Multi-objective Meta-heuristic Technique for Energy Efficient Virtual Machine Placement in Cloud Computing Data Centers

C.Vijaya[1], P. Srinivasan[2*]
[1]SCOPE, VIT University, Vellore, India,
[2]SCORE, VIT University, Vellore, India.
E-mail: c.vijaya2016@vitstudent.ac.in, srinivasan.suriya@vit.ac.in
*Corresponding author

Keywords: cloud computing, virtualization, virtual machine placement, ant colony optimization, sine cosine algorithm

Received: October 8, 2023

Cloud computing has emerged as an efficient and scalable solution for storing and processing a large amount of data. Cloud data centers provide resources on demand to consumers on a pay-per-use model. However, a large number of data centers are required to support the growing demand of cloud consumers. This needs to be handled in an optimized way to avoid resource wastage and ensure that more consumers can benefit from data centers. Virtualization is the technology of creating virtual versions of computers called Virtual Machines (VMs). The Virtual Machine Placement problem is a fundamental challenge in cloud computing, where the goal is to determine the optimal allocation of Virtual Machines to Physical Machines (PMs) within a data center. An efficient Virtual Machine Placement technique helps properly place VMs on PMs, significantly optimizing the number of servers, maintenance costs, CPU utilization, and power consumption. We present a novel hybrid approach that combines the Ant Colony Optimization (ACO) algorithm and the Sine Cosine Algorithm (SCA) for efficient VM placement. Since SCA is an emerging search algorithm that utilizes Sine and Cosine functions in the engineering field, it has been used to explore the solutions obtained by the ACO algorithm. The ACO algorithm has been applied to exploit the solutions of the search space for efficient VM placement, aiding in power management and minimizing resource wastage. The results have been verified by comparing the performance against other algorithms to prove that our proposed algorithm outperforms them.

Povzetek: Predlagan je hibridni pristop, ki združuje algoritma ACO in SCA za optimizacijo postavitve virtualnih strojev v oblaku, kar zmanjšuje porabo energije in izboljšuje izkoriščenost virov.

1 Introduction

Cloud is an infinite resource pool that leverages resources to multiple users. From the customer or user point of view, it is a scalable service where consumers can access as many resources as needed, based on the concept of pay-as-you-go or metered services. From the provider's side, there is a business angle where they need to optimize or maximize their profit without compromising on the quality of services or violating SLAs. If there is any violation of an SLA, there are long-term implications and penalties to be paid. Cloud providers like Amazon, Google, IBM, Yahoo, and Microsoft have huge data centers across the world. Resource types are classified into two categories: physical resources (computers, disks, databases, networks, and scientific instruments) and logical resources (execution, monitoring, application communication). Currently, it is estimated that physical servers consume 0.5% of the world's total electricity usage. A data center has two major components [1]:

a. Compute infrastructure (Servers, Storage and Network etc.)
b. Logistics or environmental infrastructure. (UPS, AC, overall logistics setups used to house datacenter).

There is huge amount of separate power required to manage both the components. By summing the power consumed by compute infrastructure and logistics infrastructure, 1% of total power consumed by the world is consumed by data centers. Server energy demand doubles every 5 to 6 years. So, there is a large need of power consumption so that we need to minimize the utilization of power. Majority of energy sources are fossil fuels. By burning fossil fuels, huge volume of carbon-di-oxide is emitted every year from the power plants. Sustainable energy sources are data centers. So, there is a need to reduce the power consumption of data centers and also to reduce energy consumption with minimal performance impact. Load Balancing [42] and Server Consolidation [13] are the two most efficient methods to improve the energy efficiency. According to Amazon Web services, the most important components of server downtime are power supply, Virtual Machines and storage. In paper [2], a hybrid technique called Fuzzy

HAGA algorithm has been adopted for consolidating the servers using Ant Colony Optimization and Grey Wolf Optimization for exploiting and exploring the active servers and thereby reducing the number of physical machines. The results prove that the Fuzzy HAGA algorithm performs better server consolidation than the ACS, FFC, MMAS and FFD [4]. The average power consumption and resource wastage has been minimized. Considering the Virtual machine Placement problem, Sine Cosine Algorithm performs better in searching for the solutions obtained by ACO algorithm.

1.1 Server consolidation

Organizations can reduce hardware costs, simplify management, and improve resource utilization by consolidating servers. The process of server consolidation has a significant impact on the placement of virtual machines. Proper placement of virtual machines is crucial to ensure efficient utilization of consolidated servers and achieve the goal of maximizing resource utilization and minimizing hardware costs. Tahe strategy is to schedule as many virtual machines as possible on selected multi-core nodes, considering the capacity and requirements of the nodes. Regardless of whether a virtual machine is running on a server, it consumes a certain amount of energy. In Fig. 1, the nodes consume 105 Watts when idle and an incremental amount of energy when running. For example, node-1 is fully loaded and consumes 170 Watts, while nodes 2, 3, and 4 are running 2 virtual machines each and consume 138 Watts (105 + incremental energy). If all nodes contain only 2 virtual machines, the total power consumed would be 138*4 = 552 Watts. However, by consolidating all the virtual machines to run on server node 1, the total power consumed is reduced to 485 Watts. This reduction is due to server node 1 consuming less energy in both fully loaded and idle states compared to the combined consumption of nodes 2, 3, and 4. In Fig. 2, node 1 consumes 170 Watts, while nodes 2, 3, and 4 consume 105 Watts each, resulting in a total power consumption of 485 Watts. This scenario demonstrates that consolidating virtual machines on servers minimizes resource wastage and leads to power savings. The process of selecting the appropriate physical machine for moving a virtual machine is known as the virtual machine placement problem. Virtual machines are migrated from one physical machine to another when the physical machine is either overloaded or under loaded. It has been observed that cloud servers often do not utilize servers to their maximum capacity. Therefore, reducing the number of servers in an organization is essential to prevent server sprawl, which refers to the inefficient use of underutilized servers that consume excessive space, resources,

So, the use of server consolidation is widespread in order to reduce energy consumption. The primary goal of the server consolidation issue is to decrease the number of active servers required. These servers are necessary for hosting the virtual machines that users request. We are dealing with 'n' Virtual Machines (VMs) that need placement on 'm' Physical Machines (PMs). A Physical Machine, which is also referred to as a Server machine, is a computer system dedicated to running applications, processing data, and performing tasks without virtualization. A Virtual Machine is a virtual or software emulation of a physical machine. None of the VMs have

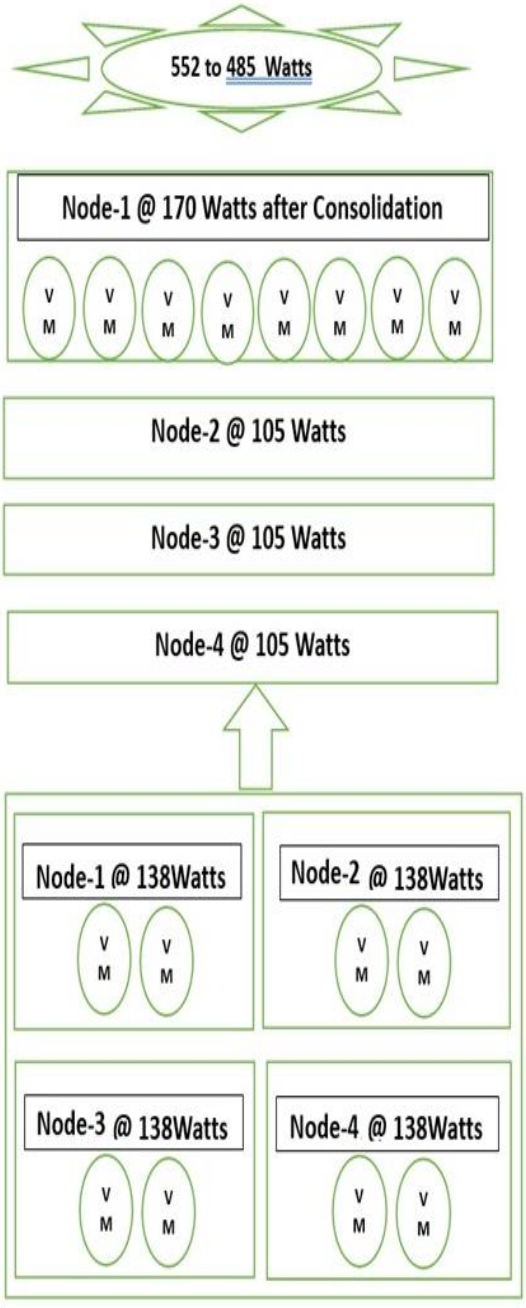

Figure 1: Power saving on server consolidation from 552 Watts to485 Watts

a capacity greater than any of the PMs. The key objective of the server consolidation problem is to minimize the number of active server machines needed for VM placement. Consider ‘n’ Virtual Machines (VMs) to be placed on ‘m’ Physical Machines (PMs). No VM is larger than any PM in capacity.D_{my} represents the CPU requirement of each VM and $T_{pr} and T_{my}$ represents full

capacity of a single physical machine. The proposed algorithm is going to reduce the wastage of resources and wastage of CPU. When the physical machine is utilized fully, then the performance of the physical machine may get degraded. So, we have an upper bound of 90 percent (fuzzy). We define two decision variables allocation matrix and binary variables. If a VM is allocated to a particular server j, then allocation matrix is set to 1 or it is set to 0, otherwise. A binary variable is used to denote whether a server is busy or not. The balance resources on each Physical Machine may vary based on the Virtual Machine placement algorithm. For all the resources to be utilized efficiently, the total cost of the wasted resources should be calculated thoroughly.

