International Journal of Swarm Intelligence Research
Volume 14 • Issue 1

Multi-Objective Binary Whale Optimization-Based Virtual Machine Allocation in Cloud Environments

Ankita Srivastava, Babasaheb Bhimrao Ambedkar University, India
Narander Kumar, Babasaheb Bhimrao Ambedkar University, India*
https://orcid.org/0000-0003-4633-1879

ABSTRACT

With the rising demands for the services provided by cloud computing, virtual machine allocation (VMA) has become a tedious task due to the dynamic nature of the cloud. Millions of virtual machines (VMs) are allocated and de-allocated at every instant, so an efficient VMA has been a significant concern to enhance resource utilization and depreciate its wastage. Encouraged by the prodigious performance of the nature-inspired algorithm, the binary whale optimization approach has been eventuated to get to grips with the VMA issue with the focus on minimizing the resource waste and volume of servers working actively. The deliberate approach's accomplishment is assessed against the literature's well-known algorithms for VMA issues. The comparison results showed that the least resource wastage fitness of 15.68, minimum active servers of 216, and effective CPU and memory utilization of 88.31% and 88.79%, respectively, have been achieved.

KEYWORDS

Binary Whale Optimization Algorithm, Cloud Computing, Metaheuristic Algorithm, Resource Allocation, Resource Wastage, Swarm Optimization, Virtual Machine Allocation, Virtualization

1. INTRODUCTION

The progression in technology has led to the emergence and evolvement of cloud computing from the basic computing paradigm of distributed, parallel, and grid computing. It has revolutionized the procedure for managing the data or information and the resources which have impacted human society socially and economically. The services are offered to the users in the resources form (e.g., storage, CPU, servers, network, and application). These resources are organized at one central point, the data center, from where accessing these resources comes at ease. The overall management of data centers is the responsibility borne by the service providers. The service providers serve the users' interests with three basic services via the internet IaaS for hardware resources, PaaS for runtime environment,

DOI: 10.4018/IJSIR.317111
*Corresponding Author

and SaaS for software resources. These former services are being accomplished with virtualization. Virtualization facilitates the creation and configuration of VMs possessing variant operating systems with varying resources (memory, CPU, and storage) that are then nailed on the host machine to furnish the services desired by users. It also expedites sharing of multiple VMs deployed on one distinct server and sharing the hardware resources. To accomplish the request or interest demanded by users, VMs are created and configured dynamically with variable configuration and resource demands. The key objective in this reference is to allocate resources in ways that they are effectively utilized, decreasing resource waste and resulting in low operational costs. Adopting an effective VMA algorithm is one way to accomplish this. VMA allows VMs to be placed on the servers such that computing resources must be efficiently utilized while also reducing the volume of active servers. With the decisive allocation techniques, there come challenges, including a reduction in QoS, reduction of performance with the curtailed energy usage, and then satisfying the user's QoS within the promised SLA. An effective VMs allocation narrows down the count of active and balanced multidimensional resource usage by servers, ultimately reducing the energy expenditure by the cloud. Figure 1 demonstrates the allocation of VMs in which all the VMs are allotted randomly to the server, while after optimization in figure 2, the VMs are consolidated in the first three servers, and the rest of the unutilized servers are turned off, thus saving the resources and energy. Identifying optimal VMs allocation belongs to the NP-complete problem (Lo, V. M. 1983), and achieving the optimum resolution of this is typically computationally infeasible, when the cloud involves multiple hosts and users (Widmer, T., Premm, M., & Karaenke, P. 2013). The issue can be addressed to minimize resource wastage and the number of active servers in the cloud data center. Various methods have been applied to resolve this issue in the literature. A theorem was given (Wolpert, D. H., & Macready, W. G. 1997), which stated that not a single meta-heuristic algorithm is available which can efficiently resolve all the optimization problems. In this regard, the WOA (Mirjalili & Lewis, 2016) has been successfully utilized for various optimization problems (Prakash, D.B. & Lakshminarayana, C.,2017; Sun, Wang, Chen, & Liu, 2018; Chen, H., Xu, Y., Wang, M., & Zhao, X., 2019; Too, J., Mafarja, M., & Mirjalili, S., 2021).

A meta-heuristic-based Genetic Algorithm (GA) has been used widely to resolve the allocation problem (Kaaouache, M. A., & Bouamama, S.,2018; Abohamama, A. S., & Hamouda, E., 2020). These methods are the victims of basically two issues. These algorithms converge when a large number of iterations are performed and claim for a large population size as opposed to other metaheuristic algorithms. Due to this, it involves a high processing time, and sometimes it also suffers from premature convergence. While other meta-heuristic algorithms like ant colony-based, grey-wolf-based algorithms converge with lower population size. This motivates exploiting different meta-heuristic algorithms that benefit from low population size and iterations or generations. The study introduces a novel state of the art for accomplishing VM allocation, which subsequently truncates the active servers' count and minimizes the undergoing resource wastage, thus enhancing the proportion of resource utilization.

Further, the study is assembled as section 2 background study for the associated work being performed, and section 3 demonstrates the proposed method. The algorithm for the given work is discussed in section 4, which is supported through simulation. The result analysis is done in section 5, and further discussion is performed in section 6, which is finally concluded in section 7.

2. LITERATURE REVIEW

Extensive advances in the cloud have led to a dramatic rise in the volume of data centers. Hence, energy consumption generated by these centers continues to upsurge the cost incurred by the cloud system. So, in this scenario mitigating the energy exhausted by these centers has now become a critical concern that needs urgent attention. Energy consumed through these data centers varies directly with resource usage, thus, to prevent energy consumption, one needs to be careful about resource consummation as it aids in reducing operating costs. The resources should be used proficiently, thus

Figure 1. VMA before optimization

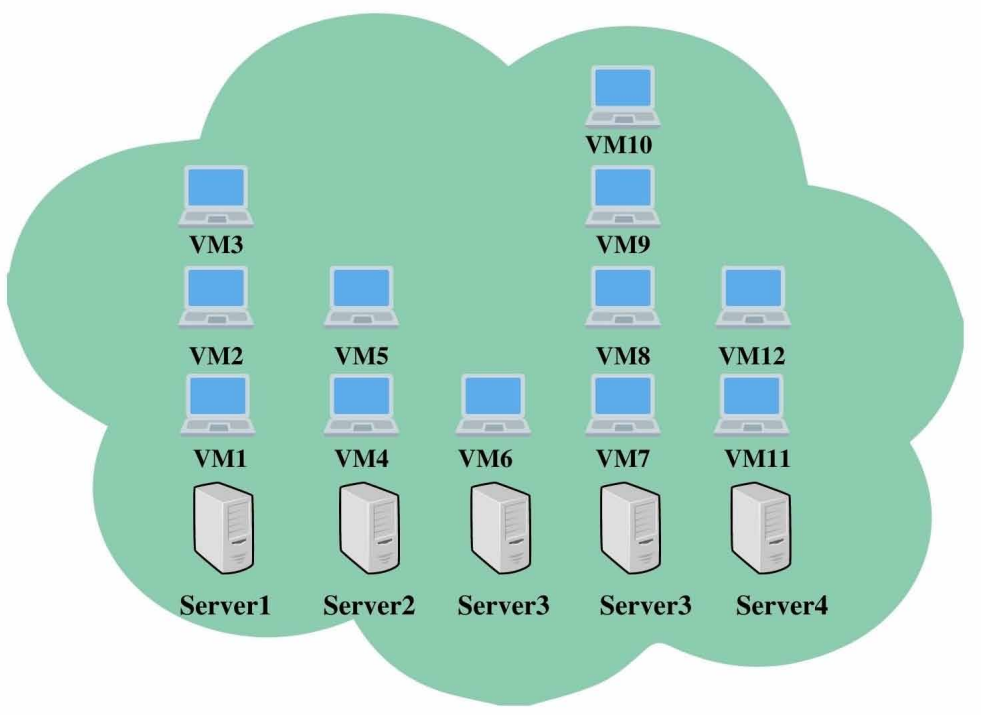
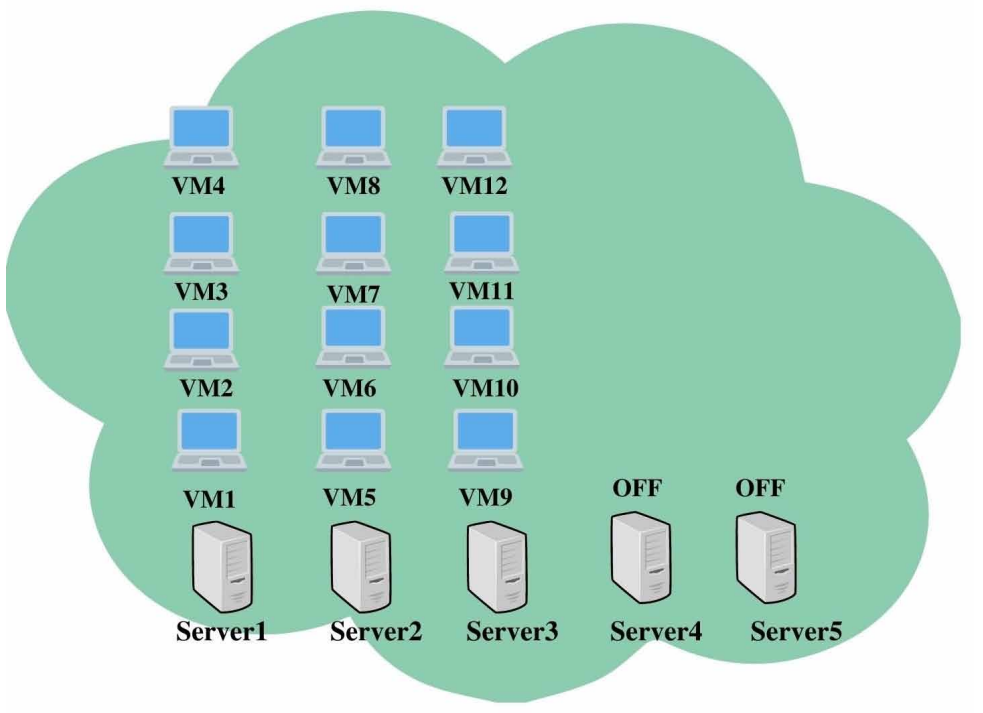


