International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Optimizing Cloud Resource Allocation: A Green Approach with Enhanced Dragonfly Algorithm for VM Selection

Ajay Prashar[1]*, Jawahar Thakur[2]

Submitted: 10/10/2023 **Revised**: 29/11/2023 **Accepted**: 09/12/2023

Abstract: To provide dependable and scalable computational capacity, cloud computing is popular among businesses, academics, and the government. In cloud data centers, virtual and physical equipment are connected by high-speed networks. Virtualization helps cloud service providers manage resources more efficiently, however poorly optimized and inefficient services hurt the system's performance. Physical Machines (PMs), Virtual Machines (VMs), and the allocation and migration strategy of the VMs over the PMs are all components of the cloud computing scheduling architecture. The research offers a unique behavior of VM selection from overutilized PM using Swarm intelligence. The overutilized PMs receive a few migrations. With other state-of-the-art optimization algorithms from the same series, the suggested algorithm design is evaluated. In order to support flexibility, the evaluation was conducted based on Quality of Service (QoS) characteristics including Service Level Agreement (SLA) violation and energy usage. The results are described along with examples of how the suggested approach significantly outperformed previous strategies in terms of QoS.

Keywords: *Cloud Computing; Quality of Service; Physical Machines; Virtual Machines; Migration*

1. Introduction

The concept of cloud computing has emerged as a focal point for businesses, captivating attention due to its transformative impact. The global exchange of information, facilitated by the widespread use of the internet [1], underscores the critical role played by cloud centers in storing and seamlessly transmitting vast amounts of data. Cloud computing encompasses a spectrum of services, including Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) [2], presenting a versatile suite of solutions for businesses. Major industry players such as Google, Microsoft, International Business Machine (IBM), and Wipro have invested significantly in establishing centralized data centers worldwide to capitalize on the advantages offered by popular cloud services. However, the expansive scale of these cloud centers, comprising thousands of physical hosts, has raised concerns about their substantial energy consumption [3]. Despite the continuous construction of new data centers, existing cloud resources often remain underutilized [4], emphasizing the need for cost-effective solutions that ensure efficient and affordable utilization of Virtual Machines (VM) [5]. In addition to high energy demand, cloud infrastructures also face challenges such as cooling overhead, resource fragmentation, and network latency, which together increase operational costs and carbon emissions. Addressing the imperative of energy-efficient data centers, virtualization emerges as a pivotal technology, providing interoperable and flexible services. Nevertheless, the challenge lies in optimizing the selection of VMs from potentially over-utilized Physical Machines (PMs). Addressing the imperative of energy-efficient data centers, virtualization emerges as a pivotal technology, providing interoperable and flexible services. Nevertheless, the challenge lies in optimizing the selection of VMs from potentially over-utilized Physical Machines (PMs). In response, the integration of machine learning, coupled with Swarm Intelligence (SI), has shown promise as a solution for refining VM selection policies [6]. Notably, previous studies [7] have illustrated the effectiveness of meta-heuristics, or SI, in tandem with machine learning for enhancing VM selection policies Beyond VM allocation, researchers are also investigating hybrid techniques that combine heuristic algorithms with predictive analytics to anticipate workload patterns and proactively adjust resource distribution. Such proactive management not only improves service reliability but also reduces the number of costly and unnecessary migrations [2]. This research explores the intersection of cloud computing, energy efficiency, and optimization strategies, with a particular focus on the synergistic application of machine learning and Swarm Intelligence for VM selection. Through a comprehensive examination of these innovative approaches, this paper aims to contribute to the evolving discourse on enhancing the sustainability and performance of cloud computing infrastructures.

