtualization is essential to improve the efficient use of clouds and resources, which has many advantages, including server integration, migration, isolation accessibility, flexible expansion, and low management overhead. Virtualization offers many energy management solutions in cloud computing [4], [5]. The virtualization process can integrate several virtual machines (VM) at the data center level by placing some virtual servers on a physical machine (PM) and then shutting down idle PMs to optimize energy consumption [6], [7]. The virtual machines placement (VMP) places the VMs on the physical machines so that the benefit of the cloud providers is maximized [8]. Each PM has limited resources to host several virtual machines, and each has its resource requests. The purpose of the VMP problem is to place the maximum possible number of VMs on the PMs while not exceeding the physical range of each physical machine and resulting in maximum profit for the provider [9].

Various methods have been proposed for locating machines in cloud computing [10], [11]. However, they are still facing the challenge of increasing the energy consumption related to data centers, raising costs, and the high execution time of tasks [12]. Huang *et al.* [13] proposed a suitable destination for selecting virtual machines by classifying workflows and integrating physical machines. Alharbi *et al.* [14] presented a linear formula for the problem of dynamic integration of VMs and limited the mapping of some VMs to private servers with particular features. They also set a threshold for the number of migrations that do not exceed that limit. Furthermore, they considered cost parameters in their formulation so that the VM is positioned to reduce the overall cost of the system. Han *et al.* [15] described an efficient energy resource management system for virtual cloud centers, which reduces operating costs and satisfies the quality of service as much as possible. Sharma *et al.* [16] proposed an approach for the VMP problem to improve cloud service quality. This work tries to use fewer hosts to reduce the overhead. Braiki and Youssef [17] have introduced a self-adaptive solution for the VMP algorithm using particle swarm optimization. Its self-adaptive feature allows the algorithm to decide when each virtual machine is placed on which physical host.

Some researchers have focused on only energy saving [18], [19]. Kaaouache and Bouamama [20] proposed an approach to solving the problem using genetic algorithms. This approach attempts to optimize the energy consumption in servers and communications with a suitable objective function. Reddy *et al.* [21] suggested a stabilization technique to reduce energy consumption in a cloud system. The authors have empirically described server power consumption as a central processing unit (CPU) performance and disk usage model. Dhaya *et al.* [22] focused on energy management methods of resources that a cloud provider can process in a virtual data center. A primary tool is live VM migration. The ability of VMs to migrate between low-overhead physical hosts provides the necessary flexibility for the resource provider to dynamically reallocate virtual machines according to current resource requirements and allocation policies. This method shuts idle physical nodes down to minimize energy consumption and optimizes placement.

Several articles have considered the service level agreement (SLA). Alharbi *et al.* [23], try to minimize SLA violations number using the ant colony algorithm. In this multipurpose algorithm, the behavior of each ant is a solution for assigning the virtual machine to the server. Solutions created by specific functions are then evaluated. If this solution is on the list of the best solutions, its pheromone level will increase. Otherwise, it will decrease. Finally, the law of updating the pheromone is applied to achieve a suitable solution for virtual machine placement. Dynamic placement algorithms can improve the integration of servers by reducing the violation SLA rate. Ranjbari and Torkestani [24] presented a dynamic placement method of virtual machines to manage breaches of service level parameters. This solution aims to decrease the needed capacity and the SLA violation rate. The management algorithm presented in this paper reduces the physical ability needed to support a specific service level violation rate. This algorithm aims to achieve the minimum cost of the data center implementation based on capacity overflow, low resource consumption, and load overflow, which leads to poor performance and breach of service level agreement.

Most previous works have tried to decrease the energy consumption of cloud sources by considering the energy crisis and global warming issues. Solutions have been presented by reducing the number of hosts and using virtualization to increase the productivity of physical resources. The main challenge in placing a virtual machine is choosing the suitable virtual machine and physical machine. Meta-heuristic methods have shown acceptable performance in the search process and are a solution to this problem. This paper attempts to overcome this challenge using the artificial bee colony algorithm (ABCA) and the imperial competitive algorithm (ICA). This paper performs the VMP in two stages: select a virtual machine to transfer to physical servers and select the most appropriate physical server. The proposed model uses the ABCA to choose the best VM and the imperial competitive algorithm to select the best physical server.