Figure 2. VMA after optimization



minimizing resource wastage. The most effective method to accomplish this is utilizing virtualization to allocate resources effectively.

Various learning has been put forward which are proficient in resource allocation. The most basic way to confront the VMA problem is by using linear programming. Stochastic Integer Programming is cast-off to reduce the price for hosting VMs in multiple cloud environments (Chaisiri, S., Lee, B. S., & Niyato, D., 2009). Heuristic algorithms encompass the Best Fit, or First Fit algorithms are also used. A Modified Best Fit Decrease algorithm is implemented to consolidate virtual machines (Beloglazov, A., & Buyya, R., 2010). The estimation module is applied to predict the future load of the system, and the scheduler is employed to schedule the expected and unpredicted loads. In turn, these schedulers enforce the technique of column generation to exploit integer linear or quadratic programming optimization problems (Vakilinia, S., Heidarpour, B., & Cheriet, M., 2016). These methods produce the locally optimal solution, bringing about the imbalanced consumption of energy and resources and leading to greater SLA violations.

Some scholars have used heuristic methods that produce higher efficiency in tackling the VMA problem. The heuristic method is unified with the machine learning technique for conducting VMA which is correlated with traffic, energy, migration, QoS, and scalability (Pahlevan, A., Qu, X., Zapater, M., & Atienza, D. 2017). An improved analytic hierarchy procedure is presented, which processes the task before the allocation, and allocation is performed using optimization methods considering the given load and bandwidth (Gawali, M. B., & Shinde, S. K., 2018). Another approach that considers VMs to server allocation initially utilizes the renowned First Fit heuristic and MBFD and then performs the iterative search for optimal mapping (Zhang, X., et al., (2019). These methods have not taken the underuse of resources as their main focus. Another approach to tackle this issue is the assignment method. The author has utilized a modified assignment method to optimize the VMs allocation by minimizing SLA violation and resource wastage (Toutov, A., et al., 2021). This method does not take into account the active server's volume.

Some scholars adopted intelligent computation for VMs allocation, which includes genetic algorithms. An allocation technique is being contrived in virtualized network function at the data center, where two genetic algorithms are analyzed for VMA and scaling (Rankothge, W., et al., 2017). A multi-objective-based genetic algorithm is presented for VMA, which performs the encoding of chromosomes using group method, crossover, and further mutation operations whose length is varying. It decodes the chromosome using a three-dimensional split method (Qiang, L., 2011). Another approach is the Ant Colony Algorithm which has been elaborated, where a collection of servers is selected to allocate VMs and data on it (Shabeera, T. P., et al., 2017). Resource allocation has been resolved as a convex optimization problem. The major drawback of this algorithm was that it was only applicable in the case of homogenous VMs, and it didn't consider the migration aspect of the VMs to utilize the resources efficiently. A deep reinforcement framework has been presented for allocating resources, resulting in energy efficiency for cloud radio access networks. In this, a learning agent having deep reinforcement is defined through several parameters like reward method, action space, and state space. Further, the action-value function is approximated using deep neural networks, and resource allocation is formulated formally (Peng, Y., et al., 2017). One of the shortcomings of this work was the large population size was considered, which has increased the processing time of the algorithm. Combinatorial optimization is undertaken, which considers Quantum Genetic Algorithm (GA) & Ant Colony Optimization (ACO) algorithm with GA for resolving resource allocation issues. A mapping process is performed between the resource allocation matrix and chromosomes of the aforesaid algorithm and aids in refining the resource utility (Xia, W., & Shen, L. 2018). It worked significantly on the small data set, but it didn't discuss the large dataset. An enhanced cuckoo search procedure is familiarized to confront the VMA issue, which effectively narrows down the energy dissipation. It is motivated by the brooding behavior of cuckoo birds. In this exploration, the process is performed via Levy flights which strived to perceive the best local solution among the specified collection of different solutions. Eventually, fitness function assessment was performed to foster the

new list for VMA. This algorithm works efficiently with fewer VMs (Barlaskar, E., et al., 2018). It didn't give any idea about CPU and memory consumption, which is one of the important aspects of VM allocation. An ACO system embedded with a new heuristic is explored. In this methodology, four steps are there, which are initializing the pheromone, defining heuristic information, construction solution, and ultimately the last step is updating the pheromone. Firstly, the pheromones are initialized in which the assignment of VMs is being performed to servers through the First Fit Decrease process. Then the heuristic routine is designed considering CPU utilization, energy exhaustion, CPU capacity, and further VM is set down on the server in accordance with this information. After these, ants construct the solution with the extreme probability utilizing the pseudo-random proportional rule. Finally, the pheromone is updated through pheromone evaporation, which attempts to substantiate the global best solution (Alharbi, F., et al., 2019). It has not considered the resource utilization and the dynamic migration of the VMs across PMs. Another nature-inspired heuristic algorithm is the Flower Pollination algorithm. It is executed on a methodology termed dynamic switching probability. A methodology is enacted that quickly identifies the closest optimal solution and maintains a balance between exploitation of local search and exploration for the global search. It takes into deliberation the storage, processor, and memory restraint of a server and allocates VMs to it by minimizing and emphasizing the consummation of energy (Usman, M. J., et al., 2019). The algorithm works for a single objective, which restricts the technique's broad work scope. An Intelligent Water Drop technique is used to allocate VMs by optimizing the task execution and security consideration (Dubey, K., & Sharma, S. C., 2020). A metaheuristic approach grounded on the binary gravitational search process is propounded, which is established on the law of gravity to perform energy-efficient VMA. In this procedure, agents are entitled as entities, and their capability is quantified in relation to their masses. Agent position is set randomly, and, in every iteration, the fitness assessment is accomplished, which signifies the mass per agent, and the fitness value is updated accordingly. Then the mass is determined for each agent based on updated fitness value which ultimately assists in the assessment of acceleration. Then a new position is calculated beside the new velocity (Abdessamia, F., et al., 2020). This approach only considers energy, but CPU usage and availability are not considered, and the algorithm was analyzed for 100 VMs only. A Fair Fit algorithm is put forward, effectively increasing resource use by restraining its imbalance across multiple dimensions (Gohil, B. N., et al., 2021). Energy efficiency, resource wastage, and performance have been addressed using neural networks. The future demands have been forecasted using a forward feed neural network, and then auto-scaling is performed based on the clustering of resource requirements predicted, and further VMs are being allocated to the servers (Saxena, D., & Singh, A. K., 2021). It suffered the drawback of manually selecting the number of nodes in the input and output layers of the predictor. While a perceptive priority-aware VM allocation is presented, which prioritizes the application based on memory, computing, and bandwidth using a machine learning-based model. The work is identified in mitigating the power consumption, average response time, and execution time (Savitha, S., & Salvi, S. 2021). The work stated above does not account for resource wastage and active servers, which automatically reduce energy consumption and enhance resource utilization. Further, the technique didn't discuss its implementation in a real working environment. An integrated methodology of Artificial Bee Colony with Chicken Swarm optimization has been implemented to allocate VMs with lesser power consumption, load, and migration cost (Pushpa, R., & Siddappa, M., 2021). Another swarm-based technique is being identified to minimize power consumption and resource wastage (Saravanan, D., et al.,2021). It only considered the two performance indicators of allocation, didn't give any idea about CPU and RAM utilization and didn't observe the migration aspect of VMs while allocating. Another approach for allocation has been proposed using Monarch Butterfly optimization, which worked on the packaging efficiency and simultaneously reduced the number of active servers (Ghetas, M. 2021). This algorithm does not account for reducing the wastage of resources and VMs migration. The algorithms discussed above suffered from various limitations, and thus it motivates the author to analyze the issue and devise a technique that could resolve the issue efficiently.

