Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Utilization based Artificial Bee Colony (UbABC) Optimization Algorithm for Virtual Machine Placement in Cloud Data Centers

Padmavathy T[1], Mehfooza M [2], Deepa D [3]
[1]*School of Computer Science and Engineering, VIT Vellore*

ARTICLE INFO

Received: 28 Dec 2024

Revised: 18 Feb 2025

Accepted: 26 Feb 2025

ABSTRACT

Cloud data centers provide services on demand to the customers through virtualization, a key technology that enables the services to be provided through Virtual Machines. Cloud data centers that host a large number of services leverage virtualization techniques to improve resource utilization and to reduce cost. The key factor to the success of the virtualization technique is to have a Virtual machine placement algorithm that optimally places the Virtual Machines on the Physical Machines of the Cloud Data center. This paper proposes a Utilization based Artificial Bee Colony Optimization algorithm that explores the placement solution based on the resource availability and utilization. The objectives of the proposed Virtual Machine placement threefold: 1. To enable the cloud service providers optimally manage a large number of heterogeneous workloads from multiple users; 2. To handle different processing requirements under different computational loads and infrastructure scales; 3. To improve the resource utilization of the active hosts. The VM allocation using the UbABC algorithm maps the VMs dynamically to the PMs and also takes care of the migrations when the underlying PM is overutilized. The optimized UbABC migrates the VMs from the overutilized to the underutilized PMs based on a certain threshold, which is determined by the Decision Tree Regression algorithm

Keywords: Cloud Computing, Genetic Algorithm, Resource Allocation, Virtual Machine, Virtualization.

INTRODUCTION

Giant cloud service providers like Google, Facebook and Microsoft deploy the services in their data centers across the globe. These services are available to the users on demand at low cost to the cloud consumers through virtualization technology. Virtualization enables sharing of the Physical Machine (PM) resources among several Virtual Machines (VMs) that are hosted on the Physical Machines (PMs). These VMs are responsible for providing the services required by the customers. This emphasizes the importance of proper placement of VMs.

Over the years with the increase in usage of the cloud services, there arises the need for providing a reliable service for the end users. The cloud service providers offer these services in various levels Infrastructure-as-a-Service (IaaS), Software-as-a-Service (SaaS), and Platform-as-a-Service (PaaS). All these models provide the services based on the pay as you use model. The services are available as open-source platforms like OpenStack, CloudStack, Shift or commercial cloud like Google, Amazon Web Services (AWS) and Azure. These services are offered through the Virtual Machines (VMs) that form the backbone of the cloud service provider. Virtualization technology enables these virtual machines to operate in a coordinated manner to provide the required services, though they operate independently. Virtualization permits the users to share the common resources of the Physical Machine (PM) to be shared by various VMs running on top of the PM.

The VMs running on the PMs have different requirements to execute the applications and to service the user request. These requirements are in the form of varied computation capability, memory requirements and networking requirements that are to be satisfied by the PMs of varied capacities. Thus, the process of mapping the VM requests with the PM is stated as a combinatorial problem with mn that spawns over ‘m’ PMs and ‘n’ VMs. With this wide

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

choice of mapping, VM placement of PM is stated as an NP-hard problem. Thus, finding out a computationally optimal solution has opened a wide option to various researches and outcomes. Many researches have been carried out by many researchers to provide an optimal or near-optimal solution for the VM placement problem.

The process of finding out the optimal solution requires the usage of various heuristic, meta-heuristic algorithms that include evolutionary algorithms, swarm intelligence, Trajectory based and Nature inspired algorithms. Brey et.al [4] improved the data center efficiency by reducing the energy consumption and increased resource utilization achieved through virtualization. In the recent literature many research works were carried out to effectively place the VMs on PMs making use of various methods and algorithms. There exists many VM placement works making use of various meta heuristic algorithms that provide solutions for varied problems.