The continuation of the article includes the following sections. Section 2 explains the proposed method in detail. The evaluation results and discussion have been expressed in section 3. The conclusion is presented in section 4.

2. METHOD

2.1. Artificial bee colony algorithm (ABCA)

Akay *et al.* [25] introduced the ABCA algorithm to optimize mathematical functions. Each solution expresses an area of the potential food area. Furthermore, the quality is the quality of the food source. Artificial bees try to find food resources by searching. ABCA uses three types of agents: employed bees (EB), onlooker bees (OB), and scouts. EBs are related to the current answers of the algorithm. At each step of the algorithm, an EB tries to provide a solution, improve it by a local search step, and find the OB bees for the current location. OBs select higher quality locations from enhanced locations. If OB could find a better location, EB would update its location; otherwise, it remains in its current location. Furthermore, an EB bee will leave its place if it cannot improve its location in a certain number of steps. If an EB leaves its place, it becomes a scout and randomly selects a new place in the search space.

2.2. Imperialist competitive algorithm (ICA)

The imperial competitive algorithm (ICA) [26] is a proposed optimization method inspired by imperialist competition's mathematical modeling. This algorithm is an optimization strategy mimicking human behavior in a social-political evolution. It starts with a random number of initial populations known in a country. Imperialists united these colonies with the policy of attraction regarding their power. The empire's power is calculated according to the imperialist state and its colonies. Mathematically, the dependence is defined according to the total imperial state power and the average of its colonies' power. Imperialism started with the forming of primitive empires. Any unsuccessful empire in the colonial competition that decreased its power will be removed from the colonial competition. Thus, the empire's stability is related to its power to absorb and subjugate the colonies. Therefore, the weaker empires gradually are removed during the imperialist rivalries because of the enormous empire's power. Empires must increase their power using developing their colonies. Thus, the power of the colonies will be closed to the empires over time, and convergence will appear. The algorithm’s end is when an empire’s location is very close to the imperialist country.

2.2.1. Absorption policy

In the absorption phase, the imperialist countries build infrastructures (transportation infrastructure, university establishment, and to increase their development. This phase of the imperialist optimization algorithm models the movement of the colonies. Figure 1 depicts this movement. Colony country has moved *X* units in the direction of the line connecting the colony to the Imperialist and has changed the position in the colony. The distance between the imperialist and the colony is denoted by *d*. *A* suitable distribution, like uniform distribution, randomly generates the value of *X*.

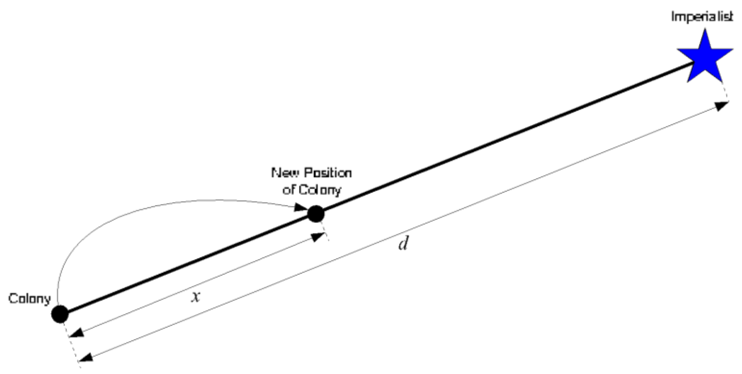

Figure 1. The move of the colonies toward the imperialist [27]

2.2.2. Absorption policy exchanging the colony and imperialist position

Some colonies can take a more suitable situation than the imperialists as the colonies move toward the colonial country. This situation can be caused by changing the place of the imperialist country and the colonized country. Then, the algorithm continues on the new position of the colonizing country, and the colonized countries move to a new place Figure 2.

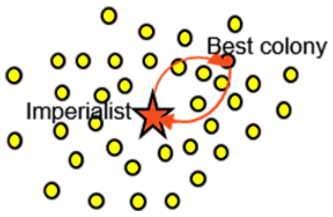

Figure 2. Exchanging the colony and imperialist position [28]

2.3. Proposed algorithm

The proposed method consists of two parts: selecting a virtual machine to transfer to physical servers and selecting the most appropriate physical server. It uses the bee colony method to choose the best virtual machine. Furthermore, the imperialist competitive algorithm tries to select the best physical server.