The Whale Optimization Algorithm (WOA) has recently piqued the interest of several scholars. Being an algorithm motivated by nature, it is ingrained in the humpback hunting procedure of whales. In terms of function, it's similar to genetic algorithms, and it has many benefits: faster convergence, fewer parameters, and simplified operation. It provides an efficient result with lower population size and iterations, reducing the algorithm's time complexity and contributing to the reason for adopting this algorithm. Furthermore, WOA can't be directly applied to discrete problems, so a binary version of WOA is required to resolve the VMA issue.

3. PROPOSED RESOURCE-AWARE VMA BASED ON BWOA

3.1 An Introduction to BWOA

Being a swarm-based optimization algorithm, the WOA is driven by the humpback whales hunting behavior, origina

lly brought forward by Seyedali Mirjalili and Andrew Lewis (Mirjalili, S., & Lewis, A. 2016). It includes three operators which simulate prey's search, encircling prey, and bubble-net foraging behavior shown by humpback whales. Contrasting to other meta-heuristic functions, WOA is extremely competitive and far superior to conventional techniques.

Standard WOA is appropriate for solving continuous space problems. However, the VMA issue belongs to the class of discrete optimization problems, so it's unfeasible to apply this algorithm directly to VMA. A modified genre of WOA is the paramount requirement for being applied to problems with discrete space. An advanced binary genre of WOA was proposed (Hussien, A. G., et al., 2020) to resolve discrete optimization problems.

BWOA is parallel to WOA, except that the whale's position is hosted in binary code. A transfer method is added in BWOA, which maps the distance vector to probability which further aids in the position update of the agent. The target is anticipated to be the recent best solution furnished through the algorithm. Then the best search agent is defined, and other agents attempt to modify the position as:

$$\vec{P}\left(k+1\right) = \overrightarrow{P^*}\left(k\right) - \vec{B} \times \vec{G} \tag{1}$$

$$\vec{G} = \left| H \times \overrightarrow{P^*}\left(k\right) - \vec{P}\left(k\right) \right| \tag{2}$$

Here, k indicates the recent iteration, B and H represent the coefficient vectors, $P*$ denotes the best alternative found so far, P stands for the position vector, and $\left|\right|$ finds absolute value and $\times$ is element by element multiplication. Vector B and H are evaluated as

$$B_{\rightarrow} = 2l_{\rightarrow} \times \lambda_{\rightarrow} - l_{\rightarrow} \tag{3}$$

$$H_{\rightarrow} = 2 \times \lambda_{\rightarrow} \tag{4}$$

Where l gets decelerated from $2to0$ over the entire duration of looping and λ is a random vector in $\left[0,1\right]$.

The mathematical simulation of the exploitation phase is as follows:

1. *Shrinking Encircling:* It is incurred by decreasing l values which has a random value in $\left[-l,l\right]$ where l is decremented from $2to0$ during the course of iteration.

Spiral Updating: It computes the distance across the prey and the agent, which is expressed as:

$$\vec{P}\left(k+1\right)=\overrightarrow{X^{'}}\times e^{\mu\beta}\times\cos\left(2\pi\beta\right)+\overrightarrow{P^{*}}\left(k\right) \tag{5}$$

$$\overrightarrow{X^{'}}=\left|\overrightarrow{P^{*}}\left(k\right)-\vec{P}\left(k\right)\right| \tag{6}$$

Where, β takes random values in the range $\left[-1,1\right]$ and μ has fixed value.

Search for prey (Exploration phase):

$$\vec{E}=\left|H\times\overrightarrow{P_{rand}}-\vec{P}\right| \tag{7}$$

$$\vec{P}\left(k+1\right)=\overrightarrow{P\_rand}\left(k\right)-\vec{B}\times\vec{E} \tag{8}$$

Where, $\overrightarrow{P\_rand}$ displays a random position vector considered from the population in the current iteration.

The sigmoid function is adopted to ensure the probability for every bit lies in $\left[0,1\right]$, the sigmoid function adopted is:

$$F\left(x_i^j\left(k\right)\right)=\frac{1}{1+e^{-S_i^j\left(k\right)}} \tag{9}$$

Here, $x_i^j\left(k\right)$ stands for the distance across the agent, and α is the random real number, hence the position for the agent is given by:

$$x_i^j\left(k+1\right)=\begin{cases}1 \ if\,\alpha<F\left(x_i^j\left(k\right)\right)\\ 0 \ otherwise\end{cases} \tag{10}$$

This is the transition of $x_i^j\left(k+1\right)$.

3.2 Resource Wastage Model

Various VMA algorithms seek to maximize the efficient exploitation of available resources by striving for the least volume of resource waste on the servers. Resource wastage is highly dependent on memory and CPU associated with the server, which is being formulated as follows:

$$R_P^{wastage}=\frac{\left|NF_p^{cpu}-NF_p^{mem}\right|+\eta}{U_p^{cpu}+U_p^{mem}} \tag{11}$$

Where $R_P^{wastage}$ denotes resource wastage, NF_p^{mem}, NF_p^{CPU} shows the remaining memory and CPU related to server p, in normalized form, U_p^{mem} (table 1) and U_p^{CPU} denotes memory and CPU usage by the server p, in the normalized form and η has a value 0.00001 which acquires generally a very small positive real number. Thus, the total-sum resource wastage can be identified as:

$$f\left(X\right)=\sum_{j=1}^{m}R_{j}^{wastage}=\frac{\sum_{j=1}^{m}y_{j}\times\left|\left(T_{j}^{cpu}-\sum_{i=1}^{n}\left(x_{ij}\times\alpha c_{i}\right)\right)-\ \left(T_{j}^{mem}-\sum_{i=1}^{n}\left(x_{ij}\times\alpha m_{i}\right)\right)\right|+\eta}{\sum_{i=1}^{n}\left(x_{ij}\times\alpha c_{i}\right)+\sum_{i=1}^{n}\left(x_{ij}\times\alpha m_{i}\right)} \tag{12}$$

3.3 Problem Formulation

The heart of the cloud is Virtualization. When the data center gets the query from the user, a VM has to create and hosted on a server in relation to the given CPU, OS, and memory request. Assigning the VM to a different server is governed by the allocation or placement strategy, which is eminently accountable for mitigating resource wastage. This paper tracks VMA for minimalizing resource wastage in association with the servers' volume which is active. It is presumed that there exist n VMs characterized through the set V = {1,2,3…… n} and m servers or servers characterized through the

Table 1. Relinquishes the description for the notations being used in problem formulation along with the proposed method

Notations	Descriptions
V	VMs List and $\left\|V\right\|=n$
P	servers List and $\left\|P\right\|=m$
U_{p}^{cpu}	CPU utilization by server p in normalized form
U_{p}^{mem}	Memory utilization by server p in normalized form
T_{j}^{cpu}	Total CPU utilization by server j
T_{j}^{mem}	Total memory utilization of server j
TU_{j}^{cpu}	Total CPU utilization by an active server j
TU_{j}^{mem}	Total memory utilization by an active server j
NF_{p}^{cpu}	Residual CPU of server p in normalized form
NF_{p}^{mem}	Residual memory of server p in normalized form
$R_{cpu}^{wastage}$	Resource wastage of server p
αmi, αci	memory and CPU requirement for VM i
Tmj, Tcj	Thresholds for memory and CPU utilization corresponding to jth physical server
x_{ij}	A variable that defines whether or not the ith virtual machine is assigned to the jth physical server.
y_{j}	This variable shows whether or not the jth physical server is in use.

set P={1,2,3……m }. Let αm_i and αc_i represent memory and CPU requirement by all generated VMs i ∈V and Tm_j and Tc_j represent memory and CPU of all servers j ∈ P. Let x_{ij} and y_j be the binary variables which are being defined as:

$$x_{ij} = \begin{cases} 1 \ if VM\, i \forall i \in V \, is allocated to server \, j \forall j \in P \\ 0 \ otherwise \end{cases} \tag{13}$$

$$y_j = \begin{cases} 1 \ if the server \, j \forall j \in P \, is in use \\ 0 \ otherwise \end{cases} \tag{14}$$

The main motive of this works is to

$$\text{Minimize } f\left(X\right) = \sum_{j=1}^{m} R_{cpu}^{wastage} \tag{15}$$

Subject to following:

$$\sum_{j=1}^{m} x_{ij} = 1 \quad \forall i \in V \tag{16}$$

$$\sum_{i=1}^{n} x_{ij} \times \alpha c_i \leq Tc_j \times y_j \forall j \in P \tag{17}$$

$$\sum_{i=1}^{n} x_{ij} \times \alpha m_i \leq Tm_j \times y_j \forall j \in P \tag{18}$$

The function in equation 15 is expressed to reduce resource waste. Equation 16 restricts VM i is to be deployed to one server only. The restriction in equations 17& 18 ensures that the allocated VM does not drive beyond the extent of the server.