A review on various meta heuristic algorithms in VM placement was elaborated by Alsadie et.al. [5]. The study reveals the usage of various meta heuristic algorithms in providing benefits such as minimizing energy consumption, minimizing network traffic, load balancing and improved resource utilization. These algorithms were based on Evolutionary Algorithms, Swarm Optimization algorithms, Nature-Inspired and Bio-Inspired algorithms for VM placements. They are Particle Swarm Optimization (PSO) [9] [10] [11] [12], Ant Colony Optimization [13] - [18] , Fish Swarm Optimization, Artificial Bee Colony Optimization(ABC) [18] - [22].

Asma Alkalbani et.al [6] proposed adapted Non-dominated Sorting Genetic Algorithm-II(adapted NSGA-II) for service placement in terms of VM placement in a Service oriented Computing environment. Gopu et.al [8] NSGA-III, adopted a multi-objective evolutionary algorithm, Non-dominated Sorting Genetic Algorithm-III(NSGA-III) based on the Pareto-based approach to simultaneously reduce the mentioned objectives to obtain a non-dominated solution. It makes use of non-dominated sorting along with the crowd distance calculation to calculate the solution density. The algorithm makes use of a process called Niching that prevents the algorithm from converging to a local optimum. Alboaneen, et al [7] proposed yet another metaheuristic co-optimization algorithm for VM placement along with task scheduling.

Artificial Bee Colony Optimization (ABC) algorithms with its strong capability for exploration of the solution space, finds an optimal solution using a greedy approach based on the objective function. Pushpa et. al [18] integrated the ABC algorithm with Chicken Swarm optimization(CSO) to place the VMs, considering the factors such as CPU utilization, memory and power consumption and were able to achieve a minimal power consumption for VM placement and reduced migration cost. Meshkati [19] et. al, proposed a hybrid algorithm, Hybrid Scheduling Framework based on ABC&PSO (HSF.ABC&PSO). With this HSF framework, the authors were able to reduce the energy consumption and the total execution times.

Mousavinasab et. al [23] presented an autonomous ABC algorithm that aimed at reducing the finish time of the workloads in the cloud environment. However, the authors tested the efficacy over predetermined workloads rather than the real-world traces. The physical components were not considered in the evaluation process, and the weak nature of the ABC in terms of exploitation remained the same. A focus on the CPU utilization over the real workload were considered as major criteria for scheduling by Guo et.al [24]. The authors were able to arrive at an optimal task scheduling that minimized the energy consumption. On the other hand, Wang et.al [9] proposed a ABC heuristics for workload prediction and optimized the VM placements. This method made use of threshold levels for specifying the resource availability. However, the algorithm failed in environments with higher resource availability.

Gao et.al [15] reduced the energy consumption and wastage of resources with the strengths of ABC. The benefits of Salp Swarm Optimization (SSA) and Moth-Flame Optimization were exploited by Taybeh Salehnia [25] et. al, to build a more robust meta heuristic model. The robust model used the minimization model of cost function and rules to select VMs and execute the workflow tasks.

There are quite a number of articles that focus on the reduction of network traffic in VM placement based on variations of the ABC algorithm. TRACTOR [22] is a minimization model for the VM placement that aims to reduce the power consumption and the network traffic. TRACTOR used a multi objective function model for VM placement based on the ABC algorithm. Seagull [26] is a multi-objective placement model that focuses on the VM placement by reducing the traffic and the power usage. Thus, there are many evolutionary objectives for VM placement and this

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

paper proposes a Utilization based Artificial Bee Colony Optimization algorithm that explores the placement solution based on the resource availability and utilization. The next section elaborates on the objectives of the research work.

OBJECTIVES

In this paper, Artificial Bee Colony Optimization algorithm is proposed that takes into consideration the various factors such as, the utilization of the resources in the PM. The objective function of the Utilization based Artificial Bee Colony Optimization (UbABC) algorithm is based on the resource utilization of the PM and the VMs hosted on the PMs.