2.3.1. Selecting a VM with the ABCA

The ABCA operates using the population criterion, like the exploratory behavior of bees. The ABCA includes three groups EB, OB, and scouts. Food resource's location and an available solution to choose the best virtual machine for placement in machines are considered. The usefulness of a food source indicates the degree to which it fits the solution to the problem, which is the choice of the best VM in this work. The amount of EB equals the amount of OB and the number of food sources. Scouts replace any food source with a new one when it is no further improved in a given cycle.

At first, the initial population is the number of solutions (Ns) that are randomly generated. Ns is equal to EB and is the food sources number. Furthermore, Ns is the population size that each solution xi $(i = 1...Ns)$ is a multidimensional vector. The bees then conduct a rotating search according to specific rules. In this study, each EB considers several VM as new candidates as a food source to update the possible solutions. Selection is made using the previous neighborhood of VMs. As shown in (1) generates candidate solution SV_i (solution i) by previous solutions as (1),

$$SV_{ij} = x_{ij} + \Phi_{ij}\,(x_{ij} + x_{kj}) \tag{1}$$

where $k \in \{1...Ns\}$ and $j \in \{1...n\}$ are indices randomly generated by a random distribution in the interval [-1,1]. The method compares candidate solutions with the old ones, and if the new sources are of better quality or equal to the old food sources, the old food sources will be replaced with new food sources. Otherwise, the old food sources will remain without any change. After returning EB to the hive, it shares information about the food sources information with the OB. Then, each OB chooses a food source regarding its fitness value. As shown in (2) calculates the probability p_i of each selection:

$$p_i = \frac{f_i}{\sum_{j=1}^{Ns} f_j} \tag{2}$$

where f_i expresses the fitness value of solution *i* which depends on the value of the fitness functions, the degree of fitness can increase the possibility of choosing a food source for each OB. After selecting the food source, each OB can find a new food source candidate from their neighbors. A greedy solution selects a food source regarding its fitness. If the position of a food source (optimal virtual machine) is not changed by a finite cycle, it is considered a solution. Scouts play an essential role in replacing food sources. Scouts replace any old food source with a new food source. A cycle limitation is a predefined number that the user defines. If source x_i wants to be replaced by scouts (3) is used.

$$x_{ij} = x_{jmin} + rand[0.1](x_{jmax} - x_{jmin}) \tag{3}$$

where x_{jmax} and x_{jmin} are the bounds of x_{ij}, a random number between [0.1] is generated by a uniform distribution. This process is repeated until the function reaches its threshold. Therefore, the bee colony algorithm determines the best VM placed in the PM. One of the essential operators used in the bee colony algorithm is the fitting operator, which with the help of other operators, produces the most optimal solutions to select the desired virtual machine. The process of transmitting messages in the cloud computing network is that the nodes in each physical machine send their message to the host, and the host also sends its messages to the data center. As shown in (4) transfers a message from a VM to a physical one after placement.

$$En = \sum_{i=1}^{n} EnVM_i + \big((n-1) * EnPH\big) + VM_M + EPR \tag{4}$$

where En refers to the energy used to transmit a message from a virtual machine to a physical machine, n is the number of virtual machines of the physical machine, and $EnVM_i$ is the energy consumed to send a message from the VM to the PM. The $EnPH$ parameter is the amount of energy the physical machine requires to receive the message. EPR indicates a physical machine's energy to process and aggregate data. Parameter VM_M expresses the energy needed to process requests by virtual machines.

Two critical factors play a role in calculating the fitness function: the amount of data overhead and the energy consumed to send information from the virtual machine to the data center. These two factors are very influential in choosing the best virtual machine. Therefore, the principal purpose of the ACBA is to minimize these factors to choose the best VM for placement in the physical machine. Furthermore, we can reduce the VMs number in the fitting function, which, like reducing the amount of data overhead, can affect the choice of virtual machines because the host consumes more energy than others. Consequently, the fitness function consists of a mapping from a virtual machine and a physical machine to bits of zero, and one is calculated as (5).

$$f_i = \frac{1}{En} + (OverF - SumEn) + (N - PHM) \tag{5}$$

where N is the total number of virtual machines in the cloud computing network, PHM determines the PMs number, and $OverF$ determines the total data overhead of all VMs. Finally, SumEn specifies the total energy required to process the virtual machines.