1.2 Server resource wastage modeling

Different virtual machine placement solutions can result in a significant variation in the remaining resources on each server. Multidimensional resources should be completely utilized. Therefore, the cost of wasted resources is evaluated by the below equation:

$$W_j = \frac{\left|L_j^{pr} - L_j^{my}\right| + \Delta}{\left|U_j^{pr} - U_j^{my}\right|} \quad (1)$$

W_j represents the resource wastage in j^{th} server. $U_j^{pr} - U_j^{my}$ represents the CPU and memory usage normalized in a physical machine. It is the ratio between used resources to the total resources available. $L_j^{pr} - L_j^{my}$ represents remaining resources in terms of CPU and memory. Δ is a small positive integer to avoid the capacity of the physical machine coming down to zero and it is set to 0.0001.

2 Related work

Virtualization is the most challenging research topic in cloud computing. The Virtual Machines are mapped to the suitable Physical Machine in datacenters [3]. R. Panigrahy et.al., have investigated the placement of one dimensional VM placement algorithm [4]. The Physical Machines should give proper support to the Virtual Machines to run the data center efficiently [5] considering multi-objectives. Many Researchers have studied the methods using the Metaheuristic Algorithms in the cloud computing environment. But these algorithms largely focus on initial placement of the VMs. Virtual Machine placement should focus on power saving and abide by the Service Level Agreement to provide Quality of service to the users [6]. There are different types of Algorithms used to place Virtual Machines for efficient Power management and Resource utilization which provides the solution for optimal placement. A survey of related problems have been studied and presented in this section.

D.Alsadie [7] in his review paper has discussed about the Meta-heuristic Algorithms used for Virtual Machine Placement such as Genetic based approaches, ACO based approaches [17],[28][26][31], BBO based approaches, PSO based approaches [41], Memetic approaches [8], Artificial Bee Colony approaches, Firefly based Algorithms and hybrid methods. In his review, it has been concluded that the hybrid meta-heuristic approaches are going to be the promising research direction in solving Virtual Machine Placement Problem.

Many researchers have studied the methods using the Meta-heuristic Algorithms in the cloud computing environment. But most of these algorithms largely focus on initial placement of the VMs. Virtual Machine placement should focus on power saving and abide by the Service Level Agreement to provide Quality of service to the users.

One of the ways to improve datacenters is by applying effective server consolidation techniques. This technique reduces the power consumption in a data center which is the most challenging task today to maintain the sustainability of the data centers. C. Sonklin et al., have presented a multi objective grouping genetic algorithm for minimizing energy consumption and also resource wastage. A fitness function has been determined that considers the two objectives with a trade-off between the objectives [9].

B. Zhang et al.,[10] have developed an algorithm to cluster the population of current generation and select individuals from different groups with reduced cross over operations. They have used the runtime to determine the preference of VMs on Physical machines and also to generate the initial solutions and proved that their algorithm performs better than the traditional genetic algorithm.

X. Wang et al.,[11] in their paper defines a mathematical model to reduce make span, cost and total tardiness. A dynamic resource allocation with pre selection has been proposed. They have used a Classifier to filter the sub problems solutions in decision space. The algorithm performed better than the traditional algorithms.

S. Garepasha et al., [12] defined a chaotic multi objective optimization algorithm for Virtual Placement in data centers... They have hybridized Sine Cosine (SCA) and Ant Lion Optimizer (ALO) to achieve load balancing of servers and also to reduce the resource wastage.

P. Boominathan et al., [13] in his work has applied fuzzy hybrid bio-inspired technique to solve the VM placement problem through server consolidation technique. Fuzzy rules were generated to choose the next VMs for the current server. Cuckoo search has been applied to find the new optimal solution. So by combining ACS and cuckoo search, they have developed an algorithm for server consolidation and by combining ACS and firefly colony algorithm, they have developed another algorithm for virtual machine placement problem. Both of them have proved to give the best A make span model, Cost and Utilization mathematical model have been designed using the effectiveness of Levy flight of Cuckoos to search the Optimal placement of the Virtual Machines is given in [23].

An optimized chaotic Grey wolf knowledge-based Ant Colony system to obtain the placement of Virtual

Network Functions and allocate paths gained by knowledge of Software defined networking controllers. It has been proved that the proposed algorithm converges in lesser iterations within less computational time. [24]

An Artificial Bee Colony [25] and Chicken Swarm optimization algorithm has been suggested in this paper for an efficient Virtual Machine Placement considering load, migration cost and power consumption to prove it performs better than existing techniques. Grey Wolf Optimization based Simulated Annealing Approach [26] is adopted for Container based virtualization rather than virtual Machine based virtualization and found to be better than other existing algorithms in terms of load variation and Make span.

In our proposed algorithm, minimizing energy consumption and minimizing the resource wastage are our main objectives. In section III, VM Placement Problem has been explained, in section IV, Evolutionary Multi-objective optimization objective function is defined. In section V, basics of ACO (Ant Colony Optimization) and SCA algorithms are explained. In section VI, ACOSCA Optimization of Virtual Machine Placement has been explained using the ACOSCA algorithm. In section VII, the experimental study and results are discussed and in section VIII, conclusion of the paper is given.

3 VM Placement problem

Virtual machine placement refers to the process of selecting an optimal physical server or host for deploying virtual machines (VMs) in a virtualized environment. It involves decisions regarding which virtual machines should be placed on specific physical servers based on factors such as resource utilization, performance requirements, and workload balancing. The goal of virtual machine placement is to optimize the allocation of resources and ensure efficient utilization of hardware infrastructure while meeting the performance and availability requirements of the virtualized applications or workloads.

In a virtualized data center, there are a large number of Physical Machines (PMs) denoted as 'm' with 'n' number of Virtual Machines (VMs) hosted on them. These Virtual Machines are hosted with many heterogeneous applications having varying resource utilization. To effectively use computing resources, it is important to distribute the VMs evenly among the PMs. Identifying where it is best to place a Virtual Machine on a Physical Machine has a direct impact on improving the resource utilization. This can be achieved by an efficient Virtual Machine Placement algorithm. When 'n' number of Virtual Machine requests arrives to a virtualized data center, where the Virtual Machines have to be placed on a set of 'm' Physical Machines efficiently is the VM Placement problem (VMP). The goal is to find the optimal arrangement of Virtual Machines across the physical servers in a data center. A VM can be placed based on two constraints. One is, a VM can be placed for the first time on a Physical Machine and the other one is, VM replacement which is for optimization purpose, in which an existing VM would be replaced without any compromise in the quality of service [3].

In cases of deviations in resource utilization in data centers by Cloud providers, such as workload variations or differences in CPU and memory usage, it is necessary to replace the virtual machines (VMs). However, it is important that the requested VM configuration by the user does not violate the Service Level Agreement (SLA). Frequent VM migrations lead to performance degradation and violations of the SLA.

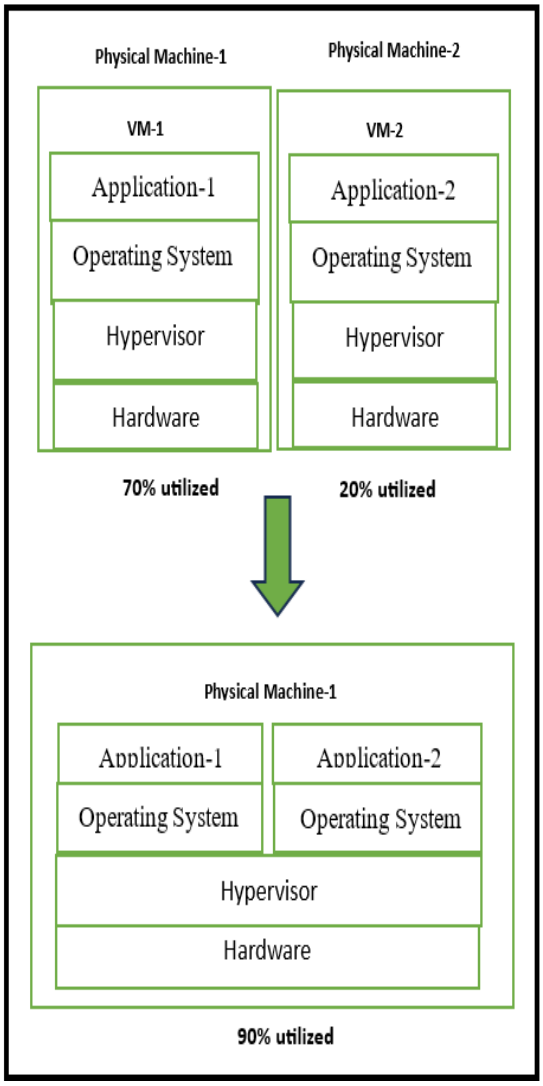

Figure 2: Virtual machine placement

Thus, it is essential to strategically place VMs in the data center to minimize migrations, enhance energy efficiency, and optimize resource utilization The Bin Packing Problem was previously used to solve the VMP problem in cloud The Bin Packing Problem was previously used to solve the VMP problem in cloud computing environments. However, it is difficult to find optimal solutions quickly using algorithms, which is why it is considered a NP-hard problem. NP-hard algorithms do not have a specified solution, but they can suggest an optimal solution. To address this, biological algorithms such as Ant Colony Optimization, Genetic Algorithm, and

Particle Swarm Optimization, which are meta-heuristic algorithms, are used to find solutions that are closer to optimal. In this paper, we propose a hybrid Optimization (ACOSCA) algorithm for Virtual Machine Placement. . The ACOSCA algorithm takes into account the dynamic requirements of users to optimize resource utilization. Additionally, several Power Aware algorithms were analyzed to optimize power consumption based on servers, network bandwidth, etc.,[29][30][31][33]. The performance of the ACOSCA algorithm was evaluated and compared to other algorithms. It was found that ACOSCA outperformed First Fit Decreasing (FFD), Max-Min Ant System (MMAS), Ant Colony System (ACS), and Fire-Fly Colony (FFC) algorithms in terms of multi-objectives, resource wastage, and power consumption of servers. The ACOSCA algorithm has been implemented using JAVA, and the results and observations showed better performance compared to existing algorithms. Results and observations showed better performance compared.