The efficacy of the proposed algorithm is tested over various other algorithms on the CloudSim Simulator. The experiments were carried out under various conditions by varying the homogeneity and heterogeneity of the cloud environment. The results have proven the efficacy of the proposed UbABC algorithm in terms of resource utilization and aims to reduce the idleness of the PMs over the cloud datacenter. This is evident from the results obtained on experimentation.

This paper is organized as follows: Methods section gives a brief analysis of the proposed work. Results secion deals with the experimental setup and discusses the results obtained. The future scope of the work is stated in Discussion Scetion that concludes the paper.

METHODS

The cloud datacentre creates different types of VMs depending on varying resource requirements. These VMs are hosted on various PMs of the cloud datacentre. The objective of VM Placement (VMP) algorithm is to find an appropriate PM such that the resource utilization is increased on the PMs and also the number of active PMs are also reduced. The proposed work for VM placement includes the Utilization based Artificial Bee Colony Optimization (UbABC) for VM placement.

ABC [28] algorithm is an optimized algorithm that incorporates the swarm intelligence into the behavior of the honey bee swarm. The optimized algorithm makes use of three bees namely, employee, onlookers and scout bees. A bee that waits to choose the food area is termed an onlooker bee and the one that performs random searches is the scout bee. An onlooker bee is responsible for selecting the food source based on the probability value determined by a fitness function that gives a fitness value proportional to the nectar value. A bee that visits the previously visited food source is an employee bee. There is one employed bee for every food source. Once the food source is exhausted, the employed bee becomes a scout bee.

The proposed approach, UbABC reduces the power consumption of the data center by reducing the number of active PMs needed to host the VMs. The PMs to host the VMs are selected based on the fitness value calculated by the fitness function of PM_i to host the VM_j as given in (Eq.1 to Eq.3). The fitness function involves continuously assessing the resource utilization of all the PMs in determining the potential candidate set of VMs that requires migration to balance the allocation state and also prevents the SLA violations. The proposed method ensures the VM allocation is optimized with respect to resource utilization and power consumption.

The system of the proposed UbABC VM allocation includes a resource logging module that keeps track of the calculation of the average utilization of the resources denoted by $util_{avg}^{cpu}(i)$ and $util_{avg}^{mem}(i)$ across all time intervals. The resource utilization in terms of memory and CPU of the PM at the current time period $util^{cpu}(i)$ and $util^{mem}(i)$.

$$util(i) = util^{cpu}(i) + util^{mem}(i) \quad \ldots\ldots Eq.(1)$$

$$util^{cpu}(i) = \frac{\sum_{j=1}^{j=n_{exec}} u_j^{cpu}}{v_j^{cpu}} \quad \ldots\ldots Eq.(2)$$

$$util^{mem}(i) = \frac{\sum_{j=1}^{j=n_{exec}} u_j^{mem}}{v_j^{mem}} \quad \ldots\ldots Eq.(3)$$

The VM allocation and Migration module receives the utilization of all the PMs of the Datacenter from the resource logging module. The VM allocation and Migration module then determines the VM to be migrated based on the

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

following factors: a)The utilization rate of the PM , b)The utilization threshold of the PM, c) The number of VM migrations and d) The number of VMs currently in execution. The VM allocation and Migration module then generates the fitness value of all the physical machines with the fitness function represented by the Eq.4.

$$F_c\,(i) = \alpha\, f_{util}(i) + \beta\, f_{mig}(i) + \gamma\, f_{exec}(i) \ldots\ldots\ldots\ldots\ldots\ldots\ldots Eq.(4)$$