[1, 2]*Department of Computer Science, Himachal Pradesh University, Shimla-171005, Himachal Pradesh, India.*
[1*]*ajayprashar93@gmail.com,* [2]*jawahar.hpu@gmail.com*

Fig 1 Network of Host and Virtual Machine [8]

Figure 1 depicts the generalized allocation procedure and placement scheme of VM. In this work, we concentrated on the VM allocation and migration process by ensuring that the machines spend the least amount of power while committing the fewest number of service level violations. A crucial technique for balancing the load and increasing DC energy efficiency is virtualization [9]. As a result, depending on how much power is used, VMs can be moved, destroyed, and created among the host computers. Energy-efficient VM management is extended to task scheduling, workload consolidation, request batching, mobile service selection, choosing distant or local clouds, etc. [10], [11].

The current works concentrated on selecting VM utilizing metaheuristics and resource-saving methods such learning automata and the Analytical Hierarchy Process (AHP) [13], [14]. These methods, however, are only able to use the power in cases where computers are substantially loaded. The present research concentrated on VM migration taking into account target systems, QoS, and VM needs for target systems, as well as resource and CPU utilization. Consequently, the following is the primary contribution of this work:

- An upgraded meta-heuristic-based algorithm architecture for the selection of the virtual machines.
- An architecture for comparing the suggested meta-heuristic design to other cutting-edge algorithms from the same series.
- A combined training and classification architecture for the VMs' migration mechanism-in cloud.

1.1 VM allocation Problem

The notations are used to define the VM allocation problem. Imagine a cloud DC with p*q we simply multiply the number of rows (p) and the number of columns (q) of racks in the data center: racks that are all filled with hosts (h). To calculate the total number of hosts (T_H), use equation 1 [15].

$$T_H = p \times q \times h \tag{1}$$

The users U={u1,u2,u3….u_n}, where n is the total number of users, make up the cloud infrastructure. Users can utilize brokers or self-submit their requests on the cloud data centre. There are P physical machines (PMs) and V virtual machines (VMs) in the cloud datacenter. The PMs and VMs may be represented as a set with PMs = {p1, p2, p3…P} and VMs={v1,v2,…V}. The virtual machines (VMs) utilize the PMs' resources as they are linked to the PMs to carry out user requests. A higher work volume will put greater strain on the system, which will raise CPU and power utilization. According to research in the literature, CPU utilization accounts for the majority of [16]'s power usage. To do this, it is necessary to ascertain the direct correlation between the cloud host's CPU performance and power utilization. The calculation of the host's CPU power consumption in MIPS is used to calculate the cumulative utilization of resources of virtual machines assigned to a certain host machine. The CPU consumed by the host is computed by dividing the total CPU capacity of the associated VMs to

the CPU utilization of the host machine is given as follows: -

$$CPU_u = \sum_{j=1}^{Vk} \frac{v_{j_{cpu}}}{P_{i_{cpu}}} \quad (2)$$

Where k is total number of VMs associated with i^{th} PM.

Due to resource failures during execution, there is a reduction in VM performance, which results in SLA breaches (SLA-V) and VM downtime. Hence, the work computes the overall decrease of the virtual machine's performance. Based on the parameters given in equation (3), the virtual machines are moved from one PM to another as follows:

$$f(CPU) = \begin{cases} -1, & p_{i_{cpu}} \leq \min_{threshold} CPU_u \ \forall v \\ 1, & p_{i_{cpu}} \geq \max_{threshold} CPU_u \ \forall v \end{cases} \quad (3)$$

The function return -1 indicates that since the PM is not being used, all of the virtual machines from the current PM, p_i, will be moved. Here, 30% CPU utilization is referred to as minimum threshold and 70% as maximum threshold. Since every virtual machine (VM) needs to be moved, algorithmic architecture is not needed in this situation to choose which VMs to choose. In the second case, the function returns 1 when the current PM pm_i's CPU utilization is higher than the CPU's maximum utilization limit. The PM is deemed to be overworked in this circumstance, and some of the VMs are chosen to move the overworked PM into a normalized PM category.

The next sections of the study are organized as Section 2 to present the detailed literature review. Section 3 demonstrates the research approach and implements and evaluates optimization models for VM placement and migration. Section 4 presents the results, findings and discussion, and the paper is concluded in section 5 followed by list of references.