Less energy consumption and less data overload will cause more amount of fitness. The proposed bee colony algorithm tries to find a suitable solution by increasing the fitting value. Therefore, all the mentioned operators are used to improve the positioning and calculate the optimal response. After choosing the best VMs for placement in the physical machine, the method chooses the best.

2.3.2. Selecting a PM with the ICA

The parameters of the ICA include the following: i) Initial population: A number of these countries are considered primitive populations (N_c). The country describes physical machines; ii) Selection: N_I is to be chosen as imperialist from the best members of this population (countries with a minor cost function); iii) Remaining: N_{col} expresses the remaining countries which belong to the system. For example, the physical machines belong to the data center; and iv) Colonies division: Each imperialist is assigned several colonies, the number commensurate with its power. The normalization cost is considered according to the cost of all imperialists, as (6).

$$C_n = max_i\{c_i\} - c_n \tag{6}$$

where c_n is the cost of the i^{th} imperialist. $max_i\{c_i\}$ is the maximum cost of imperialists, and C_n is the normalization cost. Stronger imperialists will cost more, and weaker imperialists will have fewer normalization costs. As shown in (7) calculates the relative normalized power of each imperialist (P_n).

$$P_n = \left|\frac{C_n}{\sum_{i=1}^{N_I} c_i}\right| \tag{7}$$

The proportion of colonies ruled by an imperialist is the normalized power of an imperialist. Thus, the initial number of colonies of an imperialist would be equal to (8).

$$N_{init}.Cl = round(P_n.N_{cl}) \tag{8}$$

where $N_{init}.Cl$ is the initial number of colonies of a system. N_{cl} is the number of colonized countries of early countries (physical machines). *round* expresses the closest integer to a decimal number. Each system randomly selects the original colonial countries and assigns them to the n^{th} imperialist. The colonial competition algorithm begins after calculating the initial state of all systems. The evolutionary process continues until a stop condition is met.

2.3.3. Goals of the proposed algorithm

This research pursues two general goals for locating VM in data centers of the cloud system. First, minimizing overall traffic by placing virtual machines in cloud data centers can reset the overall traffic layout between VMs and allow VMs to be on the same physical machine or switch with the same amount of traffic. Second, reducing the maximum link utilization (MLU) enables network traffic to be allocated fairly and avoids congestion. Moreover, the above criteria consider the quality of service for applications sensitive to latency.

a. Optimizing overall traffic

Two matrices are considered to optimize the overall traffic: the traffic matrix $A = (a_{ij})_{nv\times nv}$ and the transmission cost matrix $B = (b_{hp})_{np\times np}$. a_{ij} provides traffic between VM_i and VM_j, b_{hp} provides PM_h and PM_p transmission costs, *nv* is the number of VMs and *np* is the number of PMs. The transmission cost expresses the switches number that traffic between PM. The greater the number of switches results in the higher the transmission cost. The goal is to find a mapping function π(i) to encounter VM_i located on the PM so that a VM only can be on one PM, but one PM can be a host for multiple VMs. As shown in (9) calculates the objective function.

$$Minimize\ T_{cost} = \sum_{i,j=1}^{n_v} a_{ij} b_{\pi(i)\pi(j)} + \sum_{i=1}^{n_v} e_i g_{\pi(i)} \tag{9}$$

where e_i is the traffic between VM_i and the external communication of the data center, $g_{\pi(i)}$ is the transmission cost from the PM of VM_i to the external switch.

b. MLU optimization

The goal of optimization is to minimize the MLU value, which is calculated as (10).

$$Min(MLU) = Max(\frac{\sum_{(s,t)} Tr_{ij}^{st}}{CP_{ij}}) \quad (10)$$

where Tr_{ij}^{st} is traffic flow from node s to node t traversing link from node i to node j. CP_{ij} is the network capacity of a path from s to t.