where, $f_{util}(i) = |\, util(i) - util_{avg}(i)$, $f_{mig}(i) = \frac{n_{mig}^{max} - n_{mig}(i)}{n_{mig}^{max} - n_{mig}^{min}}$, and $f_{exec}(i) = \frac{n_{exec}(i) - n_{exec}^{max}}{n_{exec}^{max} - n_{exec}^{min}}$ such that $\alpha + \beta + \gamma = 1$. The fitness function is calculated based on the utilization of the physical machine, the number of VM migrations and the total number of VMs currently executing on the physical machine. The parameters are given equal weightage and thus takes the value as follows: $\{\alpha, \beta, \gamma\} = \{\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\}$. The proposed model is designed with Initialization phase, and the Optimal VM placement phase. The initialization phase is used to prepare initial set of VM and jobs with resource needs. The initialization algorithm, Algorithm 1, creates an initial solution in terms of PM-VM mapping as a solution_matrix. This solution_matrix is updated on each iteration of the algorithm based on the fitness function. The fitness of each bee in terms of the server on which the VM is placed is calculated. This fitness value is updated in every phase of the employed bees, onlooker bees and scout bees.

Algorithm 1: Initialization Phase

INPUT: List of Physical Machine $PM_i, 1 \le i \le n$ and Virtual Machine $VM_i, 1 \le i \le n$

OUTPUT: Initial Assignment of PM_i to suitable VM_j

1: Initialized set of food sources from available set of resources $VM_j = \{CPU_j, Mem_j\}$
2: Prepare a current list of Physical Machine with resource needs $PM_i = \{CPU_i, Mem_i\}$
3: Initialize fitness function and trial Counter arrays for identifying suitable virtual machine
4: Define the following early stopping parameters, $BestFitnessAchieved$ and $EarlyStoppingThreshold$
5: For each physical resource from available list $A_{Resourcelist} = \{CPU_j, Mem_j\}$ do
   For each virtual service request $VM_j, 1 \le j \le m$ do
      i. Apply random placement approach for random host to the VM
   End For
   Calculate the updated fitness solution
   Increment the trail counter from Zero
6: End For
7: Return the selected VM for PM placement
8: End Procedure

The Optimized_ ABC_VM_Placement algorithm aims to find the optimal or near-optimal solution with new set of VM placement solution generated by employed bees, updated solutions by onlooker news and with new random solutions introduced by the scout bees dropping the solutions with least fitness values that has not changed over time. Algorithm 2 employs the usage of the Threshold value that is determined by the Decision Tree Regression algorithm, a non-parametric supervised learning algorithm. The decision tree is constructed by recursive partitions and choosing the best split at each node of the tree. The splitting is done based on the calculated fitness values of the PMs and the value of the Threshold is determined based on the fitness value that contributes to variance reduction as given in Equation (5).

$$\Delta Var = Var_{parent} - \left[\frac{N_{left}}{N} Var_{left} + \frac{N_{right}}{N} Var_{right}\right] \ldots\ldots\ldots\ldots\ldots\ldots Eq.(5)$$

Here Var_{parent} represents the variance of the fitness value of the PM, represented by the current node and Var_{left} , Var_{right} represents the variance of the fitness value of the left and right subtrees of the tree after split. The threshold is then determined by the equation (6) as represented below:

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

$$Threshold = \frac{1}{N}\sum_{i=1}^{N}(Var_i - \Delta Var)^2 \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots Eq.(6)$$

Algorithm 2 Optimized VM_Placement

Input: Initial Solution of PM to VM (solution_matrix)

Output: Updated Assignment of PM to VM based on Threshold (solution_matrix)

```
e_fitness = compute_fitness_for_each_employed_bee ()
total_fitness = sum(e_fitness)
for each onlooker bee in the population do
    select an employed bee based on fitness probability
    for each VM in num_vms do
        if random () < Threshold then
            neighbor_solution = solution_matrix. copy ()
            neighbor_solution [selected_server, VM] = 1 - [selected_server, VM]
            current_fitness = objective_function(solution_matrix)
            neighbor_fitness = objective_function(neighbor_solution)
            end if
            if neighbor_fitness > current_fitness then
                    solution_matrix = neighbor_solution. copy ()
        end if
    end for
end for
```