3.1 VM Placement problem definition

Evolutionary multi-objective algorithms employ a population-based method to discover an optimal solution. These algorithms aim to optimize multiple objective functions simultaneously, resulting in Pareto optimal solutions that improve more than one objective function. The performance of a solution may be impacted in the other remaining functions. Many algorithms utilize the concept of dominance during the selection process. A multi-objective minimization problem can be defined as follows:

Minimize

$$\vec{f}(\overrightarrow{dv}) = [f1(dv_1 \dots dv_m), \dots . fn(dv_1 \dots dv_m)] \quad (2)$$

Here,

$$\overrightarrow{dv} = (dv_1 \dots dv_m) \in X$$
$$\vec{f} = (f_1 \dots f_n) \in Y$$

Where m is set of decision variables and n are the set of objectives. $\overrightarrow{dv}$ denotes the decision vector, $\overrightarrow{fv}$ denotes the objective vector, X is the variable space and Y is the objective area.

The dominating points are those in which the decision vector $\overrightarrow{dv}$has better objective than any other decision vector. In a solution set, find all the non-dominated, multi-objective solution set. Start with first decision variable. Compare first variable with all other remaining variables for domination. A solution dv_i is dominating the other solution dv_j when the first one is better in its objectives. Mark the dominating solution and all the solutions except the marked one are non-dominated solutions. Consider we have some 'n' number of Physical Machines running applications on them. In case assume that, all the applications are treated as a separate VM to be executed. Mapping a VM to a PM is a multidimensional vector packing problem. Various resource utilizations (CPU and memory) represent the various dimensions. Let us take for example, a request for a VM containing 20% CPU and 30% memory and another request of VM having 35% CPU and 40% memory. Then the usage of server shall be calculated as 55% CPU and 70% memory. The resources shall be utilized up to 90% for each dimension. This example has been illustrated in Fig.2.We need to impose a boundary 90% which is less than 100% utilization in order to avoid the performance degradation of the server otherwise, which may lead to migration of VMs.

4 Server consolidation

4.1 Objective function of server consolidation

The objective function is designed so as to reduce the quantity of physical machines as well as not violating the capacity of the physical server. Consider that we are assigned 'n' number of VMs where (i∈ I). Here VMs are applications which are need to be assigned to 'm' servers (j ∈ J). VM_{ij} andPM_j are the two binary variables used to represent the placement of VM. The first binary variable VM_{ij} denotes if a virtual machine i, is assigned to server jand the binary variable PM_j represents if server is active. Our objective is to minimize the power consumption while minimizing the resource wastage. Therefore, the server consolidation can be mathematically formulated as:

$$\text{Minimize } \sum_{j=1}^{m} PM_j \quad (3)$$

The limits on constraints are:

$$\sum_{j=1}^{m} \text{lev}_{ij} = 1, \forall i \in I \quad (4)$$

$$\sum_{i=1}^{n} \text{D}_{pr}^{i} \text{lev} \leq T_{pr}^{j}. PM_j \text{ , } \forall \text{ j} \in \text{J} \quad (5)$$

$$\sum_{i=1}^{n} \text{D}_{my}^{i} \text{lev}_{ij} \leq T_{my}^{j}. PM_j \text{ , } \forall \text{ j} \in \text{J} \quad (6)$$

$$lev_{ij} \in \{0, 1\} \text{ , } \forall i \in I \text{ and } j \in J \quad (7)$$

According to Constraint (4), one VM is leveraged to only one PM. Constraints (4) and (5) are related to the capacity constraints of the Physical Machines and constraint (7) chooses the domain of the problem. The binary decision variables states whether a server is active or not. There are possibly m^n solutions possible for Virtual Machine placement. Our ACOSCA algorithm shows how to find the best solution among this enumeration of solutions.

4.2 Virtual machine placement problem formulation

The formulation of VM placement is presented in this section. Here it is considered as a multi-objective optimization problem. In this section, the Optimization equations have been formulated for Virtual Machine Placement problem. Consider the power consumption

Let the power be

$$Pr_j \begin{cases} [(Pr_j^{busy} - Pr_j^{idle}) \times U_j^{Pr}] + Pr_j^{idle} \\ 0, \quad otherwise \end{cases} \quad (8)$$

Were, $U_C^j > 0$

We have set (Pr_j^{busy}= 215 Watts and Pr_j^{idle} = 162 Watts. Let us consider the worst-case example, that is, each Physical Machine can have only one Virtual Machine where the number of servers will be equal to the number of VMs. The experiment shows the result for 200 VMs. For the Proposed ACOSCA algorithm, the parameters are given as follows:

q=0.8, TN_A=10, M=100, $\propto$ =0.45,ρ_i=ρ_g=0.35,τ_{pj}

= τ_{mj}=90% and ɳ=0.0001 (9)

There are many resources on each node as we saw in the Server consolidation section. There are many VM placement solutions are available. These resources on all cores of a node should be utilized without wastage of resource. The Wastage resource cost is evaluated by the following equation:

$W_j = \frac{\left|L^{pr}_j - L^{my}_j\right| + \Delta}{\left|U^{pr}_j - U^{my}_j\right|}$, as we saw ion Eq.1.

W_j represents the resource wastage in the j[th] server. $U^{pr}_j - U^{my}_j$ represents the CPU and memory usage normalized in a physical machine. It is the ratio between used resources to the total resources available. $L^{pr}_j - L^{my}_j$ *i*s the capacity constraint to avoid the utilization of full capacity of the physical machine coming down to zero and it is set to.0001

To cut down the number of servers utilizing the full capacity of the nodes, the minimizing function is given as follows [27]:

Minimize

$$\sum_{j=1}^{m} y_j \quad (10)$$

The objective function for Virtual Machine consolidation is similar to server consolidation. The conditions for minimization of power and resource wastage are given as in [23]. The VM placement is given as:

Minimize

$$\sum_{j=1}^{m} PM_j = \sum_{j=1}^{m} \left[y_j \times \left((PM_j^{busy} - PM_j^{idle}) \times \sum_{j=1}^{n} lev + PM_j^{idle}\right)\right] \quad (11)$$

Minimize

$$\sum_{j=1}^{m} W_j = \sum_{j=1}^{m} \left[y_j \times \frac{\left|\left(T^j_{pr} - \sum_{i=0}^{n} (lev_{ij} R^i_{pr})\right)\right| + \epsilon}{\sum_{i=0}^{n} (lev_{ij} R^i_{pr}) + \sum_{j=1}^{n} (lev_{ij} R^i_{my})} \right] \quad (12)$$

The ACO has two heuristic functions as given below:

$$\eta_{ij1} = \frac{1}{\varepsilon + \sum_{v=1}^{j} \left(\frac{PM_v}{PM_v^{max}}\right)} \quad (13)$$

$$\eta_{ij2} = \frac{1}{\varepsilon + \sum_{v=1}^{j} W_v} \quad (14)$$

The sum of the two heuristic functions for VM-PM mapping is given as:

$$\eta_{ij} = \eta_{ij1} + \eta_{ij2} \quad (15)$$

The Pheromone content in the edge(i,j), heuristic function of the desirability of adding a VM to a PM, control parameters determines the probabilistic selection of PM.

In FFD algorithm [4], VMs are taken in a decreasing order of utilization of a software resource. The result is also scalable to larger data centers and for a larger number of VM requests. 0.5 is fixed to P and the number of VM images is increased to 2000. We can observe that when$R_{pr} = R_{my}$=25%, the time taken to calculate a new placement consisting of 1000 VMs is 31 seconds and for 2000 VMs, it takes 114 seconds. For $R_{pr} = R_{my}$=45%,

we can observe that time taken to calculate a new placement consisting of 1000 VMs is 36 seconds and for 2000 VMs, it takes only 133 seconds. Our algorithm proves that the Virtual machine placement takes less time when compared to other algorithms and hence can be used in large data centers too. The number of Physical Machines used is different in both cases. Our algorithm proves that the Virtual machine placement takes less time than other algorithms and hence can be used in large data centers too where it will execute out the placement algorithm faster than the existing algorithms.