3. RESULTS AND DISCUSSION

3.1. Simulation Environment

This study has used CloudSim Simulator [29], [30] to evaluate the proposed method. The operating system was a 32-bit Windows 7. Features of the simulation hardware of the computer include 4 GB RAM, usable memory is 3.06 GB, Intel processor (Q 720 @ 1.60GHz 1.60 GHz – (CoreTM) i7 CPU).

3.2. Parameters

This section expresses the characteristics of the simulation environment. Table 1 shows the essential parameters and fundamental arguments for simulating the proposed method. Furthermore, Table 2 describes the parameters of the optimization algorithm.

Table 1. The proposed system simulation input parameters and arguments

Parameters and arguments	Value
Number of virtual machines	256
Number of hosts	64
Minimum number of processors for each virtual machine	1
Maximum number of processors for each virtual machine	8
Minimum amount of RAM for each virtual machine	1024
Maximum amount of RAM for each virtual machine	8192
Minimum amount of disk space for each virtual machine (GB)	1 GB
Maximum amount of disk space for each virtual machine (GB)	8 GB
Minimum number of processors for each host	1
Maximum number of processors for each host	32
Minimum amount of RAM for each host	1024
Maximum amount of RAM for each host	32768
Minimum amount of disk space per host (GB)	1 GB
Maximum amount of disk space per host (GB)	300 GB

Table 2. Optimization algorithm parameters

Initial population	265
Number of particles	64
Number of iterations of the algorithm	100

3.3. Results

For evaluation, the proposed method, Monarch butterfly optimization, and salp swarm algorithms have been compared based on energy, CPU, and RAM consumption criteria. Figure 3 shows the three algorithms' energy consumption with working loads from 100 to 400 VM. The proposed method with low energy consumption reduces the cost for cloud service providers. High energy consumption related to data centers results in increasing costs for cloud computing service providers. Furthermore, it causes increasing CO_2 emissions. Figure 3 shows that the proposed method consumed less energy than the compared methods. This energy consumption reduction states that using two meta-heuristic algorithms in choosing the appropriate virtual and physical machine has reduced the number of hosts and ultimately reduced energy consumption.

Figure 4 compares the average CPU and RAM usage of the methods. Figure 4 shows that CPU usage is approximately 95% for the proposed solution, while 90% in the Monarch butterfly optimization and 84% in the salp swarm algorithm. RAM usage is about 93% for the proposed algorithm, approximately 85% for monarch butterfly optimization, and 80% for the salp swarm algorithm. Furthermore, it results in the proposed algorithm increasing resource usage significantly.

Figure 5 depicts the host shutdowns number of the evaluated three methods. Furthermore, it expresses that the total shutdown of the host in the proposed solution compared to other solutions is much higher.

Consequently, the proposed solution saves a significant percentage of all physical machines. Therefore, the proposed algorithm solution effectively improves the rate of resource usage by increasing the host capacity and shutting down more physical machines.

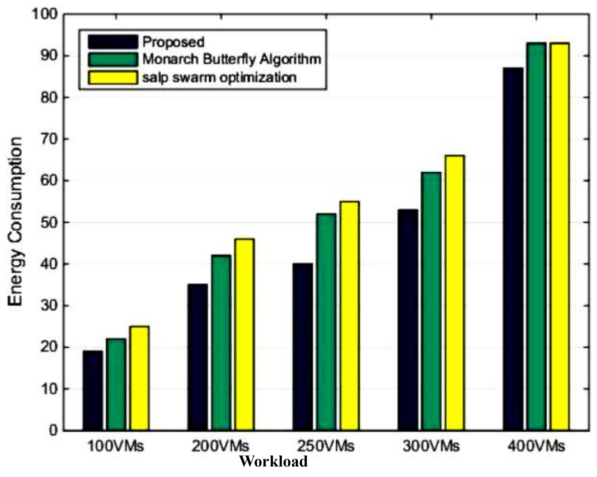

Figure 3. Energy consumption with workloads from 100 VM to 400 VM by the proposed algorithm, Monarch butterfly optimization, and salp swarm algorithms