2. Literature Review

The optimization of VM placement and migration is essential for effective resource utilization and energy consumption reduction within cloud computing environments. This literature review critically examines various challenges and issues associated with VM migration, particularly in consideration of different Quality of Service (QoS) parameters. Key challenges often hinge on the continuity of network connections and the complexities surrounding the migration of data,

emphasizing storage and memory aspects. In a study by Ruan et al. (2019), a method for determining optimal operating utilization levels for host machines is proposed. The "PPR Gear" method, taking into account different sampling levels of utilization, calculates Performance-to-Power Ratios (PPR). The authors introduce a framework utilizing PPR for allocating and migrating VMs across different host types, ensuring the most power-efficient operation by balancing host utilization and energy consumption.

Wei, Hu, and Wang (2020) addressed the bin packing problem of idle and working machines, developing exact algorithms and best-fit strategies. Their experiment considered various data instances from data centers (DCs), evaluates algorithm performance in terms of computation time concerning both PMs and VMs. Different instances and variants were examined, including total energy consumption considerations. Talwani et al. (2022) focused on providing a suitable maximum number of resources to fulfill service level agreements. Leveraging virtualization as a crucial technology in cloud computing, they introduced a novel machine-learning-based method for dynamically integrating VMs based on adaptive forecasts of utilization thresholds. Ahmadi et al. (2022a) presented a flexible approach for addressing the challenge of VM selection in the cloud computing environment, employing a hierarchical-based decision-making process. Simulation analysis involving 1000 VMs demonstrates a 23% reduction in energy consumption and a 49% decrease in VM migrations, contributing to overall improved performance. In a hybrid optimization approach proposed by Khan and Santhosh (2022), the CS and PSO algorithms were combined to manage VM migration in a cloud environment. The research aimed to reduce energy consumption, calculation time, and migration expenses while maximizing resource use. Results indicated that the proposed technique achieves an energy consumption of 0.470 watts with a load of about 0.0025, highlighting the potential for further improvements in migration performance through multi-optimization techniques.

Table 1. Comparison of Existing VM Allocation Techniques

AUTHORS AND YEAR	FOCUS	METHODOLOGY	KEY FINDINGS
Ruan et al. (2019) [16]	Optimal utilization levels for host machines	"PPR Gear" method, balancing host utilization and energy consumption	Framework for VM allocation and migration based on PPR
Wei, Hu, Wang (2020) [17]	Bin packing problem for idle and working machines	Exact algorithms, best-fit strategies	Performance evaluation, considering computation time, PMs, and VMs
Aboamama and Hamouda (2020) [18]	Genetic Algorithm based VM placement	Simulation using MATLAB	Energy consumption, Wastage of resources, and elapsed run time
Tarahomi et al. (2021) [15]	Micro genetic approach	Simulation using the CloudSim toolkit	SLA violation, energy consumption, number of server shutdown
Talwani et al. (2022) [12]	Fulfilling service level agreements	Machine-learning-based method for dynamically integrating VMs	Determination of a suitable maximum number of resources
Ahmadi et al. (2022a) [14]	Flexible VM selection in the cloud	Hierarchical-based decision-making process	23% reduction in energy consumption, 49% fewer VM migrations
Khan and Santhosh (2022) [13]	Hybrid optimization for VM migration	Combination of CS and PSO algorithms	Energy consumption of 0.470 watts, load of about 0.0025, potential for further optimization

The literature mentioned above makes it evident that an energy-efficient VM placement strategy employing an optimization approach is required, as the majority of researchers are unable to offer an optimized VM placement technique. Some researchers restricted their use of the machine learning technology to placement that is effective. Therefore, it is evident from carefully reading the literature that an optimization approach is required to optimize the placement of virtual machines that are equipped with machine learning techniques.