5 Basics of ant colony optimization and sine cosine algorithms

5.1 Ant colony optimization

Ant colony optimization is an optimization technique learnt from the behavior of the ants in searching their food by finding the nearest path from their habitat to the food source. Ants find their path by choosing random decision taken by the amount of Pheromone (a kind of saliva like substance) secreted by the ants on the way to the food source. The information passed by the other ants also is used to find the optimal solution. The information to assign VMi to PMjis given as below:

$$\eta_{ij} = \frac{|D^i_{pr} + D^i_{my}|}{|L^i_{pr} + L^i_{my}| + \in} \tag{16}$$

The Pheromone update is done in order to increase the Pheromone value corresponding to the good solutions and decreasing the Pheromone values is done by evaporation of the pheromones, that is, the odor of the non-optimal solutions is decreased.

The Pheromone trail update is given by:

$$\tau_{ij} = \begin{cases} \frac{\sum_{u \in \Omega_{k(j)}} \tau_{ui}}{|\Omega_{k(j)}|} & if\ \Omega_{k(j)} — \{i\} \neq 0\ , \\ 1,\ otherwise \end{cases} \tag{17}$$

The solution is constructed using Pseudo random proportional rule as given in [27].

$$I = \begin{cases} \text{argmax}_{u \in \Omega_k}\ (j)\left\{\propto_p\ x\ \tau_{ij}(-\propto_p) \times \eta_{ij}\right\},\ c < c_0 \\ \text{explore b,}\ \ otherwise \end{cases} . \tag{18}$$

Where c is a probabilistic parameter, which is distributed uniformly in the range of [0, 1]. c_0 is a variable which has value in the range of 0 and 1. If c is lesser than or equal to c_0 , then exploitation process takes place, and if it is greaterthan c_0, it is called exploration of newsolutions.$\propto_p$ is the variable through which the user can control the pheromone trail. Using the roulette wheel selection method, we select the random variable b, and using the proportional rule random probability distribution [13][33].

$$P^k_{i,j} = \frac{\propto_{pr} \times \tau_{uj} + (1 - \propto_{pr}) \times \eta_j}{\sum_{u \in \Omega_{k(j)}} (\propto_{pr} \times \tau_{ij+(}(1 - \propto_{pr}) \times \eta_j}, \quad i \in \Omega_{k(j)} \tag{19}$$

The parameters alpha, beta is called control parameters and evaporation rate is also a parameter to find the best optimal path based on the probabilistic value.

$\propto_{pr}$ is the control parameter of pheromone trail.

$$\Omega_{k(j)} = \begin{cases} i \in \{1, n\}\left(\sum_{u=1}^{m} lev_{iu} = 0\right)\ \wedge \\ \left(\left(\sum_{u=1}^{n}\left(lev_{uj} \times D^i_{pr}\right)\right) + D^i_{proc}\right) \leq T^j_{pr}\right)\ \wedge \\ \left(\left(\sum_{u=1}^{n}\left(lev_{uj} \times D^i_{my}\right)\right) + D^i_{my}\right) \leq T^j_{pr}\right) \end{cases} \tag{20}$$

The local updating of the Pheromone is done using the below equation:

$$\tau_{ij} = (1 - \varphi_g)\tau_{ij}(t-1) + \varphi_1 . \tau_0 \tag{21}$$

The ant reduces the pheromone trail when a VM is assigned to Physical Machine, j. Here, the pheromone decay $is\ = \varphi_1 \in \{0,1\}$ and τ_0 indicates the initial value of the pheromone. Fitness function of the derived solutions is evaluated using the cost function designed by H. Salami et al.,[23]. According to the fitness of the VM, it will be packed in the PM as given below:

$$\frac{D^{pr}_i + D^{my}_i}{\left(T^{pr}_i - \sum_{k=1,k\neq i}^{n} D^{pr}_k\right) + \left(T^{my}_i - \sum_{k=1,k\neq i}^{n} D^{my}_k\right)} \tag{22}$$

The Pheromone update to form the best solution is given by the following equation.

$$\tau_{ij}(t) = (1 - \varphi_g)\tau_{ij}\tau_{ij}(t-1) + \varphi\delta\tau^{best}_{ij} \tag{23}$$

Here $\varphi_g \in \{0,1\}$ represents the evaporation rate which represents non-optimal solutions', which might bias the ants to travel towards non promising area of the search space. A pheromone evaporation rate is important in deciding the effectiveness of the calculation.

$$\tau^{best}_{ij} = \begin{cases} f_{sol}(S^{gb})\ if\ \ VM_i\ is\ placed\ in\ server\ j. \\ 0\ \ otherwwise\ , \end{cases} \tag{24}$$

5.2. Sine cosine algorithm

The Sine Cosine algorithm is a population based probabilistic search algorithm that updates the position of search agents to improve its population based on Sine and Cosine functions and obtain the solution for the problem. It switches between ACO and SCA models to achieve the best result. SCA is used to search locally for the solution obtained by each Ant. The working mechanism of SCA begins by randomly producing an initial set of x_i solutions X with D dimensions. Afterwards, for each solution, it calculates the fitness values. The solution with the best fitness value (f_b) is considered as the best solution (x_b). The x_b and the parameters β_i = 1, 2, 3, 4 are used to update the solutions as given in (17).

SCA repeats these steps until the global best condition is met, as shown in Table I. The updating procedure is followed as it is described in [32]. SCA updates its population using the sine function as given in the equation below:

$$x_i(g+1) = x_i(g) + \beta_i \times sin\beta_2 \times |\beta_3 x_b \times x_i(g) - x_i(g)| \quad (25)$$

and updates its population using the cosine function as follows:

$$x_i(g+1) = x_i(g) + \beta_i \times sin\beta_2 \times |\beta_3 x_b x_i(g+1) = x_i(g) + \beta_i \times sin\beta_2 \times |\beta_3 x_b \times x_i(g) - x_i(g)| \times x_i(g) - x_i(g)| \quad (26)$$

Where g is the number of present iterations and x_i is the optimal solution. As we can evaluate Sine and Cosine of any real number, both of these functions are defined by real numbers. The points are taken in the range (-1,1) and the shape repeats after every 2π as shown in the Fig.3. Now $x_i \in (0,2\pi)$ is a random variable and β_3 is also a random variable. The SCA algorithm uses the objective function [27] to switch between these equations as shown in the following:

$$x_i(g+1) = \begin{cases} x_i(g) + \beta_1 \times sin(\beta_2) \times |\beta_3 x_b(g) - x_i(g)| \text{ if } \beta_4 < 0.5 \\ x_i(g) + \beta_1 \times cos(\beta_2) \times |\beta_3 x_b(g) - x_i(g)| \text{ if } \beta_4 > 0.5 \end{cases} \quad (27)$$

where, $x_b(g)$ and $x_i(g)$ define the target and the current solutions of iteration g, respectively. $\beta_i, where\ i = 1,2,3,4$ represents some random number that add weights to x_b to check whether it stochastically preserves when ($\beta_3 > 1$) or not when ($\beta_3 < 1$), influencing in finding the domain. β_1 is applied to switch between exploration and exploitation in order to define the optimal part for the next solution. This part may be higher than the upper bound or lower than the lower bound, thus, it is updated as follows:

$$\beta_1 = \sigma - g\frac{\sigma}{g_{max}} \quad (28)$$

where g, σ and g_{max} define the current iteration, a constant, and the iterations number, respectively. At the starting of the iteration, a bigger convergence factor is expected to improve the searching process and as the iterations increases, it converges to a lesser value.

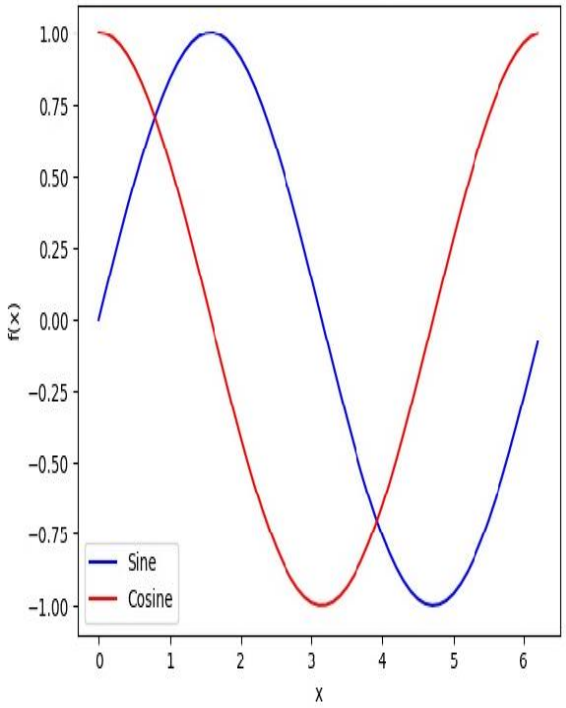

Figure 3: Sine and Cosine with the range of [-1,1]

Most of the Swarm intelligence algorithms reach the best solution by exchanging the information between the swarm's individuals. Here the Ants are going to exchange the information obtained by applying Sine Cosine algorithm. In the reference paper [16], the excellent performance of SCA has been proved and applied on the airfoil design problem successfully. It has been proved that the Sine Cosine Algorithm works well in optimization algorithms in finding the local best solutions. Sine Cosine algorithm begins with randomly generated solutions and updates its position corresponding to the best candidate using the Sine Cosine functions for exploiting the solutions generated from the ACO algorithm in searching the local regions in the search space.

Table 1: Sine cosine algorithm

1. Produce a set of solutions x_i, (Search agents) where i ranges from $1\ to\ n$
2. Compute the fitness value f for each solution.
3. Define x_b that has the best fitness value f_b
4. Update $\beta_i, i = 1,2,3,4$ values.
5. Use Eq. (19) to update X.

end while

7. Output x_b

In the Table I given above,

$x_i(g)$ represents the ant i, at itertion g.