RESULTS

The experiments were carried out in the CloudSim toolkit [29] to test the efficacy of the proposed UbABC algorithm over other meta-heuristic algorithms in the VM placement. A datacenter with the fat tree topology was created with 50 to 500 Physical Machines with heterogeneous configuration. The VM placement algorithm available in CloudSim is extended with UbABC. The UbABC algorithm was run on various iterations by varying the number of VMs between 100 and 1000. The efficacy of the cloudlet allocation was evaluated based on the metrics such as makespan and fitness function values. Makespan, which indicates the total time required to host and execute all the cloudlets across virtual machines, contributes mostly to the testing of the efficacy. UbABC makes a harmonious distribution that minimizes the makespan as shown in the figures 1-3. The simulation parameters are listed in table 1.

Table 1: Simulation Parameters

Data Centers	No.of Data Brokers	No.of Physical Machines	No.of Virtual Machines	Host MIPS	Host RAM	No.of PEs	VM MIPS	VM RAM	No.of. PEs allocated to VM	Cloudlet Length
1	1	100-200	100-1000	1000	20 GB	04-06	100-900	2048	1	1000-2000

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

The total execution time measured in terms of makespan which is represented by the total time taken for the execution of all cloudlets. From the figures 4-7, it is evident that makespan is lower in case of UbABC compared to FCFS, SJF, RoundRobin and PSO. With reduced makespan, the cloud service providers manage a large number of heterogeneous workloads from multiple users and also provide an optimized VM placement, with VMs being effectively utilized by actively processing tasks for a significant portion of the total makespan. Minimal makespan enhances user satisfaction, optimizes resource utilization, and improves the overall quality of service. By altering the number of PMs and VMs, multiple operational environments are mimicked to provide the insight of the performance of the proposed UbABC under different computational loads and infrastructure scales. With varied MIPS, the efficiency of UbABC in handling VMs with different processing requirements is proved.