3. Vm Movement and Placement in An Energy Efficient Manner

When a VM is assigned to a PM in a VMMP cloud DC, the energy usage is mostly during this process. This includes VM instantiation, memory allocation, cache warm-up, virtualization layer initialization and the initial I/O handshakes with storage and network subsystems, all of which add a measurable power spike. When a computational request is sent to the data center, it is deployed according to a set of settings that take into account the various computing resources, such as CPU use, execution time, and memory capacity. In practice, schedulers also consider bandwidth needs, topology awareness, thermal/power caps, and the current consolidation level of each host. After designating the VM to the PM, the execution procedure is complete. During runtime, the system periodically re-evaluates host utilization; if the VM's resource profile deviates from expected bounds, a migration decision may be triggered to restore balance and prevent SLA penalties.

The choice and migration of VM for effective placement forms the foundation of the proposed dragonfly-based optimization approach. For energy-efficient placement, optimization models like firefly, cuckoo search, ant colony optimization, and many more are built on the cloud. The provided efforts, however, are still restricted to discussing the corresponding characteristics for VM placement and selection for migration To ensure fair comparison, all methods are evaluated under the same workload mix, identical host capacities, and common measurement windows for energy and SLA..

This study applies many optimization methods, including PSO, ACO, CS, and dragonfly. These methods have been put into practice, and performance has been compared further to ascertain the optimal outcomes. Migration decisions account for their own costs—downtime, transient performance loss, and the extra energy consumed during state transfer—so that the net benefit remains positive. The following is listed as the primary issue during VM placement: - Two distinct circumstances are taken into consideration throughout the resource allocation process. Due to the fact that there are more user requests than there are slots available, the PM is seen to be overloaded in the first scenario. Additional computational resources can alleviate the overload in such a situation, but they will cost more and have a lower return on investment (RoI). Operationally, an upper utilization threshold flags overload; a minimal-impact subset of VMs is then selected for migration based on fitness and expected load reduction, and targets are chosen to satisfy capacity and SLA constraints.

In the second scenario, there are fewer VM requests, but the PM is still accessible to offer services. As a result, there is a condition of idle PM, also uses power to meet the needs of other users. Here a lower utilization threshold triggers consolidation: VMs are packed onto fewer hosts to curb idle energy, and freed PMs are transitioned to low-power or sleep states without affecting service delivery.

Fluctuating workload (oscillation) scenario: workloads may swing between under- and over-utilization in short intervals. To avoid repeated moves, the policy applies utilization bands (upper/lower thresholds), time-based dampening, and migration cooldowns so that only sustained imbalances trigger action.

.

Table 2 Simulation Parameters

S. No.	Parameter	Value
1.	Number of DC's	3
2.	Total number of PM's	10-55
3.	Total number of tasks	15-30
4.	Total number of VM's	50-500
5.	Memory (VM)	2 Gb
6.	Memory (Host)	4 Gb
7.	CPU capacity of Host (MIPS)	500-2500
8.	CPU capacity of VM (MIPS)	100-1000
9.	Bandwidth of VM	100 Mbit/sec
10.	Bandwidth of PM	1 Gbits/sec
11.	Gradient Value	6.48
12.	Workload coefficient	0.1x-0.4x

VM allocation and migration in cloud data centers is organized into a series of systematic phases. Each stage has been carefully designed to minimize energy consumption while reducing SLA violations and ensuring balanced workload distribution across physical machines (PMs). The complete workflow is summarized below:

- **Host State Classification:** The process begins with monitoring the utilization levels of all physical machines. A clustering method (k-means) is applied to classify the hosts into three distinct states: normal, overloaded, and underloaded. This classification is crucial because overloaded PMs cause performance degradation and SLA violations, whereas underloaded PMs consume energy inefficiently while hosting very few VMs.
- **Identification of Candidate Virtual Machines:** From each overloaded host, the virtual machines (VMs) that contribute most to excessive CPU utilization are identified. These VMs are grouped and represented as decision-making entities that form the candidate list for possible migration.
- **Dragonfly Algorithm:** The Dragonfly Algorithm (DF), a swarm intelligence-based meta-heuristic, is then employed to evaluate and select the best migration candidates.
- **Reward-Based Fitness Evaluation:** Each VM candidate is evaluated through a reward matrix mechanism. VMs that satisfy the fitness criteria are assigned higher rewards, while less efficient candidates receive lower values.
- **Migration Decision and Placement Strategy**: Once migration candidates are identified, they are mapped onto suitable target PMs. The placement strategy considers multiple constraints, including CPU capacity, memory limits, and SLA thresholds, to ensure that migrations do not create new hotspots or idle wastage.
- **Continuous Iteration and Consolidation**: The workflow is iterative in nature. As workloads in the data center are dynamic, the monitoring and selection process repeats periodically, allowing the system to adapt to changing demand patterns.

Algorithm 1: Proposed Dragonfly Algorithm

1. Input: HL, At where HL is the overload host list At is the allocation table
2. Output: V ML is the vm list to be migrated
3. [dg, dc] = kmeans(At, 3) // divide the allocation table into 3 states as normal, overloaded and underloaded.
4. Where dg is the dragon group and dc is the dragon centroid [global food]
5. For1(i = 1 : Len((H1))
6. vms = Find (AL, HLs[i]) //Find all vms of concerned host
7. drg = vms // consider vms as dragons
8. For2(j = 1 : Len(drg)
9. Dgj = dg[j]; find the state of current dargon
10. Lf = 10; // create a reward matrix that holds reward value for each fight
11. For3(k = 1 : if)
12. Ep = 30; exp = 60; where ep and exp is swarm size percentage of exploitation and exploration.
13. A = Cos(At[vms], C)/ Eucl(At[drp], C);
14. //Define alignment as the ratio of cosine similarity to Euclidean distance of the all vm's parameters to the global food parameter defined as cohesion
15. S = Cos(At[drp], C)/Eucl(At[drp], C); // Define separation as the ratio of cosine similarity to Euclidean distance of the group parameters to the global food parameters defined as cohesion.
16. [f, fv] = DragonFitness(A, S, C); where f is Boolean value for the fit unfit[1,0]
17. and fv is the fitness value from the fitness function.
18. If(f == 1) //Assign reward for the flight.
19. R[k] = 100 − fv; //Assign reward for the flight
20. Endif
21. Rm = Mean(R) // Compute mean of rewards
22. If (Rm ≤ 60) Vml. append(drg[j]); //ThisVMisselectedforthrmigrationEndif
23. Define DragonFitness[A, S]

24. F, fv = 0
25. If(fv = (A − S)/A) ≤ .30f; Endif
26. Return f, fv
27. End DragonFitness
28. EndFor2
29. EndFor1
30. Return vml

The proposed dragonfly algorithm is used to formulate the following feature:

The constraints of PM's, such as CPU and Memory capacity utilization, are taken into account when placing VMs in PMs, and are written as ($CPU^{VM} \leq CPU^{PM}$) and ($M^{PM} \leq M^{VM}$) for all practical VM placements (x_{PmVm}). It is completely pre-emptive, meaning that a PM (Pm) may only host one VM at a time, $\sum_{Vm} x_{PmVm} \leq 1$" at any given moment.It is made aware that the procedure must be ON before all VMs are put when the placement imitates. The earliest time a VM for a PM may be completed is called the Makespan time $\sum_{Vm} T_{Vm}^{Vm} x_{PmVm}$. Two separate forms of energy are taken into account in the suggested work scenario: processing energy (A_{energy}), which is used while placing VMs on the PM, and idle energy (I_{energy}), which is used when the PM is operational but not hosting any VMs. Total energy consumption is calculated as the product of active and idle energy ($T_{energy} = A_{energy} + I_{energy}$).The processing energy A_{energy} is influenced by the VM's execution time (T_{Vm}^{Vm}) and energy efficiency [(E_{PmVm}].The energy efficiency for the positioning of the VM to the PM is given as follows, with the least and greatest amounts of energy used by the PM over the course of an hour represented as [E^{-}_{Pm}, E^{+}_{Pm}] Energy consumption by PM when in an idle state is also based on idle time and the minimal energy intake per hour.