3.4 Solution Representation

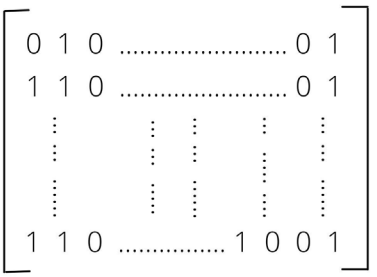
In BWOA each solution is represented by a decision variable x_{ij} which depicts the allotment of VM j on the server i. Each solution or whale in the VMA issue is represented by a binary matrix as represented through figure 3 with the condition as every VM should have been allotted to one and only one server.

3.5 Modification in BWOA for the VMA

A few modifications are required for applying the BWOA to VM allocation problems, which definitely needs to be performed.

1. *Initialization:* Initially, the whale population is initialized randomly within the binary search space so it's not necessary that the generated population will satisfy the constriction of Equations 16& 17. An initialization function is obligatory to initialize the desired population. At the commencement of the procedure, every VM is allocated to the server with the probability $1/m$. To allot the VM i to the server or server j a random array of $1Xn$ which contains the value between $\left[1, m\right]$ with no duplicates. The x_{ij} is set up as 1 in accordance with the random array, and if it satisfies the constraint, then it is allotted to the first server, otherwise to the second, and the process goes on.

Figure 3. Solution Representation



2. *Position update of the whales:* Each whale updates its position with respect to Equations 1, 5, and 8 in accord with the given random probability. After the update, the value for each position associated to the agent is squashed using equation 18. In all these updates, it might be possible the solution becomes infeasible by violating equation 16. To keep track of VMs, a vm_status array is implemented to restrict the update of the elements in column jth column in x_{ij} which obtained the value 1 in the calculated solution so far. At the start of every iteration, the vm_status array is initialized by 0 and before the updation of x_{ij} it is checked whether $vm\_status\left[j\right]$ has any value. If some value exists with it, then it demonstrates the VM j has already been allotted to the server from x_{ij} to x_{i-1j}. Then, x_{ij} is not updated, and it is only updated when $vm\_status\left[j\right]=0$ and $\alpha < S(x_i^j\left(k\right)$ then x_{ij} is set up as 1. The vm_status array has substantially increased the algorithm's effectiveness.
3. *Function Mapping:* The redeployment of VMs does not necessarily change the status of the server. The original BWOA is unexpected to converge to a globally optimal solution. If it does, the whales' position will be zero, increasing the likelihood of modifications in the associated bit. The search will be more random and lacking in direction. The original mapping does not look out for the VM allocation because if the whale's position leads to VM reallocation, then it is not necessary that the status of the server also changes. If server X is hosting VM a,b,c, and server Y is hosting VM d then both have status 1. If VM c is migrated from server X to Y then also their status will remain the same. Thus, to confirm the allocation problem, the sigmoid function needs updating, which is as follows:

$$F\left(x_i^j\left(k\right)\right)=\begin{cases}1-\dfrac{2}{1+e^{-S_i^j(k)}} & S_i^j\left(k\right)\le 0\\ \dfrac{2}{1+e^{-S_i^j(k)}}-1 & S_i^j\left(k\right)>0\end{cases} \tag{19}$$

4. *Redeployment of VM:* With the above information, one can determine the distance of whale in each generation and evaluate the modifications required in VM reallocation. When a bit of whale displacement transitions to 1, the related server remains unchanged; nevertheless, if a bit of whale displacement transitions to 0, the corresponding server requires some adjustment. A new list of VMs is being generated, which needs deployment on the server. In this work, BFD (Best Fit Decrease) is employed for the reallocation of VMs. This scheme helps in reducing the active server's volume to a great extent. Then the fitness is determined again, and the global best solution is obtained.

4. ALGORITHM

4.1 Data Structure

- **Whale[wnum]:** It specifies the total-sum size of whale's population present.
- **Iteration:** It defines the total iteration in BWOA.
- **Server_List:** It's the servers' listing that holds information about each one, such as its ID, CPU, and disc, including the volume of VMs assigned to it.
- **Vm_List:** It provides all the information associated with a VM.
- **RandWhale:** It stores the position linked with the given whale.
- **Wvalue[wnum]:** It provides the optimum resource wastage of the current whale in the given iteration.
- **G_best:** It stores the effective position of a whale in the population in the given iteration.
- **G_value:** It gives the minimum resource wastage fitness for the whale population in the given iteration.

The pseudocode for the suggested technique is displayed by algorithm 1 in figure 4. It depicts the overall procedure followed in allocation. The first step is to generate wnum number of whales, each of which has the same server list and VM list but each has its own deployment of VMs as their allocation on servers is random. Then among the listing given for various allocations, the global best allocation is determined through the given fitness function providing the minimum resource wastage. This provides initial preparation for the implementation of BWOA (lines 2-10). Next, the BWOA is utilized to update each whale's position to procure the optimal global allocation of VMs (lines 11-21). After that, the global optimal allocation of the whale is the desired VM allocation (line 22).

Algorithm 2 in figure 5 is responsible for the initial allocation of VMs on servers; first, a random array vmRand is generated which has a random value from 1 to m. Each VM selects the server with the probability 1/m, a server is allocated to a VM only if it has enough resources to accommodate it. The server's displacement is set to 1(from 0) if it does not already host any other VMs, indicating that the server has already been assigned to the VM. If the current server is insufficiently resourced, the algorithm will move on to the next server until it finds one.

Algorithm 3 in figure 6 depicts the process of upgrading each whale position during each iteration. It updates each bit of whale, which represents each server (lines 2-12). Then it applies the transition function on each server. The transition value obtained decides whether or not the provided server requires any adjustments. If the transition value is 0, the server is added to the list of servers that require VM redeployment (lines 13-18). The list of VMs is procured from the list and allocated with the BFD strategy on the servers. The information is updated in the serverList (lines 19-22).

5. SIMULATION

The proposed procedure's effectiveness has been assessed through simulation experiments. The simulation's output is contrasted with other algorithms. The simulation for the framework has been enforced on Cloudsim using the Eclipse platform and runs on a PC Intel Core CPU i3-7020U and 4.0 GB Ram and Windows 10 Home as the operating system. Each experiment is performed with a similar VMs number and servers. The start-up population of whales is initialized by 50, and maximum iterations are initialized to 100. To ensure the methodology's applicability, the proposed server configuration has been kept equivalent to the one available in the market. The approach is analyzed against the Genetic Algorithm (GA), First Fit Algorithm (FFA), and Best Fit Decrease algorithm (BFD). FFA is a VM allocation bin packing method in which a VM is designated to the first available server with all of the needed resources. BFD is the modified form of the Best First algorithm, which first arranges VMs in the decreasing order of their requisitioned resources, and then it is allocated to the most fitted available

Figure 4. Algorithm 1 for VMA according to BWOA

```
Algorithm-1

Input: vm_List, wnum, Iteration, server_List
Output: G_best
(1)  G_best = Null, G_value = ∞
(2)  for j in wnum
(3)      server_List = RandomVMallocation (vm_List)
(4)      Whale[j] = server_List
(5)      Wvalue[j] = fitness (server_List)
(6)      if (G_value > Wvalue[j])then
(7)          G_best = Whale[j]
(8)          G_value = Wvalue[j]
(9)      end if
(10) end for
(11) while Iteration >0
(12) for j in wnum
(13)     Whale[i] = BWOA (server_List, Whale[j])
(14)     Wvalue[i]= fitness (Whale[j])
(15)     if (G_value > Wvalue[j]) then
(16)       G_best = Whale[j]
(17)       G_value = Wvalue[j]
(18)     end if
(19) end for
(20)     Iteration = Iteration -1
(21) end while
(22) return G_best
```

Figure 5. Algorithm 2 for Random initial allocation of VM

```
Algorithm-2

Input : vm_List
Output: server_List
(1) x=zeros(m,n)
(2) vmRand=random(n,1)//Generates a nx1 vector with all values distributed randomly between [1,m]
(3) for server i in server_List
(4)   for vm j  in vm_List
(5)   if(server i has sufficient resources to deploy vm j) then
(6)       server.add(vm)
(7)       if(server is empty)
(8)         server.setDisplacement(1)
(9)       end if
(10)      x[vmRand[i],i]=1 // Allot vm j to server i
(11)  end if
(12)  end for
(13)end for
```

server with all the resources needed. Both algorithms follow the greedy approach to furnish the solution. GA is the metaheuristic population-based approach working closely with the concept of survival of the fittest and reproduction. To develop the solution, it selects the best individual from a given population and performs crossover and mutation. The execution of the algorithms on the simulator is represented in figure 7, figure 8, figure 9, and figure 10. The inference of the result is as follows:

5.1 Resource Wastage fitness

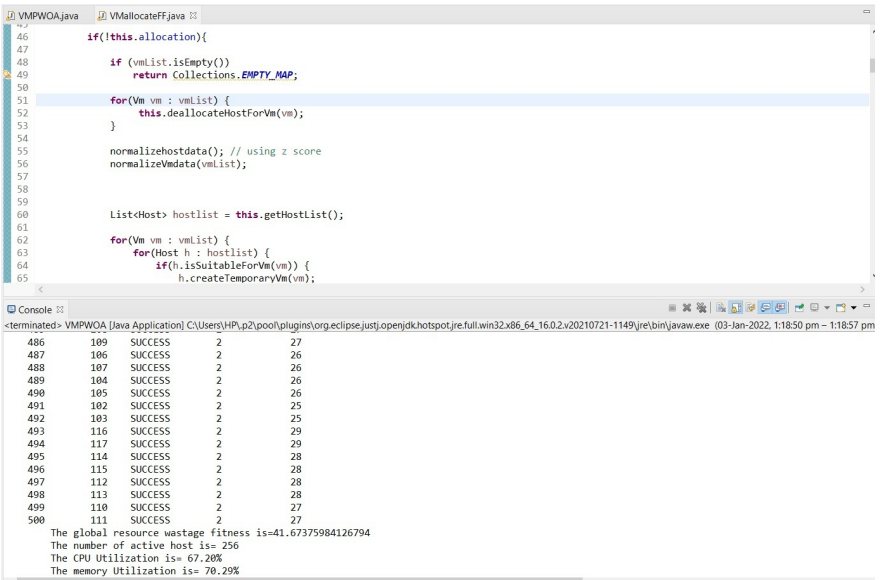
The proposed method is compared with the three available algorithms in correspondence with resource wastage. Figure 11 shows the resource wastage achieved from the given four algorithms under the condition that different numbers of VMs and servers are deployed. Table 2 represents the statistical data of the same. The iteration parameter is initiated to 40, and the BWOA population size is also set to 50. The outcome accomplishes that the anticipated approach can get the minimal resource

Figure 6. Algorithm 3 for BWOA

Algorithm-3

Input:*vm_List, server_List, Whale*
Output:server_List
(1) Update *B, H, p* and *l*
(2) **if**(p<0.5)
(3) **if**(|*B*|<1)
(4) Whale's position is updated according to Eq. 1
(5) **else if**(*B*|≥1)
(6) RandWhale , A random whale chosen from the list
(7) Whale's position is updated according to Eq.8
(8) **end if**
(9) **end if**
(10) **else if**(p≥0.5)
(11) Whale's position is updated according to Eq.5
(12) **end if**
(13) Update the whale according to Eq. 18
(14) **for** server in ServerList
(15) S = Whale.getServer(server.getId()).getDisplacement()
(16) S'= S.getTransition()
(17) **if**(S=0) then
(18) ServerListRedeploy.add(server)
(19) **endif**
(20) **end for**
(21) MigratedVMList = RedeployServerList.getAllVms()
(22) Reallocate VMs from MigratedVMList to the servers in BFD order.
(23) Update the vm_satus in server_List according to the latest reallocation.
(24) **return** server_List

Figure 7. FF representation



wastage against the three other algorithms in all the servers and VMs. It can be observed from the figure the FF algorithm shows the maximum resource wastage, and as the VM is scaled up to 500, the wastage is maximum. The BFD algorithm has performed better than FF but has as compared to GA and BWOA its efficiency is less. It can be seen from the table the BWOA has reduced the resource wastage by 62% approximately as compared to FF and this reduction is 36% as against BFD. The GA has also shown a comparable result to BWOA; unfortunately, its efficiency is less than BWOA. The control of the resource wastage is mainly because the adaptive variation of the search agent has led to a smooth transition between the exploration and exploitation phase. Moreover, the approach proposed globally optimizes, ensuring the minimum resource wastage. Whereas the FFA and BFD

Figure 8. BFD representation

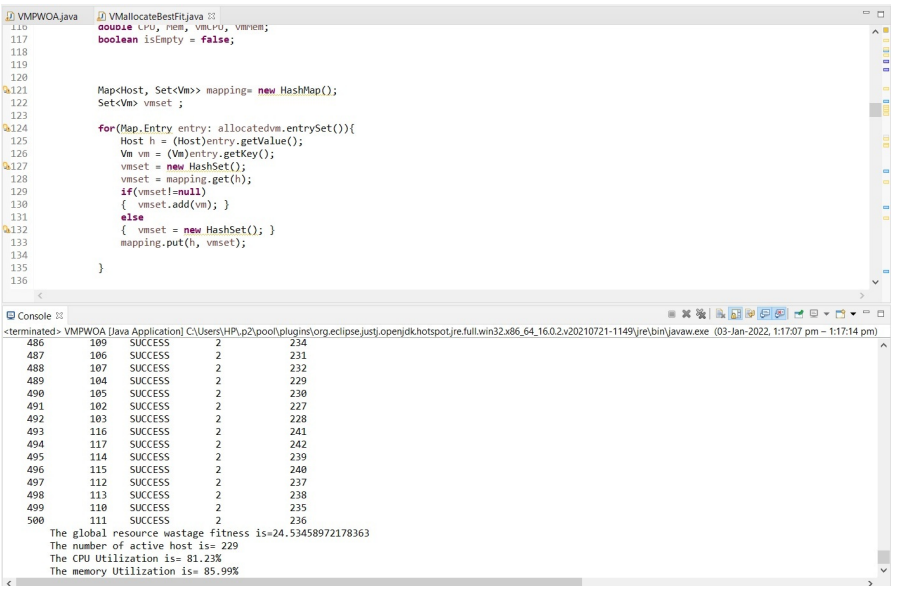
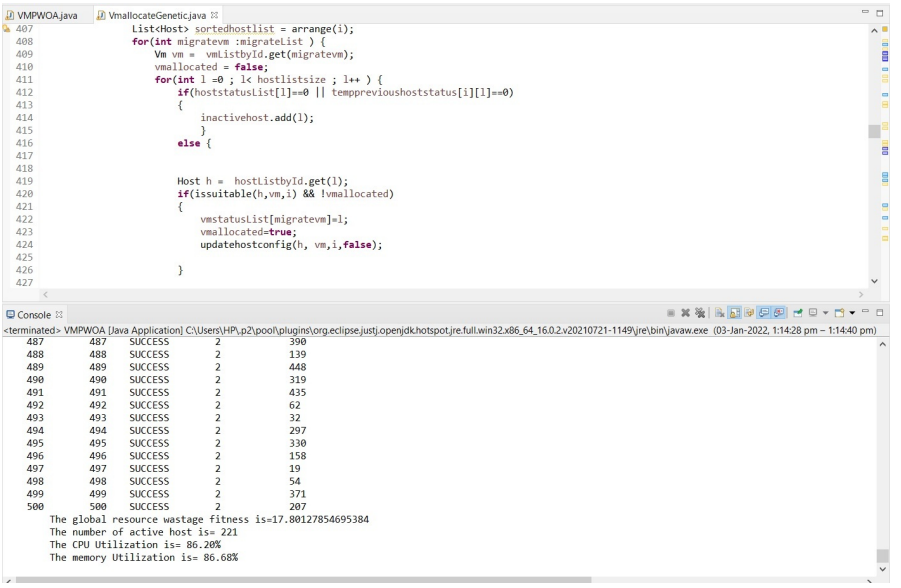


Figure 9. GA representation



heuristic algorithms perform the operation in a greedy manner that lacks such global optimization, leading to greater resource wastage.