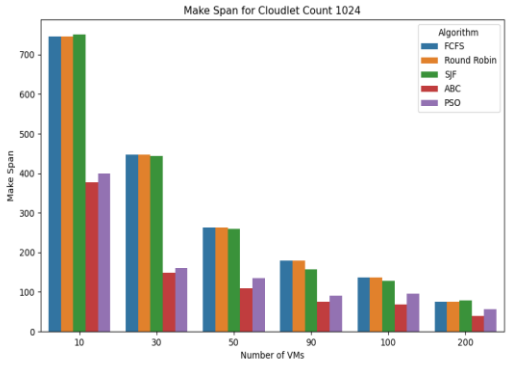

Figure 1: Makespan on Varied Iterations, Cloudlet = 1024

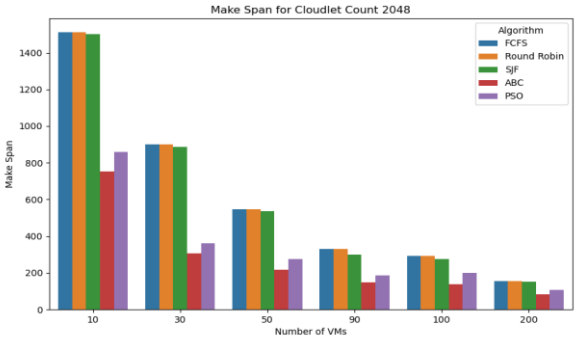

Figure 2: Makespan on Varied Iterations, Cloudlet = 2048

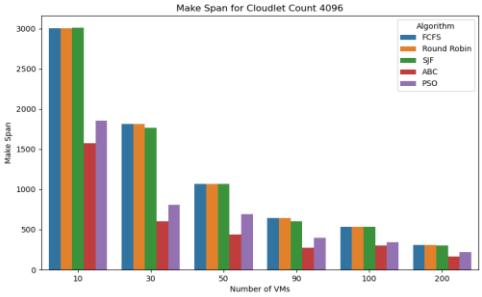

Figure 3: Makespan on Varied Iterations, Cloudlet = 4096

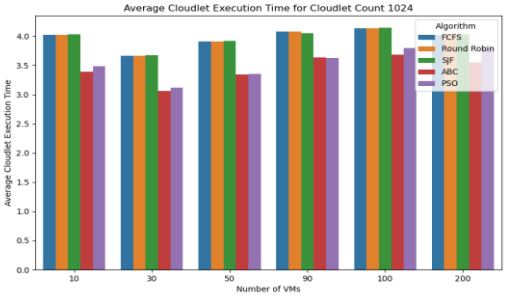

Figure 5: Average Cloudlet Execution Time, Cloudlet = 1024

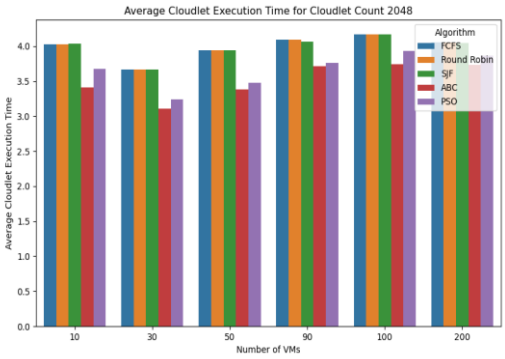

Figure 6: Average Cloudlet Execution Time, Cloudlet = 2048

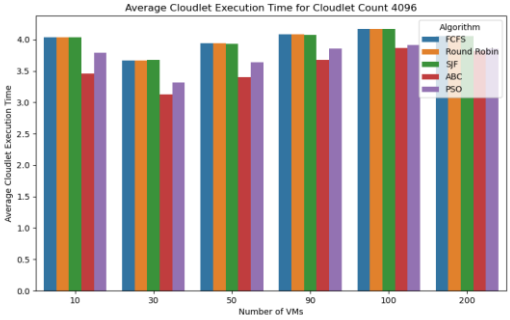

Figure 7: Average Cloudlet Execution Time, Cloudlet = 4096

The VM allocation using the UbABC algorithm maps the VMs dynamically to the PMs and also takes care of the migrations when the underlying PM is overutilized. The optimized UbABC migrates the VMs from the overutilized to

Journal of Information Systems Engineering and Management
2025, 10(38s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

the underutilized PMs based on a certain threshold, which is determined by the Decision Tree Regression algorithm. Thus, UbABC takes care of optimized placements avoiding the SLA violations by proactively addressing the resource shortages. The proposed system proactively manages the resource allocation by predicting the CPU thresholds marking the overutilization using the Decision Tree Regression algorithm that improves the prediction accuracy and the robustness of the learning model. Table 2 shows the results of various simulations.

Table 2 : Comparison of Number of Migrations

PMs	VMs	No.of. Migrations			
		First Fit	Best Fit	PSO	UbABC
50	100	17	21	4	2
25	250	6	7	2	0
250	500	10	8	1	0
500	750	12	14	1	0

A detailed analysis of UbABC placement reveals that the algorithm outperforms its traditional counterparts. The results exhibited in the Table 2 shows the lowest number of migrations by optimally allocating the VMs thereby reducing the overheads and the performance degradations associated with migrations. Table 2 shows the average number of migrations with an average low number of migrations with UbABC. This minimal migration indicates the optimal allocation of VMs, thereby emphasizing the relative stability of UbABC in comparison to other native algorithms as shown in theTable 2.

Moreover, the energy consumption is also lesser as the number of active hosts is also lesser in case of UbABC. Table 3 shows the number of idle hosts which proves that the resource utilization is higher in terms of UbABC. UbABC packs the VMs on a lesser number of PMs but the energy consumption increases on the PMs, which is evident from table 4. In future, the energy consumption is to be optimized across various PMs, on various runs of the simulation.