$x_b(g)$ represents the best ant position

$\beta_i, where\ i = 1,2,3,4$ represents some random values.

6 ACOSCA algorithm for VM placement

In the SCA, to decide the distance of the next searching region, the position updating equation uses the destination point. We need to make a decision on which VM should be placed on the selected server, as described in (29). This section provides an explanation of the rules created to determine the appropriate choice of VM_i for the selected Physical MachinePM_i to be placed in the currently chosen server as given in (29).

The Fuzzy rules to be followed to place a VM on a PM are given below.

If β_{ij} is low and ηij is low then the efficacy e_{ij} of choosing VM i is very very low.

If β_{ij} is medium and ηij is low then the efficacy e_{ij} of choosing VM i is very low.

If βij is high and ηij is low then the efficacy e_{ij} of choosing VM i is low.

If βij is low and ηij is medium then the efficacy e_{ij} of choosing VM i is low.

If βij is medium and ηij is medium then the efficacy e_{ij} of choosing VM i is medium.

If βij is high and ηij is medium then the efficacy e_{ij} of choosing VM i is high.

If βij is low and ηij is high then the efficacy e_{ij} of choosing VM i is high.

If βij is medium and ηij is high then the efficacy e_{ij} of choosing VM i is very high.

If βij is high and ηij is high then the efficacy e_{ij} of choosing VM i is very very high

Minimum operation for fuzzy implication and max-min operator for complication has been used in this method.

Table 2: ACOSCA Algorithm for VM Placement

1. Initialize the required quantity of Physical Machines (PMs) and required quantity of Virtual Machines
2. The Capacity constraint is set for Physical Machines
3. The basic requirement demands of VMs are initialized
4. The maximum number of iterations is fixed.
5. Initialize the Pheromone matrix τ_{ij} and the total number of ants TN_A
6. Use the procedure given in Table II to create Virtual Machine instances.
7. $Loop - 1$:
8. For each Ant uptoTN_A
9. $Loop - 2$:
   a. Take a server which has not been used so far from a set of Physical servers
   b. $Loop - 3$:For No. of VMs $= 1\ to\ n$
      i. Determine the desirable Heuristic
      ii. data from equation (16)
      iii. Determine the probabilistic Movement from equation (19)
      EndFor\
   c. Pick a VM from the list of VMs for placement
   d. Usestate-transition conditions applying (30) and (31)
   e. If VMs are still remaining then Go to $Loop - 3$
   f. The Pheromones are updated to the local best solution using Equation (21)
10. Go to $Loop - 2$
11. The objective function is set as in Eq. (3)
12. Apply SCA procedure to obtain new optimal Solutions given in Table I
13. Update the Ant Pheromones values by updating the global updating condition given in Equation (23)

go to $Loop - 1$

14. Print the global best solution and the fitness output value

15. Output the value

So, we employ a hybrid technique that utilizes minimal and max-min operations in implication and composition, respectively. As a result, the maximum efficiency e_{ij}^{k} is obtained for every virtual Machine i. The Fuzzy Strategy and Fuzzy Probable strategy rule

mentioned in reference (13), has been applied to determine which VM should be assigned to each server. Therefore, it decides whichVM_i should be assigned for an individual server j.

$$I=\begin{cases} \textbf{Fuzzy strategy}, \mathbf{q} \leq \mathbf{q_0} \\ \textbf{Fuzzy probable strategy}, \mathbf{q} > \mathbf{q_0} \end{cases} \quad (29)$$

The fuzzy probable strategy is derived from Sine Cosine Algorithm as mentioned in Table I. In the output, we will get the number of virtual machines to be placed in that corresponding server. To implement the exploitation process, a fuzzy technique is followed by implementing the fuzzy protocol. These are the state transition rules

Fuzzy Strategy:

$$\lfloor \mathbf{e_{u^*}j} \rfloor = \frac{\sup}{\mathbf{u} \in \Omega_{\mathbf{k(j)}}} \{\mathbf{e_{uj}}\} \quad (30)$$

where **Fuzzy Probable Strategy**: $= u^*$. The Sine Cosine algorithm is applied here for Fuzzy probable strategy in findingX_{ij}^k.

Fuzzy Probable Strategy:

The formula for Fuzzy probable strategy is given below:

$$X_{ij}^k = \frac{e_{ij}^k}{\sum_{u \in \omega_k(j)} e_{uj}^k} \quad (31)$$

The ACO algorithm is most suitable for solving heuristic approach problems with the help of heuristic information given by η_{ij}. This heuristic information depends upon the current ant position. For each ant, η_{ij} is calculated and all ants start with a set of VMs to be arranged on a list of servers. During the construction of the solution, an ant chooses the next Physical Machine based on the Pseudo random proportional rule as given in Equation.10. According to this rule, the ant selects the next VM to be placed on the current node. The pheromone deposit on the way to the PM and the heuristic of SCA both assist in determining the optimal VM to be placed on the Physical Machines. In Table III, th**e** algorithm proposed in [33] is described for generating random problem instances. Each problem instance consists of 200 VMs, with the number of VMs equal to the number of servers in the configuration. In this case, only one VM can occupy each server. A solution is randomly selected and 20 iterations are performed on each instance to generate a random number, the function rand () is used. $D_{pr(i)}$ and $D_{my(i)}$represent the demand for CPU and Memory, respectively

Table 3: Algorithm to create the processor and memory instances in random

for i = 0 to (n-1) do

$D_{pr(i)} = rand(\overline{2\ D_{pr}})$

$D_{my(i)} = rand(\overline{2\ D_{my}})$

s=Numbers generated using function *rand (1.0)*

if (s<P $\wedge$ $D_{pr(i)} \geq D_{pr}$) $\vee$ (s $\geq$P $\wedge D_{pr(i)} < D_{pr}$ then

$$D_{my(i)} = D_{my(i)} + \overline{D_{my}}$$

Endif

Endfor

In the above table, the variables used are:
$D_{pr(i)}$ - CPU Usage Demand
$D_{my(i)}$ - MemoryUsage Demand
P - Reference Probabilty

6.1 Experimental study

In this section, the experimental results of the ACOSCA algorithm to solve the Virtual Machine Placement have been given below from Tables IV to VIII. VM instances based on their resource requirements and maps them to servers in decreasing order of a particular resource like CPU utilization or memory usage. It then places the VM to the first fit PM. From the tables shown below, it is evident that FFD consumes more power and wastes more resources compared to the other algorithms [4]. Second, the MMAS [33] algorithm is a stochastic approach that dynamically adjusts the exploration and exploitation trade-off based on evolving system conditions. From the

readings of the Tables, it is better than FFD but does not give an admirable result when compared to our approach. Next, we take FFC algorithm [13]in which the fireflies build the solution using incremental stochastic solution by assigning the VMs based on probability. The VMs are mapped on to the PMs until a lower bound has been attained. The performance of this algorithm is also is not satisfactory when compared to the readings of the Tables listed above. At last, we compare our algorithm with the ACO algorithm [31. The ACO algorithm is best suitable for reducing the resource wastage and improving the resource utilization. When comparing the performance of ACO with our proposed algorithm, more resource wastage and power consumption has been observed. So, we take the behaviour of this algorithm to optimize the execution of our algorithm and it is combined with the SCA algorithm which is best suited for the local search process due to its convergence process. So ACO algorithm is used for global searching in finding the best suitable server and Sine Cosine Algorithm is applied in local searching process of mapping the VM to the Server which has the suitable configuration of the VM instance. From Fig.4, it is evident that the ACOSCA algorithm consumes less power and minimizes the resource wastage to a considerable level also improving the system performance by executing in lesser time duration.

Table 4: Requirements of vm placement (reference values = 25% and 45%, probability values = (-0.754 and -0.755) -a comparison

Algorithm	R_{pr}= R_{my} =25%				R_{pr}= R_{my} = 45%			
	Probability Value = -0.754				Probability Value = -0.755			
	Power (w)	**Resource Wastage (W_j)**	**Fitness ($x10^{-3}$)**	**Time (s)**	**Power (w)**	**Resource Wastage (W_j)**	**Fitness ($x10^{-3}$)**	**Time (s)**
ACOSCA	20400	6.08	917	7.13	30661	10.98	903	8.52
ACS	20635	7.22	889	5.31	30975	11.11	891	6.13
FFC	20980	7.31	859	5.37	31305	11.32	822	6.31
MMAS	21900	7.86	852	5.60	31338	11.88	826	6.46
FFD	24798	8.16	716	8.34	34959	18.91	746	24.51

Table 5: Vm table 5: requirements of vm placement (reference values = 25% and 45%, probability values = (-0.348 and -0.374) - a comparison

Algorithm	R_{pr}= R_{my} =25%				R_{pr}= R_{my} = 45%			
	Probability Value = -0.348				Probability Value = -0.374			
	Power (w)	**Resource Wastage (W_j)**	**Fitness ($x10^{-3}$)**	**Time (s)**	**Power (w)**	**Resource Wastage (W_j)**	**Fitness ($x10^{-3}$)**	**Time (s)**
ACOSCA	20279	5.17	910	7.11	30411	10.61	880	8.5
ACS	20450	5.72	885	5.31	30776	10.82	896	6.09
FFC	21576	6.13	857	5.39	31165	11.02	828	6.79
MMAS	21633	6.05	849	5.63	31270	11.9	838	6.42
FFD	24670	7.36	711	8.02	34702	17.13	759	23.14