$$E_{PmVm} = E^{-}_{Pm} + (E^{+}_{Pm} - E^{-}_{Pm}).\, e^{CPU_{PM}^{CPUVM}} \quad (4)$$

To calculate idle power usage, the m[th] VM and PM are both assumed to be idle.

$$E_{VM_i}^{idle} = \begin{cases} Power_{server_l}^{idle} & \exists \frac{z}{U_{VM_i}^{z}} = 100\% \\ \frac{\sum_z \alpha_z . U^{z}_{VM^i}}{\sum_z \alpha_z} . Power_{server_l}^{idle} & otherwise \end{cases} \quad (5)$$

Where $_{VM_i^i}$ is the utilization of resource z by ith VM and α_z is the weight given to resource z. This means that if a virtual machine is fully used, the amount of idle power it consumes is comparable to the amount of idle power consumed by servers.

To accomplish the stated effects, the proposed work modifies the dragonfly fitness function. The suggested dragonfly algorithm is used to calculate the PM's effectiveness in hosting the VM. the effective use of PM to prevent energy waste. The suggested optimization method is also contrasted with other state-of-the-art algorithms, which are provided in the following sections, in order to help the reader, understand the proposed algorithm. Table 3 [19] provides an illustration of the DF algorithm's parametric parameters.

Table 3 Parameter settings for Dragonfly Algorithm

Parameter	**Value**
Number of Simulation Rounds	50
Number of search agents	5
Search Domain	[0 1]
Dimension	Number of features acquired in the data
Number of runs	10

In the process, three the nature-inspired optimization algorithms Cuckoo Search Algorithm, Particle Swarm Optimization, and Ant Colony Optimization are utilised to justify the effecctivness of DF to tackle challenging optimization issues. Every single one of these algorithms has special traits and ways of locating the best answers. These three strategies will be covered in this section along with their key aspects.

3.1.1. Cuckoo Search (CS) Optimization

A number of eggs are employed in the CS method, which was inspired by nature, to symbolize the PM [35]. A set of VMs is represented by (1–12) for four different PMs, such as "R," "S," "U," and "W." R={1,2,3,4}, S={ 5,6,7,8}, U={9,10,0,0},W={11,12,0,0}. Therefore, the symbols for nest are 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 0, 0, and 11, 12, 0, 0 [36]. A metaheuristic algorithm called CS was created with the behavior of cuckoos in mind. Given their incredible talents, such as putting eggs in robust nests or choosing other eggs to be their own eggs, cuckoos have a tendency to deposit their eggs in other people's nests. In no time, and with a high degree of precision for laying eggs, parasitic cuckoos wandered to locate nests where other cuckoos deposit their eggs. When this happens, the other cuckoo will take the eggs out of the nest, which lowers the likelihood that they are real eggs.

- Levy Flights: Levy flights, a sort of random walk, are used as a major component of CSA to explore the search space. This allows global exploration and aids in escape local optima.
- Nest Selection: An ongoing population of nests (solutions) is maintained by the algorithm. Levy flights, which imitate brood parasitism, are used to replace certain nests with fresh ones throughout the search process.
- Nest abandonment: This conduct resembles the cuckoos' rejection of host nests. Better solutions replace some of the earlier ones, which aids in focusing the search on promising regions.
- Global and Local Search: CSA is suited for a variety of optimization issues since it combines global exploration with local exploitation.