Table 3: Analysis of Number of Idle Hosts

PMs	VMs	No. of Idle Hosts			
		First Fit	Best Fit	PSO	UbABC
50	100	6	4	30	30
25	250	6	4	30	30

Table 4: Analysis of Energy Consumption

PMs	VMs	Energy Consumption(kW-h)			
		First Fit	Best Fit	PSO	UbABC
50	100	2.73	1.75	4.4	4
25	250	2.94	2.94	5	3.72
250	500	2.79	2.79	5	4.61
500	750	2.8	2.8	5	5

The efficacy of the algorithm has emerged from the fitness function that plays a vital role in the VM placement. The nest section concludes the research with with the discussions on the results and the future works.

DISCUSSION

Utilization based ABC provides faster convergence to optimal solutions since it focuses on the promising solutions by reducing the number of iterations needed to converge to the placement solution. Moreover, the number of idle hosts are higher than the traditional algorithms leading to tightly packed VM placement thereby providing improved resource utilization. But this increases the amount of energy needed to host the VMs on PMs and to meet the functional requirements. In future, the efficacy of the algorithm can be tested on fixing up the threshold values on the resource utilization across multiple heterogeneous datacenter placements.

REFRENCES

[1] Xu, G., Ding, Y., Zhao, J., Hu, L., & Fu, X. (2013). A novel artificial bee colony approach of live virtual machine migration policy using Bayes theorem. The Scientific World Journal, 2013.
[2] Chen, X., Gu, C., Gao, X., Shen, Y., Sun, Z., & Huang, H. (2024). Virtual Machine Placement for Minimizing Image Retrieval Cost and Communication Cost in Cloud Data Center. IEEE Transactions on Network and Service Management.
[3] Kaur, A., Kumar, S., Gupta, D., Hamid, Y., Hamdi, M., Ksibi, A., ... & Saini, S. (2023). Algorithmic approach to virtual machine migration in cloud computing with updated SESA algorithm. Sensors, 23(13), 6117.
[4] Brey, T., & Lamers, L. (2009). Using virtualization to improve data center efficiency. the green grid, whitepaper, 19.
[5] Alsadie, D. (2022). Virtual Machine Placement Methods using Metaheuristic Algorithms in a Cloud Environment-A Comprehensive Review. International Journal of Computer Science & Network Security, 22(4), 147-158.
[6] Alkalbani, A. M., Al Ruqeishi, K. B., Salah, A., & Mohamed, M. F. (2023). Virtual machine placement in service-oriented computing environments. Service Oriented Computing and Applications, 1-9.
[7] Alboaneen, D., Tianfield, H., Zhang, Y., & Pranggono, B. (2021). A metaheuristic method for joint task scheduling and virtual machine placement in cloud data centers. Future Generation Computer Systems, 115, 201-212.
[8] Gopu, A., Thirugnanasambandam, K., AlGhamdi, A. S., Alshamrani, S. S., Maharajan, K., & Rashid, M. (2023). Energy-efficient virtual machine placement in distributed cloud using NSGA-III algorithm. Journal of Cloud Computing, 12(1), 124.
[9] Wang, S., Liu, Z., Zheng, Z., Sun, Q., & Yang, F. (2013, December). Particle swarm optimization for energy-aware virtual machine placement optimization in virtualized data centers. In 2013 International Conference on Parallel and Distributed Systems (pp. 102-109). IEEE.
[10] Xiong, A. P., & Xu, C. X. (2014). Energy efficient multiresource allocation of virtual machines based on PSO in cloud data center. Mathematical Problems in Engineering, 2014.
[11] Gao, J., & Tang, G. (2013, October). Virtual machine placement strategy research. In 2013 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (pp. 294-297). IEEE.
[12] Dashti, S. E., & Rahmani, A. M. (2016). Dynamic VMs placement for energy efficiency by PSO in cloud computing. Journal of Experimental & Theoretical Artificial Intelligence, 28(1-2), 97-112.
[13] Feller, E., Rilling, L., & Morin, C. (2011, September). Energy-aware ant colony based workload placement in clouds. In 2011 IEEE/ACM 12th International Conference on Grid Computing (pp. 26-33). IEEE.
[14] Ferdaus, M. H., Murshed, M., Calheiros, R. N., & Buyya, R. (2014). Virtual machine consolidation in cloud data centers using ACO metaheuristic. In Euro-Par 2014 Parallel Processing: 20th International Conference, Porto, Portugal, August 25-29, 2014. Proceedings 20 (pp. 306-317). Springer International Publishing.
[15] Gao, Y., Guan, H., Qi, Z., Hou, Y., & Liu, L. (2013). A multi-objective ant colony system algorithm for virtual machine placement in cloud computing. Journal of computer and system sciences, 79(8), 1230-1242.
[16] Tawfeek, M. A., El-Sisi, A. B., Keshk, A. E., & Torkey, F. A. (2014). Virtual machine placement based on ant colony optimization for minimizing resource wastage. In Advanced Machine Learning Technologies and