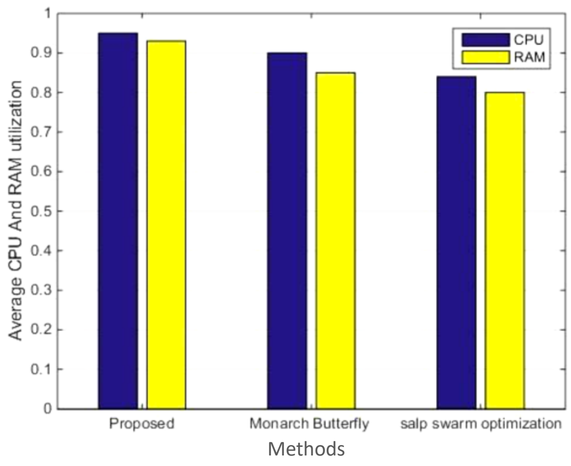

Figure 4. Average CPU and RAM usage for the proposed and compared methods

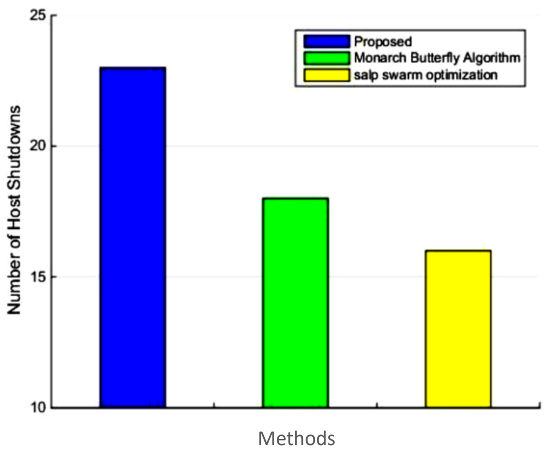

Figure 5. Comparison of the number of hosts shut down in the proposed method and methods compared

Finally, by reducing the number of hosts, the proposed method can reduce energy consumption. Consequently, it reduces the cost of cloud service providers. This improvement allows cloud services to be offered more reasonably priced for users.

4. CONCLUSION

An essential goal of cloud environments is to decrease resource provider and client costs. The proposed model uses the bee colony algorithm to select the best virtual machine and the imperial competition algorithm to choose the best physical server. The evaluation results depicted that the proposed algorithm has an average CPU usage of approximately 95% and average RAM usage of about 93%, which is more than both Monarch butterfly optimization and salp swarm algorithms. Furthermore, the total shutdown of the host in the proposed solution compared to other solutions is much higher. Therefore, the proposed solution saves a significant percentage of usage of all physical machines.

REFERENCES

[1] D. A. Shafiq, N. Z. Jhanjhi, and A. Abdullah, "Load balancing techniques in cloud computing environment: A review," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 7, pp. 3910–3933, Jul. 2022, doi: 10.1016/j.jksuci.2021.02.007.

[2] M. Kumar and S. C. Sharma, "PSO-based novel resource scheduling technique to improve QoS parameters in cloud computing," *Neural Computing and Applications*, vol. 32, no. 16, pp. 12103–12126, Aug. 2020, doi: 10.1007/s00521-019-04266-x.

[3] M. Kumar, A. Kishor, J. Abawajy, P. Agarwal, A. Singh, and A. Y. Zomaya, "ARPS: An autonomic resource provisioning and scheduling framework for Cloud platforms," *IEEE Transactions on Sustainable Computing*, vol. 7, no. 2, pp. 386–399, Apr. 2022, doi: 10.1109/TSUSC.2021.3110245.

[4] A. Rashid and A. Chaturvedi, "Virtualization and its role in cloud computing environment," *International Journal of Computer Sciences and Engineering*, vol. 7, no. 4, pp. 1131–1136, Apr. 2019, doi: 10.26438/ijcse/v7i4.11311136.

[5] R. Rastogi and N. Aggarwal, "A review on virtualization and cloud security," in *2022 2nd International Conference on Innovative Practices in Technology and Management (ICIPTM)*, Feb. 2022, pp. 162–166, doi: 10.1109/ICIPTM54933.2022.9754172.

[6] S. Atiewi, A. Abuhussein, and M. A. Saleh, "Impact of virtualization on cloud computing energy consumption," in *Proceedings of the 2nd International Symposium on Computer Science and Intelligent Control*, Sep. 2018, pp. 1–7, doi: 10.1145/3284557.3284738.