5.2 Number of Active Servers

Besides resource utilization, the other major parameter which defines the efficaciousness of VMA is mostly the volume of working servers. The VMA has shown more effectiveness while there are lesser active servers. The system is configured in a similar way as before, only the iteration parameter is 50,

Figure 10. BWOA representation

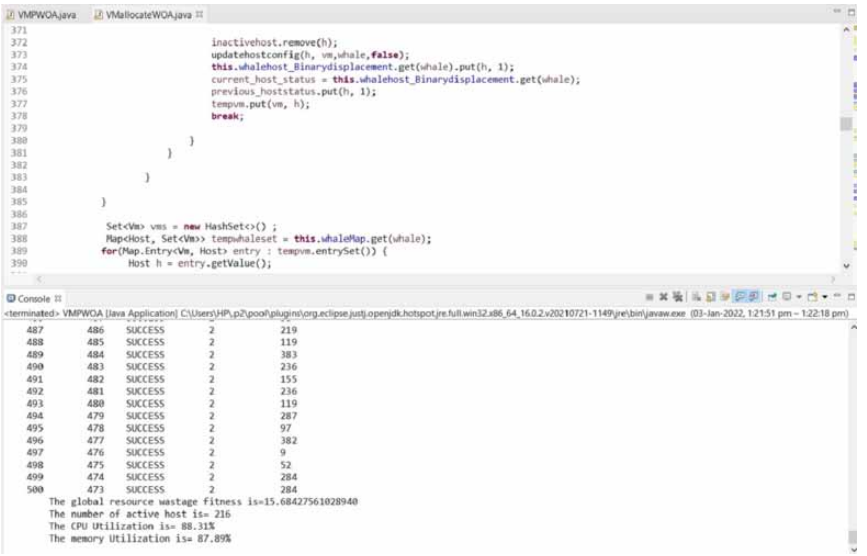


Table 2. Comparison of Resource Wastage Fitness

	Number of Virtual Machines				
Techniques	**100**	**200**	**300**	**400**	**500**
FF	11.0015	16.1356	25.1247	34.4372	41.6737
BFD	6.6825	9.8961	14.3863	19.8849	24.5345
GA	4.6637	7.4562	12.6731	14.9631	17.8013
BWOA	3.3410	5.0827	9.8816	12.4574	15.6842

and the population is 50. As can be observed from figure 12, the suggested methodology based on BWOA creates lesser active servers than other algorithms. It can be inferred from the figure in the case of 100VMs that all the methods have created a comparable number of active servers, and as the number of VMs increases, FF is consuming more servers compared to the other approaches. BFD and GA have shown quite comparable consumption of servers. Table 3 shows that when the number of VMs is less, BWOA creates 19% less active servers compared to FF, and when the number of VMs is increased to 500, it creates 15% less active servers. In BWOA, 216 active servers are utilized for allocating the VMs, which is the least compared to BFD and GA, which used 229 and 221 servers, respectively. It is also inferred that as VMs grow in number, so does the algorithm's performance.

5.3 Memory & CPU Utilization

Memory and CPU utilization is the memory or CPU utilized by the VMs of the working server. It is given as follows:

$$TU^{CPU} = \frac{1}{\eta}\sum_{\lambda=1}^{\eta}\left[\frac{\sum_{i=1}^{n}\left(x_{i\lambda} \cdot \alpha c_i\right)}{U_p^{cpu}}\right] \tag{20}$$

Figure 11. Resource Wastage Fitness

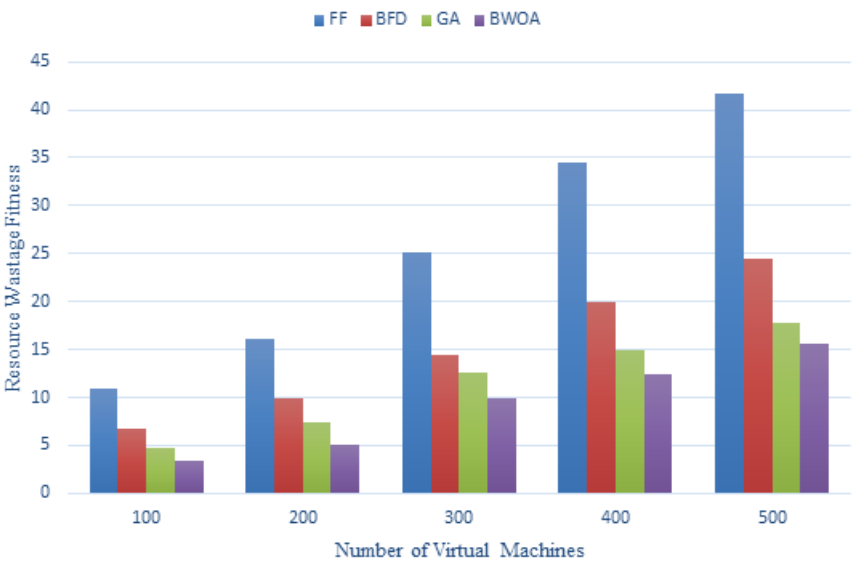
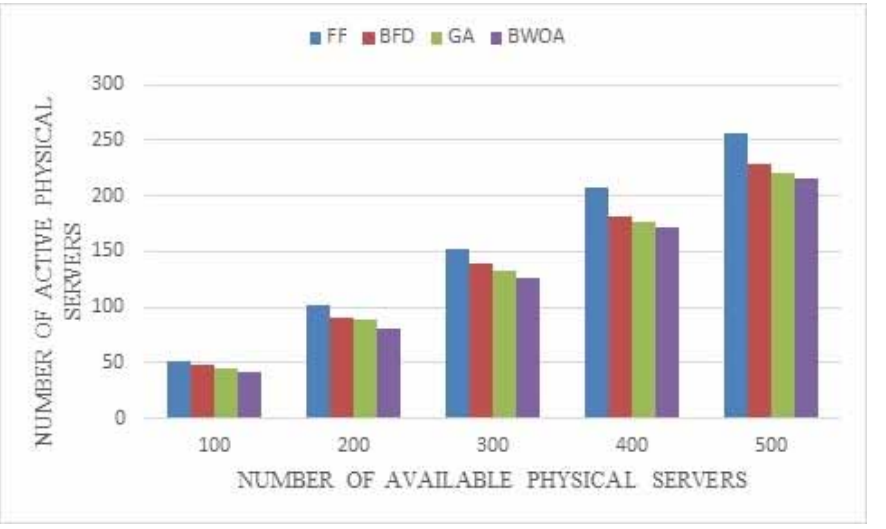


Table 3. Comparison of the Number of Active Servers

	Number of Virtual Machines				
Techniques	**100**	**200**	**300**	**400**	**500**
FF	52	102	153	208	256
BFD	48	91	139	181	229
GA	45	89	133	177	221
BWOA	42	81	126	171	216

Figure 12. Number of Active Physical Servers



$$TU^{mem} = \frac{1}{\eta}\sum_{\lambda=1}^{\eta}\left[\frac{\sum_{i=1}^{n}\left(x_{i\lambda} \cdot \alpha m_i\right)}{U_p^{mem}}\right] \tag{21}$$

Here η is the total sum volume of servers running VMs on it.

It is evident from figures 13 and 14 and table 4 and table 5 the method proposed has distributed the resources to VMs in a balanced manner making the uttermost CPU utilization and memory utilization. It can be concluded from figure 13 that FF algorithm's memory utilization is the least, while BFD and GA have shown a quite comparable results in terms of memory utilization. BWOA has shown the maximum utilization of memory. Table 4 shows that 87.89% of memory is utilized through the proposed approach, which is the maximum, while FF has consumed 70.29%, which is the least. It can also be inferred that as the number of VMs increases, memory utilization has also improved.

Figure 14 shows the CPU utilization by the proposed approach and other approaches. The least utilization of CPU can be observed in FF algorithm, while BFD and GA utilization of CPU is greater than FF but less than BWOA. BWOA utilizes the CPU most efficiently as compared to others. It can be concluded from table 5, BWOA uses the CPU by 88.31%, which is the maximum as compared to others. FF does the minimum utilization of 67.24% in the 500 VMs scenario.

5.4 Iterations and Population

The following analysis is performed on the iteration scale. With a similar scenario as discussed before and the VMs volume as 50 and the population as 50, resource wastage efficiency is observed. Figure 15 demonstrates that as the iteration value escalates, resource wastage fitness becomes more stable, and as the iteration goes beyond 200, the fitness resource wastage becomes almost consistent.

The whale population size is another crucial element that significantly affects the algorithm's efficacy. The algorithm's efficiency, convergence, and resource waste value are all affected because of the population's small size. Figure 16 illustrates that resource wastage fitness remains stable as

Table 4. Comparison of Memory utilization

	Number of Virtual Machines				
Techniques	**100**	**200**	**300**	**400**	**500**
FF	70.11	70.23	70.52	70.14	70.29
BFD	84.01	85.21	85.32	85.11	85.99
GA	85.23	85.56	85.51	86.89	86.68
BWOA	86.11	86.27	86.58	87.45	87.89

Table 5. Comparison of CPU utilization

	Number of Virtual Machines				
Techniques	**100**	**200**	**300**	**400**	**500**
FF	65.40	65.61	65.86	66.52	67.24
BFD	79.12	80.23	80.68	80.99	81.47
GA	85.10	85.51	85.57	85.99	86.20
BWOA	87.23	87.56	87.89	88.20	88.31