Research Article

Applications: Second International Conference, AMLTA 2014, Cairo, Egypt, November 28-30, 2014. Proceedings 2 (pp. 153-164). Springer International Publishing.

[17] Malekloo, M., & Kara, N. (2014, December). Multi-objective ACO virtual machine placement in cloud computing environments. In 2014 IEEE Globecom Workshops (GC Wkshps) (pp. 112-116). IEEE.

[18] Pushpa, R., & Siddappa, M. (2021). An optimal way of VM placement strategy in cloud computing platform using ABCS algorithm. International Journal of Ambient Computing and Intelligence (IJACI), 12(3), 16-38.

[19] Meshkati, J., & Safi-Esfahani, F. (2019). Energy-aware resource utilization based on particle swarm optimization and artificial bee colony algorithms in cloud computing. The Journal of Supercomputing, 75(5), 2455-2496.

[20] Kimpan, W., & Kruekaew, B. (2016, August). Heuristic task scheduling with artificial bee colony algorithm for virtual machines. In 2016 Joint 8th International Conference on Soft Computing and Intelligent Systems (SCIS) and 17th International Symposium on Advanced Intelligent Systems (ISIS) (pp. 281-286). IEEE.

[21] Javadi-Moghaddam, S. M., & Dehghani, Z. (2023). Virtual machine placement in cloud using artificial bee colony and imperialist competitive algorithm. International Journal of Electrical & Computer Engineering (2088-8708), 13(4).

[22] Nabavi, S. S., Gill, S. S., Xu, M., Masdari, M., & Garraghan, P. (2022). TRACTOR: Traffic-aware and power-efficient virtual machine placement in edge-cloud data centers using artificial bee colony optimization. International Journal of Communication Systems, 35(1), e4747.

[23] Mousavinasab, Z., Entezari-Maleki, R., & Movaghar, A. (2011). A bee colony task scheduling algorithm in computational grids. In Digital Information Processing and Communications: International Conference, ICDIPC 2011, Ostrava, Czech Republic, July 7-9, 2011, Proceedings, Part I (pp. 200-210). Springer Berlin Heidelberg.

[24] Guo, L., Zhao, S., Shen, S., & Jiang, C. (2012). Task scheduling optimization in cloud computing based on heuristic algorithm. Journal of networks, 7(3), 547.

[25] Salehnia, T., Ghaffari, A., & Abualigah, L. (2024). Workflow Scheduling in Cloud System: An Optimal Rule-Based Virtual Machine Selection Technique Using MFSSA.

[26] Nabavi, S., Wen, L., Gill, S. S., & Xu, M. (2023). Seagull optimization algorithm based multi-objective VM placement in edge-cloud data centers. Internet of Things and Cyber-Physical Systems, 3, 28-36.

[27] Nauman, U., Zhang, Y., Li, Z., & Zhen, T. (2024). Secured VM Deployment in the Cloud: Benchmarking the Enhanced Simulation Model. Applied Sciences, 14(2), 540.

[28] Karaboga, D., & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. Journal of global optimization, 39, 459-471.

[29] Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A., & Buyya, R. (2011). CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software: Practice and experience, 41(1), 23-50.