[7] V. K. Sharma, A. Singh, K. R. Jaya, A. K. Bairwa, and D. K. Srivastava, "Introduction to virtualization in cloud computing," in *Machine Learning and Optimization Models for Optimization in Cloud*, 1st Editio., Chapman and Hall/CRC, 2022, pp. 1–14.

[8] M. Masdari and M. Zangakani, "Green cloud computing using proactive virtual machine placement: Challenges and issues," *Journal of Grid Computing*, vol. 18, no. 4, pp. 727–759, Dec. 2020, doi: 10.1007/s10723-019-09489-9.

[9] M. C. Silva Filho, C. C. Monteiro, P. R. M. Inácio, and M. M. Freire, "Approaches for optimizing virtual machine placement and migration in cloud environments: A survey," *Journal of Parallel and Distributed Computing*, vol. 111, pp. 222–250, Jan. 2018, doi: 10.1016/j.jpdc.2017.08.010.

[10] H. Xing, J. Zhu, R. Qu, P. Dai, S. Luo, and M. A. Iqbal, "An ACO for energy-efficient and traffic-aware virtual machine placement in cloud computing," *Swarm and Evolutionary Computation*, vol. 68, Feb. 2022, doi: 10.1016/j.swevo.2021.101012.

[11] R. Pushpa and M. Siddappa, "Adaptive hybrid optimization based virtual machine placement in cloud computing," in *2022 4th International Conference on Smart Systems and Inventive Technology (ICSSIT)*, Jan. 2022, pp. 1–9, doi: 10.1109/ICSSIT53264.2022.9716298.

[12] H. Zhuang and B. Esmaeilpour Ghouchani, "Virtual machine placement mechanisms in the cloud environments: a systematic review," *Kybernetes*, vol. 50, no. 2, pp. 333–368, Mar. 2021, doi: 10.1108/K-09-2019-0635.

[13] Y. Huang, H. Xu, H. Gao, X. Ma, and W. Hussain, "SSUR: An approach to optimizing virtual machine allocation strategy based on user requirements for cloud data center," *IEEE Transactions on Green Communications and Networking*, vol. 5, no. 2, pp. 670–681, Jun. 2021, doi: 10.1109/TGCN.2021.3067374.

[14] H. A. Alharbi, T. E. H. Elgorashi, and J. M. H. Elmirghani, "Energy efficient virtual machines placement over cloud-fog network architecture," *IEEE Access*, vol. 8, pp. 94697–94718, 2020, doi: 10.1109/ACCESS.2020.2995393.

[15] Y. Han, D. Guo, W. Cai, X. Wang, and V. C. M. Leung, "Virtual machine placement optimization in mobile cloud gaming through QoE-oriented resource competition," *IEEE Transactions on Cloud Computing*, vol. 10, no. 3, pp. 2204–2218, Jul. 2022, doi: 10.1109/TCC.2020.3002023.

[16] M. Sharma, M. Kumar, and J. K. Samriya, "An optimistic approach for task scheduling in cloud computing," *International Journal of Information Technology*, vol. 14, no. 6, pp. 2951–2961, Oct. 2022, doi: 10.1007/s41870-022-01045-1.

[17] K. Braiki and H. Youssef, "Multi-objective virtual machine placement algorithm based on particle swarm optimization," in *2018 14th International Wireless Communications & Mobile Computing Conference (IWCMC)*, Jun. 2018, pp. 279–284, doi: 10.1109/IWCMC.2018.8450527.

[18] M. Kumar, S. C. Sharma, A. Goel, and S. P. Singh, "A comprehensive survey for scheduling techniques in cloud computing," *Journal of Network and Computer Applications*, vol. 143, pp. 1–33, Oct. 2019, doi: 10.1016/j.jnca.2019.06.006.

[19] N. Chaurasia, M. Kumar, R. Chaudhry, and O. P. Verma, "Comprehensive survey on energy-aware server consolidation techniques in cloud computing," *The Journal of Supercomputing*, vol. 77, no. 10, pp. 11682–11737, Oct. 2021, doi: 10.1007/s11227-021-03760-1.