Figure 13. Memory Utilization

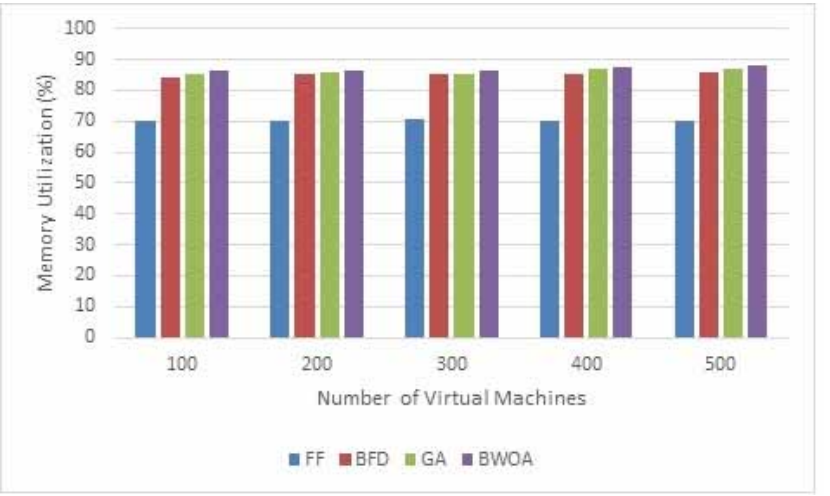
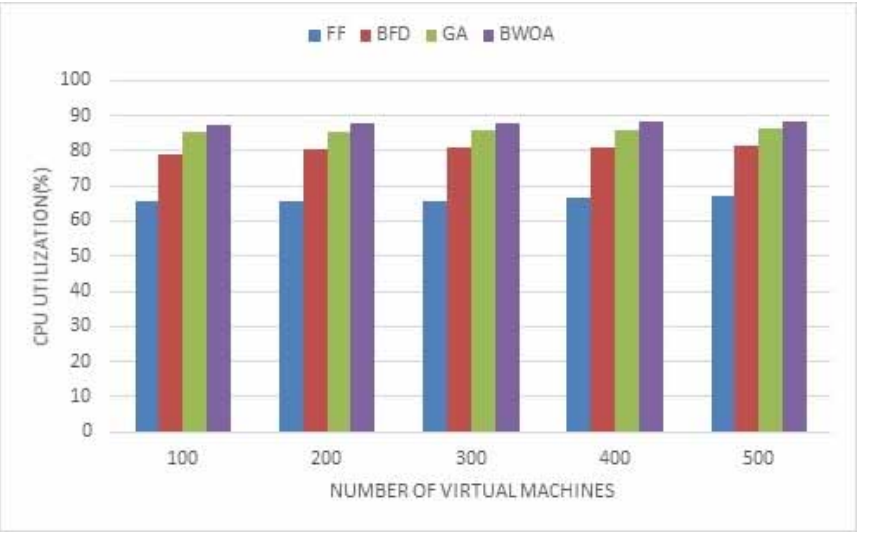


Figure 14. CPU Utilization



the population grows. Thus, it can be inferred that on the larger scale of the population, consistency in the result can be obtained.

6. DISCUSSION

Cloud computing is a technology that has gained widespread acceptance in industries ranging from business to research. The field requires a lot of research for its full and enhanced utilization. In such an infrastructure, VM allocation plays a crucial role in efficiently using all resources. Several existing policies have considered energy, bandwidth, memory and CPU utilization, Response time, and execution time. Very few works have been done, taking resource wastage and active servers' volume as the main consideration. Resource wastage and active servers' volume have been taken

Figure 15. Number of Iterations (50 VMs)

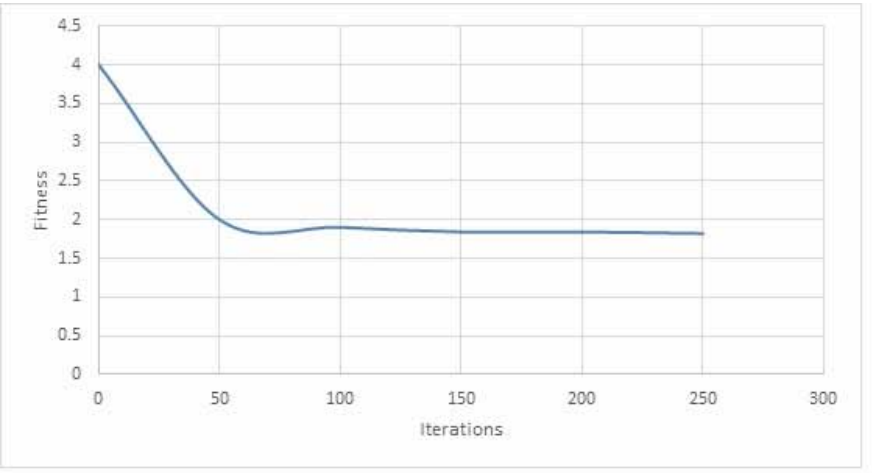
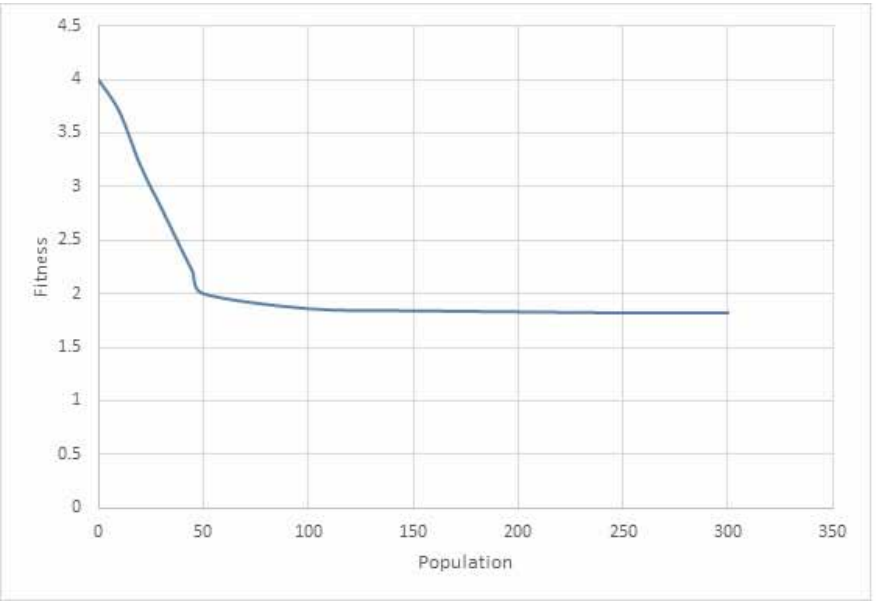


Figure 16. Population Size (50 VMs)



into account to perform an effective allocation in the study. Furthermore, the computed results were compared to FF, BFD, and GA, and it was discovered that the computed results were more efficient.

The work's significance can indeed be summarized as:

- The VM's allocation is heavily influenced by resource use, such as CPU usage, the number of active servers, and memory usage. This has the effect of lowering the cost and energy of the allocation policy.
- The work has adopted BWOA to allocate the VM to the server. This strategy is discrete and applies exploration and exploitation abilities. It easily transitions between exploration and exploitation owing to its adaptive variation, resulting in faster convergence and better results.

- In addition, BFD was employed to accomplish the migration to minimize the number of active servers drastically.
- The outcome of the research helps in efficiently allocating VMs on servers.
- The findings of the study will aid in allocating the VMs such that wastage of resources is highly reduced and the servers' active volume is also minimized to a large extent.
- This further helps in the efficient usage of resources, reducing the energy consumption and amount of resources required and thus reducing the cost of resources and infrastructure.

The overall contribution illustrates the assessment of VMs allocation in the cloud environment and is significant. There are still certain limitations that can be addressed in future efforts. The following are some of them:

- More parameters can be included in addition to the identified ones.
- The results may vary with the number of factors evaluated in the fitness function.
- The cloud metric contemplated in the research proposed is the servers' quantity in active mode and resource wastage, this might be stretched to other metrics like execution time, traffic diversion, SLA violation, and the impact of their collocation on the efficiency.

The overall objective of the research initiative is to minimize resource wastage and active servers' volume with respect to CPU and memory configuration. The outcome of this work will help researchers and cloud practitioners address the allocation issue. The obtained result is verified by comparing them with other conventional approaches.

7. CONCLUSION

This paper enumerates an innovative strategy for optimizing VM allocation with diverse data centers, aiming to lower the volume of servers in active mode and mitigate system resource waste. A modified BWOA is applied where VMs allocation has been performed in the binary space. Firstly, the population is initialized randomly, and the search agent's position is updated for every iteration, which is transformed into binary using a sigmoid function. Gradually, it converges to deliver a globally optimized solution. The framework's performance suggested is validated through simulation. Besides, it provides an optimized solution for memory and CPU consumption. The implication of other relevant parameters on the algorithm's efficiency is also discussed. The feasibility of the effort is asserted based on the implementation analysis and results obtained. The algorithm's capabilities can be extended by including more metrics like execution time, bandwidth, SLA violation, and security as future work. The algorithm can be applied to other cloud issues like task scheduling, load balancing, and dynamic migration.

CONFLICTS OF INTEREST

The authors of this publication declare there is no conflicts of interest.