Table 6: Vm requirements of vm placement (reference values = 25% and 45%, probability values = (-0.072 nd -0.052) - a comparison

Algorithm	R_{pr}= R_{my} =25%				R_{pr}= R_{my} = 45%			
	Probability Value = -0.072				Probability Value = -0.052			
	Power (w)	Resource Wastage (W_j)	Power *(w)*	*Resource* Wastage (W	*Power (w)*	*Resource Wastage (W_j)*	*Power (w)*	*Resource Wastage (W_j)*

ACOSCA	18074	3.96	18074	3.96	18074	3.96	18074	3.96
ACS	18254	4.02	18254	4.02	18254	4.02	18254	4.02
FFC	18443	4.09	18443	4.09	18443	4.09	18443	4.09
MMAS	19491	5.86	19491	5.86	19491	5.86	19491	5.86
FFD	24466	6.79	24466	6.79	24466	6.79	24466	6.79

Table 7: Vm requirements of vm placement (reference values = 25% and 45%, probability values = (0.371 and 0.398) - a comparison

Algorithm	$\mathbf{R_{pr}}$= $\mathbf{R_{my}}$ =25%				$\mathbf{R_{pr}}$= $\mathbf{R_{my}}$ = 45%			
	Probability Value = 0.371				Probability Value = 0.398			
	Power (w)	Resource Wastage ($\mathbf{W_j}$)	Fitness (x$\mathbf{10^{-3}}$)	Time (s)	Power (w)	Resource Wastage (W_j)	Fitness (x$\mathbf{10^{-3}}$)	Time (s)
ACOSCA	17847	3.39	934	7.19	27856	4.77	938	8.95
ACS	18200	3.58	908	5.35	28200	5.75	921	6.21
FFC	18215	3.61	884	5.34	28365	5.86	849	6.34
MMAS	19016	3.94	872	5.68	28316	5.89	856	6.42
FFD	21871	4.23	742	7.86	33654	10.18	776	21.12

Table 8: Vm requirements of vm placement (reference values = 25% and 45%, probability values = (0.755 and 0.75) - a comparison

Algorithm	$\mathbf{R_{pr}}$= $\mathbf{R_{my}}$ =25%				$\mathbf{R_{pr}}$= $\mathbf{R_{my}}$ = 45%			
	Probability Value = 0.755				Probability Value = 0.751			
	Power (w)	Resource Wastage (W_j)	Fitness (x$\mathbf{10^{-3}}$)	Time (s)	Power (w)	Resource Wastage (W_j)	Fitness (x$\mathbf{10^{-3}}$)	Time (s)
ACOSCA	17294	2.01	944	7.07	26862	3.25	944	8.76
ACS	17309	2.02	932	7.04	26881	3.29	932	8.41
FFC	17870	2.61	887	5.24	27332	3.46	847	6.24
MMAS	19038	2.76	876	5.41	28334	3.91	859	6.14
FFD	21270	3x.41	744	7.63	33400	5.42	776	21.06

7. Discussion of the observations

The result of the experiment demonstrates the effectiveness of the hybrid meta-heuristic ACOSCA algorithm in addressing the Virtual Machine Placement problem. The Number of CPUs and Memory in the VMs are taken in the X-axis The correlation coefficient of the CPU and Memory are found out statistically and this metric is taken on X-axis and compared against the Power consumed by the Physical Machines totally on Y axis Also, the same correlation coefficient is used for the resource wastage also. The number of Physical Machines saved in comparison to the other algorithms. The VM demand instances were generated using [33], and the mapping was performed with a worst-case complexity assumption of assigning one VM to a single PM. A total of 200 VMs were used in the experiment, with the potential for scalability to mega data centers and up to 2000 VM requests. The variables were initialized with the following values: q_0= 0.8. Na=10, M=100, $\alpha = 0.45$, ρ_l=ρ_g=0.35, τ_{pg}=τ_{mj}=90%, η =0.0001. The experiment was executed for 20 runs, and the outcomes are presented in Tables IV to VIII. The results indicate that the hybrid approach of ACO and SCA exhibits reduced power consumption and resource wastage compared to FFD, FFC, MMAS, and ACS. The experimental results illustrate the application of the hybrid multi-objective Ant Colony Optimization algorithm in solving the VM placement problem. The comparison metrics used include average power consumption, average resource wastage, average fuzzy fitness, and CPU execution times. From the results obtained in the Tables IV-VIII and from the Fig.4, we could infer that the hybrid approaches outperform the other single optimization algorithms considered in terms of these measures. The experiment proves that it is scalable and so suitable for mega data centers also. The aforementioned tables compare the ACOSCA algorithm with the FFD, FFC, MMAS, and ACS algorithms. Initially. the FFD algorithm organizes VM instances based on their resource requirements and maps them to servers in decreasing order of a particular resource like CPU utilization or memory usage. It then places the VM to the first fit PM. From the tables, it is evident that FFD consumes more power and wastes more resources compared to the other algorithms [4]. Second, the MMAS [33] algorithm is a stochastic approach that dynamically adjusts the exploration and exploitation trade-off based on evolving system conditions. From the readings of the Tables, it is better than FFD but does not give an admirable result when compared to our approach. Next, we take FFC algorithm [13]in which the fireflies build the solution using incremental stochastic solution by assigning the VMs based on probability. The VMs are mapped on to the PMs until a lower bound has been attained. The performance of this algorithm is also is not satisfactory when compared to the readings of the Tables listed. At last, we compare our algorithm with the ACS algorithm [31. The ACS algorithm is best suitable