3.1.2 Ant Colony Optimization (ACO)

The VM placement problem in the cloud context is solved by the metaheuristic algorithm known as ACO [20]. The researchers alter the swarm algorithm's behavioral design since creating a new swarm series algorithm would need many academics from many domains. The ACO approach uses ant foraging behavior to determine the optimal path. The pheromone is released by the ants as they go along their course while foraging. The fundamental ACO work is tailored in a number of studies, since it has already been demonstrated that the authors adopt behavioral modification in the algorithm design [21]. Ants are used as VMs in the selection process, and in the majority of the instances examined, PC has been seen as a crucial criterion for migration. The salient features of this algorithm are as follows:

- Ants leave pheromone trails to communicate with one another and discover the quickest route to food sources, and this is how ACO was inspired.
- The quality of solutions is represented by ACO using pheromone trails. Ants leave behind and travel along these trails to skew the search in favour of routes with higher pheromone concentrations.
- ACO combines local exploration (ants modifying pheromone levels along their present journey) and global exploration (pheromone evaporation and updating in accordance with the best solutions discovered).
- Positive feedback loops are created as a result of the ants' preference for paths with higher pheromone concentrations, which encourages the selection of better solutions over time.
- ACO is flexible and can be used to resolve combinatorial optimization issues. It works especially well for issues with network routing, scheduling, and travelling salesperson issues.

3.1.3 Particle Swarm Optimization (PSO)

The social behaviour of fish and birds, in which members of a group search cooperatively for the greatest potential food sources or sites, served as the model for PSO.

The salient features of PSO are as follows:

- Particle Movement: PSO makes use of a population of moving particles in the search space. Based on its own experience and the experience of its neighbours, each particle modifies its position.
- Updates to Velocity and Position: Based on their previous best-known solution and the best-known solution for the entire swarm, particles change their velocity and position.
- Global Exploration: Early in the search phase, PSO places a strong emphasis on global exploration by modifying particle velocities to go towards promising areas of the search space.
- Convergence: By settling on the particles' most well-known solutions over time, PSO gradually turns its attention to local exploitation.

4. Result and Discussion

A comprehensive set of simulation tests has been carried out to assess the effectiveness of the suggested secure

cloud model. MATLAB was used for the simulation, along with an Intel Core i3 CPU running at 2.30 GHz oscillator frequency, a 64-bit operating system, and 4 GB of RAM. We have two players in this experiment: one is a cloud customer, and the other is a cloud server. The cloud user behaves as an authorized user and functions as the owner of the data. Conversely, a cloud server functions similarly to a cloud service provider. The evaluation of the developed model's performance is detailed below.

4.1 Performance Analysis

The following assessment criteria are used to assess the proposed work.

- Energy Consumption: This is determined by adding up all the power used for tasks like VM migration and allocation that keep the SLA met.
- SLA-V: This is the ratio of power used to intended use and service level violations.

Table 4 Comparative Analysis for Energy consumption (kWh) using different optimization techniques

Total VM	Total PM	Proposed Dragonfly Technique (kWh)	CS Technique (kWh)	ACO Technique (kWh)	PSO Technique (kWh)	DF Technique (kWh)
50	10	8.62	9.41	9.55	9.68	9.47
100	15	8.88	9.72	9.83	9.97	9.76
150	20	8.73	9.58	9.69	9.84	9.62
200	25	8.95	9.81	9.93	9.06	9.85
250	30	8.54	9.39	9.52	9.65	9.44
300	35	8.69	9.51	9.64	9.79	9.58
350	40	8.83	9.66	9.77	9.91	9.71
400	45	8.57	9.42	9.55	9.70	9.49
450	50	8.46	9.29	9.41	9.56	9.36
500	55	8.38	9.21	9.34	9.48	9.28

The comparison of various methods to ascertain the amount of energy used by the PM and VM in the datacenters is displayed in Table IV and Figure 2. The energy consumption of the dragonfly technique is 8.67 kWh on average, but the energy consumption of the CS, ACO, PSO, and DF techniques are 9.50 kWh, 9.62 kWh, 9.76 kWh, and 9.56 kWh, respectively. The workload is accessible.