[20] M. A. Kaaouache and S. Bouamama, "An energy-efficient VM placement method for cloud data centers using a hybrid genetic algorithm," *Journal of Systems and Information Technology*, vol. 20, no. 4, pp. 430–445, 2018, doi: 10.1108/JSIT-10-2017-0089.

[21] V. Dinesh Reddy, G. R. Gangadharan, and G. S. V. R. K. Rao, "Energy-aware virtual machine allocation and selection in cloud data centers," *Soft Computing*, vol. 23, no. 6, pp. 1917–1932, Mar. 2019, doi: 10.1007/s00500-017-2905-z.

[22] R. Dhaya *et al.*, "Energy-efficient resource allocation and migration in private cloud data centre," *Wireless Communications and Mobile Computing*, vol. 2022, pp. 1–13, Feb. 2022, doi: 10.1155/2022/3174716.

[23] F. Alharbi, Y.-C. Tian, M. Tang, W.-Z. Zhang, C. Peng, and M. Fei, "An ant colony system for energy-efficient dynamic virtual machine placement in data centers," *Expert Systems with Applications*, vol. 120, pp. 228–238, Apr. 2019, doi: 10.1016/j.eswa.2018.11.029.

[24] M. Ranjbari and J. Akbari Torkestani, "A learning automata-based algorithm for energy and SLA efficient consolidation of virtual machines in cloud data centers," *Journal of Parallel and Distributed Computing*, vol. 113, pp. 55–62, Mar. 2018, doi: 10.1016/j.jpdc.2017.10.009.

[25] B. Akay, D. Karaboga, B. Gorkemli, and E. Kaya, "A survey on the artificial bee colony algorithm variants for binary, integer and mixed integer programming problems," *Applied Soft Computing*, vol. 106, Jul. 2021, doi: 10.1016/j.asoc.2021.107351.

[26] S. Hosseini and A. Al Khaled, "A survey on the imperialist competitive algorithm metaheuristic: Implementation in engineering domain and directions for future research," *Applied Soft Computing*, vol. 24, pp. 1078–1094, Nov. 2014, doi: 10.1016/j.asoc.2014.08.024.

[27] B. Abdi, H. Mozafari, A. Ayob, and R. Kohandel, "Imperialist competitive algorithm and its application in optimization of laminated composite structures," *European Journal of Scientific Research*, vol. 55, no. 2, pp. 174–187.

[28] R. Yazdani, M. Hajimahmoodzadeh, and H. R. Fallah, "Adaptive phase aberration correction based on imperialist competitive algorithm," *Applied Optics*, vol. 53, no. 1, pp. 132–140, Jan. 2014, doi: 10.1364/AO.53.000132.

[29] P. Humane and J. N. Varshapriya, "Simulation of cloud infrastructure using CloudSim simulator: A practical approach for researchers," in *2015 International Conference on Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials (ICSTM)*, May 2015, pp. 207–211, doi: 10.1109/ICSTM.2015.7225415.

[30] E. Rani and H. Kaur, "Study on fundamental usage of CloudSim simulator and algorithms of resource allocation in cloud computing," in *2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, Jul. 2017, pp. 1–7, doi: 10.1109/ICCCNT.2017.8203998.

BIOGRAPHIES OF AUTHORS

Seyyed-Mohammad Javadi-Moghaddam was born in Qaen, Iran, in 1975. He received the B.E. degree in computer engineering from the Ferdowsi University of Mashhad, Iran, in 1998, the MSc degree in Computer Software from AZAD University of Mashhad, Iran, in 2007, and Ph.D. degree in Computer Science from National Technical University of Athens, Greece in 2016. In 2007, he joined the Department of Computer Engineering, PNU University of Iran, as a Lecturer. Since September 2016, he is with the Department of Computer Engineering, Bozorgmahr University of Qaenat, Iran as an Assistant Professor. His current research interests include cloud computing, imbalanced data, and distributed systems. He can be contacted at email: smjavadim@buqaen.ac.ir.

Zahra Dehghani was born in Qaen, Iran, in 1985. She received a B.E. degree in computer engineering from the Payam Noor University of Birjand, Iran, in 2007. She took a Master's degree in computer engineering from the AZAD University of Birjand, Iran, in 2020. She can be contacted at email: dehghanizahra90@gmail.com.