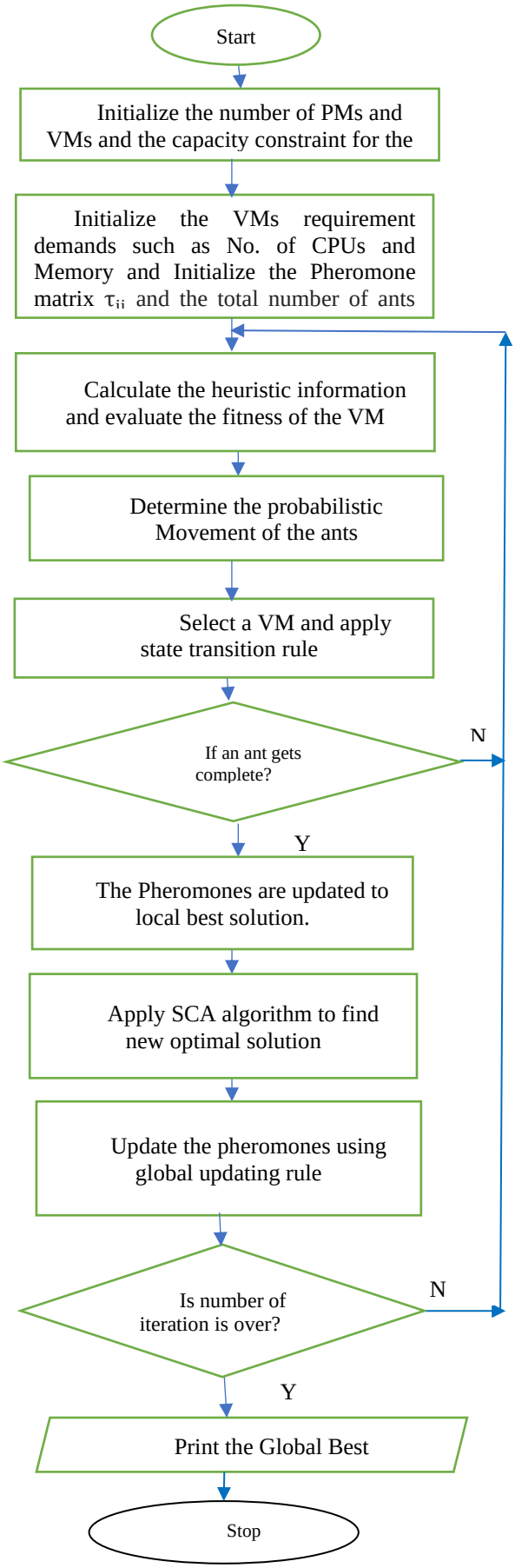

Figure 4: Flowchart for ACOSCA algorithm

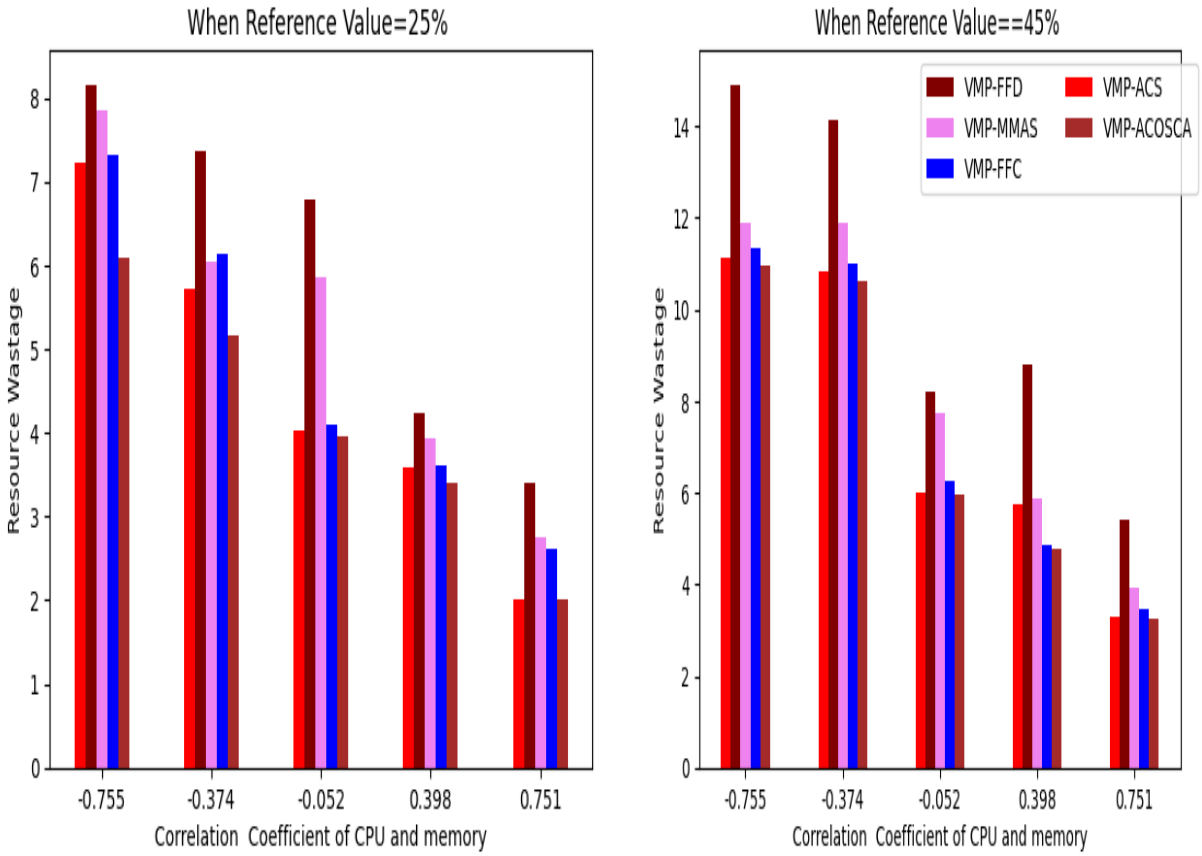

Figure 5: Comparison of algorithms for Resource wastage when the reference values = 25% and 45%

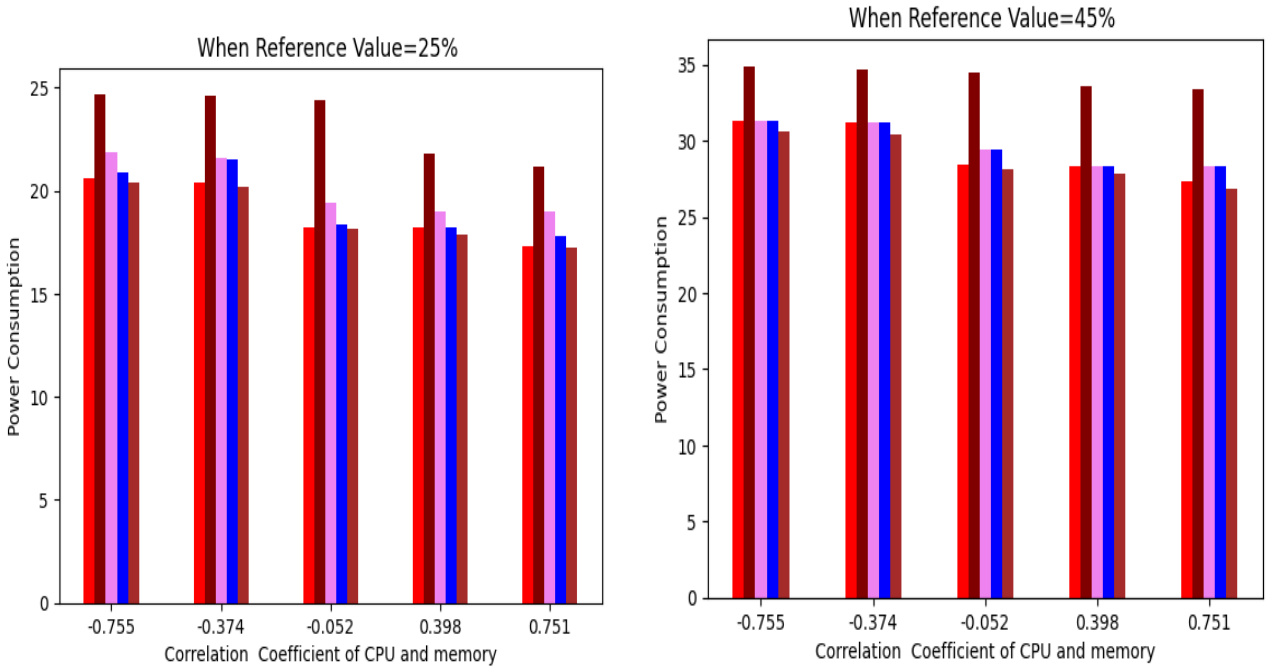

Figure 6: Comparison of algorithms for power consumption when the reference values = 25% and 45%

for reducing the resource wastage and improving the resource utilization. When comparing the performance of ACS with our proposed algorithm, more resource wastage and power consumption has been observed. So, we take the behaviour of this algorithm to optimize the execution of our algorithm and it is combined with the SCA algorithm which is best suited for the local search process due to its convergence process. So ACO algorithm is used for global searching in finding the best suitable server and Sine Cosine Algorithm is applied in local searching process of mapping the VM to the Server which has the suitable configuration of the VM instance. The flow of ACOSCA algorithm is given in the flowchart in Fig.4. From Fig.5 and 6, it is evident that the ACOSCA algorithm consumes less power and minimizes the resource wastage to a considerable level also improving the system performance by executing in lesser time duration.

8 Conclusion

In conclusion, the use of ACO and Sine Cosine algorithms has shown promising results in improving the performance of VM placement problems. The Novelty in our algorithm is the use of fuzzy rules to make the decision to place a VM in an active server or not. If both the power consumption and resource wastage are low, then it is considered as a good solution. However, there are also other bio-inspired optimization algorithms such as .Moth Search Algorithm (MSA)[7], Earthworm Optimization Algorithm, Elephant Herding Optimization(EHO), Rhino Herd Algorithm (RHA) and Monarch Butterfly Optimization(MBO), that can be explored for achieving better performance, though known for its efficient searching and faster convergence, has been successfully applied to optimize solutions generated by the ACO algorithm The experimental result proves that the ACOSCA algorithm improved the executing time by 3%, reduction in power consumption by 24% and increase in resource utilization by 16%.

This has resulted in server consolidation and increased resource utilization through the packing of maximum VMs in a physical machine. Consequently, resource wastage has been minimized, making the technique more efficient. In our technique, we used SCA algorithm to optimize the solutions generated by the ACO algorithm. The ACOSCA algorithm has demonstrated superior performance compared to other algorithms like ACS, MMAS, FFC, and FFD. In our technique, we used SCA algorithm to optimize the solutions generated by the ACO algorithm. The results of ACOSCA algorithm show a better performance than the compared ACS, MMAS, FFC and FFD algorithms.

Efficient VM placement plays a crucial role in ensuring the migration of VMs without any performance degradation [34]. Reports, such as the one from Uptimes Institutes Intelligence, indicate that consolidating workloads on the optimal number of servers increases the energy efficiency of data centers [35]. Looking ahead, there is potential for further improvement by continuously learning from the bio-inspired optimization algorithms mentioned above. Hybridizing these algorithms could lead to even more optimal and effective solution. The limitations of the algorithm are that the ACOSCSA algorithm heavily depends on the initial configuration of the algorithm parameters such as pheromone update where the inadequate initialization would lead to suboptimal solution. The Sine Cosine algorithm is a derivative free algorithm to exploit the neighborhood of potential solutions.

References

[1.] A.J. Younge, G. Laszewski, L. Wang, S.L. Alarcon, W. Carithers, “Efficient Resource management for Cloud Computing Environments”, Proc. IEEE Conference on Green Computing, Chicago, IL, USA, Aug.2010.

[2.] C.Vijaya and P. Srinivasan, “A Hybrid Technique for Server Consolidation in Cloud Computing Environment”, Cybernetics and Information Technologies”, vol.20, no.1, Mar.2020, pp.36-52.

[3.] N. Chamas, F.L. Pires, B. Baran, “Two-Phase Virtual Machine Placement Algorithms for Cloud Computing: An Experimental Evaluation under Uncertainty”, IEEE conference (CLEI), Cordoba, Argentina, Dec.2017.

[4.] R. Panigrahy, K. Talwar, L. Uyeda and U. Wieder, “Heuristics for Vector Bin Packing”, Research. microsoft.com, Available: https://www.microsoft.com/en-us/research/wp-ontent/uploads/2011/01/VBPackingESA11.pdf

[5.] H. Shirvani and Mirsaeid, “Bi-objective Webservice Composition problem in Multicloud Environment: A bi-objective time varying Particle Swarm optimization Algorithm”, Journal of Experimental and theoretical Artificial Intelligence, Vol.33, no 2, pp.179-202, Mar.2021.

[6.] Z. Usmani, S. Singh, “A survey of Virtual Machine Placing Techniques in a cloud data center”, Proc. (ICISP2015), Volume 78, Nagpur, India, Dec.2015, pp.491-498.