FUNDING STATEMENT

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors. Funding for this research was covered by the author(s) of the article.

REFERENCES

Abdessamia, F., Zhang, W. Z., & Tian, Y. C. (2020). Energy-efficiency virtual machine placement based on binary gravitational search algorithm. *Cluster Computing*, *23*(3), 1577–1588. doi:10.1007/s10586-019-03021-0

Abohamama, A. S., & Hamouda, E. (2020). A hybrid energy–aware virtual machine placement algorithm for cloud environments. *Expert Systems with Applications*, *150*, 113306. doi:10.1016/j.eswa.2020.113306

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

Barlaskar, E., Singh, Y. J., & Issac, B. (2018). Enhanced cuckoo search algorithm for virtual machine placement in cloud data centres. *International Journal of Grid and Utility Computing*, *9*(1), 1–17. doi:10.1504/IJGUC.2018.090221

Beloglazov, A., & Buyya, R. (2010). Adaptive threshold-based approach for energy-efficient consolidation of virtual machines in cloud data centers. *MGC@ Middleware, 4*(10.1145), 1890799-803.

Chaisiri, S., Lee, B. S., & Niyato, D. (2009, December). Optimal virtual machine placement across multiple cloud providers. In 2009 IEEE Asia-Pacific Services Computing Conference (APSCC) (pp. 103-110). IEEE. doi:10.1109/APSCC.2009.5394134

Chen, H., Xu, Y., Wang, M., & Zhao, X. (2019). A balanced whale optimization algorithm for constrained engineering design problems. *Applied Mathematical Modelling*, *71*, 45–59. doi:10.1016/j.apm.2019.02.004

Dubey, K., & Sharma, S. C. (2020). An extended intelligent water drop approach for efficient VM allocation in secure cloud computing framework. *Journal of King Saud University-Computer and Information Sciences*.

Gawali, M. B., & Shinde, S. K. (2018). Task scheduling and resource allocation in cloud computing using a heuristic approach. *Journal of Cloud Computing*, *7*(1), 1–16.

Ghetas, M. (2021). A multi-objective Monarch Butterfly Algorithm for virtual machine placement in cloud computing. *Neural Computing & Applications*, *33*(17), 1–15. doi:10.1007/s00521-020-05559-2

Gohil, B. N., Gamit, S., & Patel, D. R. (2021). Fair Fit—A Load Balance Aware VM Placement Algorithm in Cloud Data Centers. In *Advances in communication and computational technology* (pp. 437–451). Springer. doi:10.1007/978-981-15-5341-7_35

Hussien, A. G., Hassanien, A. E., Houssein, E. H., Amin, M., & Azar, A. T. (2020). New binary whale optimization algorithm for discrete optimization problems. *Engineering Optimization*, *52*(6), 945–959. doi:10.1080/0305215X.2019.1624740

Kaaouache, M. A., & Bouamama, S. (2018). An energy-efficient VM placement method for cloud data centers using a hybrid genetic algorithm. *Journal of Systems and Information Technology*, *20*(4), 430–445. doi:10.1108/JSIT-10-2017-0089

Li, Q., Hao, Q. F., Xiao, L. M., & Li, Z. J. (2011). Adaptive management and multi-objective optimization for virtual machine placement in cloud computing. *Jisuanji Xuebao*, *34*(12), 2253–2264. doi:10.3724/SP.J.1016.2011.02253

Lo, V. M. (1983). *Task assignment in distributed systems* [Doctoral dissertation]. University of Illinois at Urbana-Champaign.

Mirjalili, S., & Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software*, *95*, 51–67. doi:10.1016/j.advengsoft.2016.01.008

Pahlevan, A., Qu, X., Zapater, M., & Atienza, D. (2017). Integrating heuristic and machine-learning methods for efficient virtual machine allocation in data centers. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, *37*(8), 1667–1680. doi:10.1109/TCAD.2017.2760517

Peng, Y., Kang, D. K., Al-Hazemi, F., & Youn, C. H. (2017). Energy and QoS aware resource allocation for heterogeneous sustainable cloud datacenters. *Optical Switching and Networking*, *23*, 225–240. doi:10.1016/j.osn.2016.02.001

Prakash, D. B., & Lakshminarayana, C. (2017). Optimal siting of capacitors in radial distribution network using whale optimization algorithm. *Alexandria Engineering Journal*, *56*(4), 499–509. doi:10.1016/j.aej.2016.10.002

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*, *12*(3), 16–38. doi:10.4018/IJACI.2021070102

Rankothge, W., Le, F., Russo, A., & Lobo, J. (2017). Optimizing resource allocation for virtualized network functions in a cloud center using genetic algorithms. *IEEE eTransactions on Network and Service Management*, *14*(2), 343–356. doi:10.1109/TNSM.2017.2686979

Saravanan, D., Rajakumar, R., Sreedevi, M., Dinesh, K., Sudha, S. V., Anguraj, D. K., & Mubarakali, A. (2021). Multi-objective swarm-based model for deploying virtual machines on cloud physical servers. *Distributed and Parallel Databases*, 1–19. doi:10.1007/s10619-021-07341-2

Savitha, S., & Salvi, S. (2021, April). Perceptive VM Allocation in Cloud Data Centers for Effective Resource Management. In *2021 6th International Conference for Convergence in Technology (I2CT)* (pp. 1-5). IEEE. doi:10.1109/I2CT51068.2021.9417960

Saxena, D., & Singh, A. K. (2021). A proactive autoscaling and energy-efficient VM allocation framework using online multi-resource neural network for cloud data center. *Neurocomputing*, *426*, 248–264. doi:10.1016/j.neucom.2020.08.076

Shabeera, T. P., Kumar, S. M., Salam, S. M., & Krishnan, K. M. (2017). Optimizing VM allocation and data placement for data-intensive applications in cloud using ACO metaheuristic algorithm. *Engineering Science and Technology, an International Journal, 20*(2), 616-628.

Too, J., Mafarja, M., & Mirjalili, S. (2021). Spatial bound whale optimization algorithm: An efficient high-dimensional feature selection approach. *Neural Computing & Applications*, *33*(23), 16229–16250. doi:10.1007/s00521-021-06224-y

Toutov, A., Toutova, N., Vorozhtsov, A., & Andreev, I. (2021, January). Multicriteria Optimization of Virtual Machine Placement in Cloud Data Centers. In *2021 28th Conference of Open Innovations Association (FRUCT)* (pp. 482-487). IEEE. doi:10.23919/FRUCT50888.2021.9347607

Usman, M. J., Ismail, A. S., Chizari, H., Abdul-Salaam, G., Usman, A. M., Gital, A. Y., & Aliyu, A. (2019). Energy-efficient virtual machine allocation technique using flower pollination algorithm in cloud datacenter: A panacea to green computing. *Journal of Bionics Engineering*, *16*(2), 354–366. doi:10.1007/s42235-019-0030-7

Vakilinia, S., Heidarpour, B., & Cheriet, M. (2016). Energy efficient resource allocation in cloud computing environments. *IEEE Access: Practical Innovations, Open Solutions*, *4*, 8544–8557. doi:10.1109/ACCESS.2016.2633558

Widmer, T., Premm, M., & Karaenke, P. (2013, February). Energy-aware service allocation for cloud computing. In *Proceedings of the International Conference on Wirtschaftsinformatik* (pp. 1147-1161). Academic Press.

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, *1*(1), 67–82. doi:10.1109/4235.585893

Xia, W., & Shen, L. (2018). Joint resource allocation using evolutionary algorithms in heterogeneous mobile cloud computing networks. *China Communications*, *15*(8), 189–204. doi:10.1109/CC.2018.8438283

Zhang, X., Wu, T., Chen, M., Wei, T., Zhou, J., Hu, S., & Buyya, R. (2019). Energy-aware virtual machine allocation for cloud with resource reservation. *Journal of Systems and Software*, *147*, 147–161. doi:10.1016/j.jss.2018.09.084

Ankita Srivastava completed her B.Tech in IT and M.Tech in CS from Dr. A.P.J. Abdul Kalam Technical University, Lucknow, UP, India. Currently, she is pursuing her PhD in Computer Science from Babasaheb Bhimrao Ambedkar University (A Central University), Lucknow, UP, India. Her research interests are Cloud Computing, Nature-Inspired Metaheuristics, Optimization Techniques.

Narander Kumar received his Post Graduate Degree and Ph. D. in CS & IT, from the Department of Computer Science and Information Technology, Faculty of Engineering and Technology, M. J. P. Rohilkhand University, Bareilly, Uttar Pradesh, INDIA in 2002 and 2009, respectively. His current research interest includes Quality of Service (QoS), Cloud Computing, Computer Networks, Resource Management Mechanism, in the networks, Performance Evaluation. Presently he is working as Assistant Professor, in the Department of Computer Science, Babasaheb Bhimrao Ambedkar University (A Central University), Lucknow, UP, India.