[7.] D. Alsadie “Virtual Machine Placement Methods using Meta-heuristic Algorithms in a Cloud Environment - A Comprehensive Review”, Journal of Computer Science and Network Security (IJCSNS), Volume 22, No.2, Apr.2022.

[8.] S.Y. Rashida and M. Saba, Md. M. Ebadzadeh, A. M. Rahmani, “A Memetic Grouping Genetic Algorithm for Cost Efficient VM Placement in Multi

Cloud Environment", Journal of Cluster Computing, Volume 23, Issue 2, Jun.2020, pp.797-836.

[9.] C.Sonklin, K. Sonlin, "A Multiobjective grouping Genetic Algorithm for Server Consolidation in Cloud Data Centers", Proc. (JCSSE2023), IEEE, Phitsanulok, Thailand, Aug.2023. pp.1-16

[10.] B. Zhang, X. Wang and H. Wang, "Virtual Machine Placement Strategy using Cluster based genetic algorithm", Journal of Neurocomputing, vol. 428, Mar.2021. pp. 310-316.

[11.] X.Wang, H. Lou, Z. Dong, "Decomposition based Multi objective Evolutionary Algorithm for Virtual Machine and Task Joint Scheduling of Cloud Computing in data space", Journal of Swarm and Evolutionary Computation, vol. 77, Mar.2023, pp. 1-17.

[12.] S. Garepasha and Md. Masdar, "A Discrete Chaotic Multi-objective SCA-ALO Optimization Algorithm for Optimal Placement in Cloud Data Centers", Journal of Ambient Intelligence and Humanized Computing, Vol.12, Issue.10, Oct.2021, pp.9323-9339.

[13.] P. Boominathan, M. Aramudan and Ra. K. Saravanaguru, "Fuzzy Bio-Inspired Hybrid Techniques for Server Consolidation and Virtual Machine Placement in Cloud Environment", Cybernetics and Information Technologies, vol. 17, no.4, Nov. 2017, pp. 52-68.

[14.] M.G. Gagwero and L. Caviglione, "Model Predictive Control for Energy Efficient, Quality-Aware Virtual Machine Placement", IEEE Transactions on Automation Science and Engineering, vol.16, no.1, Jan.2019, pp.420-432

[15.] S. Sharma, S. Kumar, S. Mohapatra, R. Rani, "Discrete Gravitational Search Algorithm for Virtual Machine Placement in Cloud Computing", International Journal of Advanced Science and Technology, vol.29, no..8s, Apr. 2020, pp. 1261-1267.

[16.] M.Wang and G. Lu, "A modified Sine Cosine Algorithm to solve Optimization problems", Journal of IEEE Access, Feb. 2021. pp.27434-27450.

[17.] H Xing, J.Zhu, R.Qu, P.Dai, S.Luo, Muhammad and A.Iqbal, "An ACO for Energy efficient and traffic Aware Virtual Machine Placement in Cloud Computing", Journal of Swarm and Evolutionary Computation, Volume 68, Feb. 2022,pp.1-18.

[18.] N.E. Chalabi, A. Attia, A. Bouziane and M. Hasaballah, "An improved Marine Predator Algorithmbased on Epsilon dominance and Pareto archive for Multiobjective optimization", Journal of Engineering Applications of Artificial Intelligence, Volume 119, no,1,Mar. 23,pp.1-18

[19.] M.A. Basset, R. Mohamed and S. Mirjalili, "A Novel Whale optimization Algorithm integrated with Neldar Mead Simplex for Multi-objective Optimization Problems", Journal Of Knowledge based Systems, Volume 212, Jan. 2021,pp.1-28

[20.] Z. Ding, L. Cao, L. Chen, D. Sun, X. Yi. Zhang and Z. Tao, "Large Scale Mutimodel Muti-objective Evolutionary Optimization" based on Hybrid hierarchical Clustering, Journal Of Knowledge based Systems, vol.266, Apr.2023. pp.1-22

[21.] Z. Xiang. G. Zhou and Q. Luo, "Golden Sine Cosine Salp Swarm Algorithm for Shape Matchnig using Atomic Potential Function", Journal of Expoert Systems, vol.39, no15, Nov.2021. pp.1-24.

[22.] Z. Xiang. G. Zhou and Q. Luo, "A New fusion of SalpSwarm with Sine Cosine for Non-Linear Function", Journal of Engineering with Computers, Journal of EWngineering with Computers, Jan2020, pp.185-212.

[23.] H. Salami, A. Bala and S. Sait, "An Energy Efficient Cuckoo Search Algorithm for Virtual Machine Placement in Data Centers", The Journal of Supercomputing, vol.77, no.11, Apr 2021, pp.1-28.

[24.] Farshin and S. Sharifian, "A Modified Knowledge based Ant Colony LAgorithm for Virtual Machine Placement and Simultaneous Routing of NFV in distributed Cloud Architecture", The Journal of Supercomputing, vol.75, no.8, mar. 2019.pp.5520-5550.

[25.] S.K. Sharma and W. Ghai, "Artificial Bee Colony Optimized VM Migration and Allocation using Neural Network Architecture", Journal of Advanced Technology and Engineering Exploration", vol.10,no.102, May. 2023, pp.590-607.

[26.] M.Patra, S. Misra, B. Sahoo and A. Turuk, "GWO-Based Simulated Annealing Approach for Load balancing in Cloud for hosting Container as a service", Journal of Applied Sciences, vol.12, no.21, Nov 2022, pp.1-22.

[27.] V.Maniezzo, "Exact and Approximate Non-Deterministic Tree Search Procedures for Quadratic Assignment Problem", Jounal on Computing, vol.11, no.9, pp.358-369, Nov.1999.

[28.] T.P. Shabeera, S.D. Madhukumar, S.M. Salam and K. Murali Krishnan, "Optimizing VM Allocation and Data Placement for Data-intensive Applications in Cloud using ACO Meta-heuristic Algorithm", Journal of Engineering Science and Technology, vol.20, no.1, Feb.2017. pp.616-628.

[29.] H. Feng, Y. Deng and J.Li, "A Global Energy Aware Virtual Machine Placement Strategy for Cloud Data Centers", Journal of System Architecture, Vol.116, no.1, Jun.2021., pp.1-12.

[30.] H.Z. JingW.F. Liu, Q. Wang, W. Zhang and Q. Zheng, "Power-Aware and Performance – Guaranteed Virtual Machine Placement in the cloud", IEEE Transactions on Parallel and Distributed Systems, Volume 29, No.6, 2018, pp.1385-1400.

[31.] Y. Qin, H. Wang, F. Zhu, L. Zhai, "A Multi-objective Ant Colony System Algorithm for Virtual Machine Placement in Traffic intense data Centers", Journal of System Architecture, Vol. 6, Oct. 2018, pp.1-12.

[32.] S.A. Mirjalili, "SCA-A Sine Cosine Algorithm for solving Optimization Problems", Journal of Knowledge based Systems, vol.96, no.1, Mar. 2016. pp.120-133.

[33.] Y. Gao, H. Guan, Z. Qi, Y. Hou and l. Liu, "A Multi-Objective Ant Colony System Algorithm for Virtual Machine Placement in Cloud Computing", Journal of Computer and System Sciences, 2013, pp.1230-1242.

[34.] S. Kosuru, D. Midhun chakkaravarthy and Md. Ali Hussain, "An Intelligent Energy Minimization Algorithm with Virtual Machine Consolidation for Sensor based decision support system", Journal on Measurement: Sensors, Volume 27, Jun.23,pp.1-27.

[35.] R. Bashroush and A. Lawrence, "Beyond PUE: Tackling IT's wasted Terawatts"; Uptime Institute, 2020, Available: https://uptimeinstitute.com/beyond-pue-ackling-it%E2%80%99s-wasted-terawatts.

[36.] Chayan Bhatt and Sunita Singhal," Hybrid Metaheuristic Technique for Optimization of Virtual Machine Placement in Cloud", IJFIS, 2023; Vol.23(3): 353-364

[37.] H.O. SalamS. M. Sait, A. Bala, « An energy-efficient cuckoo search algorithm for virtual machine placement in cloud computing data centers", Journal of Supercomputing, Apr 2021 ; 77(11),1-28

[38.] Anitha Ponraj, "Applying self-aggregation Optimistic virtual machine placement in cloud data centers using queuing approach to load balancing", FGCS, Vol.93, Apr-2019, pg:338-344

[39.] Boominathan Perumal, Aramudhan Murugaiyan, "A Firefly Colony and Its Fuzzy Approach for Server Consolidation and Virtual Machine Placement in Cloud Datacenters", Advances in Fuzzy Systems, Article ID 6734161, 2016.

[40.] Badieh Nikzad, Behnam Barzegar and Humayun Motameni, SLA Awareand Energy efficient Virtual machine placement and server consolidation in heterogeneous DVFS enabled cloud data centers.", IEEE Access, Aug 2022, vol.10, pg:81787 to 81804. DOI:10.1109/ACCESS.2022.3196240

[41.] Logistics Distribution Route Optimization Based on Improved Particle Swarm OptimizatioN, Hai Zhao*1, Ashutosh Sharma2, VOL. 47 (2023) 243-252, DOI.org: 10.31449/inf. v47i2.4011

[42.] Ali wided, Kazar Okba, "A Novel Agent based Load Balancing Model for maximizing Resource Utilization in grid Computing, Vol.43, No.3, 2019, DOI.org/10.31449/inf. v43i3.2944.