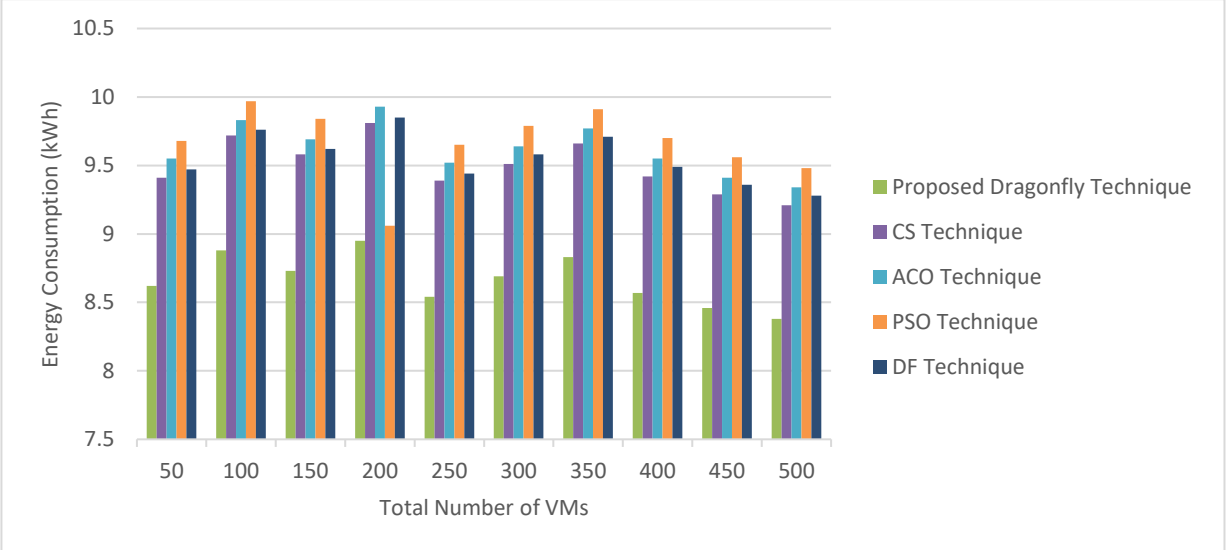

Fig 2 Comparative Analysis for Energy Consumption (kWh)

The analysis's findings demonstrate that the suggested approach outperforms alternative methods. As a result, the dragonfly strategy for VM placement uses less energy and improves upon the CS, ACO, PSO, and DF techniques by approximately 8.74%, 9.88%, 11.16%, and 9.31% respectively.

Table 5 Comparative Analysis for SLA Violation using different optimization techniques

Total VM`	Total PM	Proposed Dragonfly Technique	CS Technique	ACO Technique	PSO Technique	DF Technique
50	10	0.088	0.101	0.110	0.126	0.107
100	15	0.094	0.108	0.118	0.125	0.117
150	20	0.096	0.106	0.120	0.127	0.115
200	25	0.101	0.114	0.129	0.132	0.121
250	30	0.098	0.113	0.127	0.135	0.120
300	35	0.104	0.122	0.124	0.144	0.125
350	40	0.108	0.119	0.129	0.142	0.134
400	45	0.109	0.120	0.135	0.147	0.131
450	50	0.112	0.126	0.133	0.143	0.132
500	55	0.115	0.130	0.139	0.149	0.139

The comparison of various methods for SLA violations resulting from contract negotiations between users and service providers in datacenters is displayed in Table V and Figure 3. The suggested dragonfly method results in an average SLA violation of 0.103, compared to 0.116, 0.126, 0.137, and 0.124 for the CS, ACO, PSO, and DF techniques. The analysis's findings demonstrate that the dragonfly method outperforms other methods. Consequently, the suggested dragonfly approach improves upon the CS, ACO, PSO, and DF techniques by 11.20%, 18.25%, 24.82%, and 16.94%, respectively, in managing the breach in service level. The dragonflies' heuristic search mechanism and energy-efficient method are to blame for this improvement.

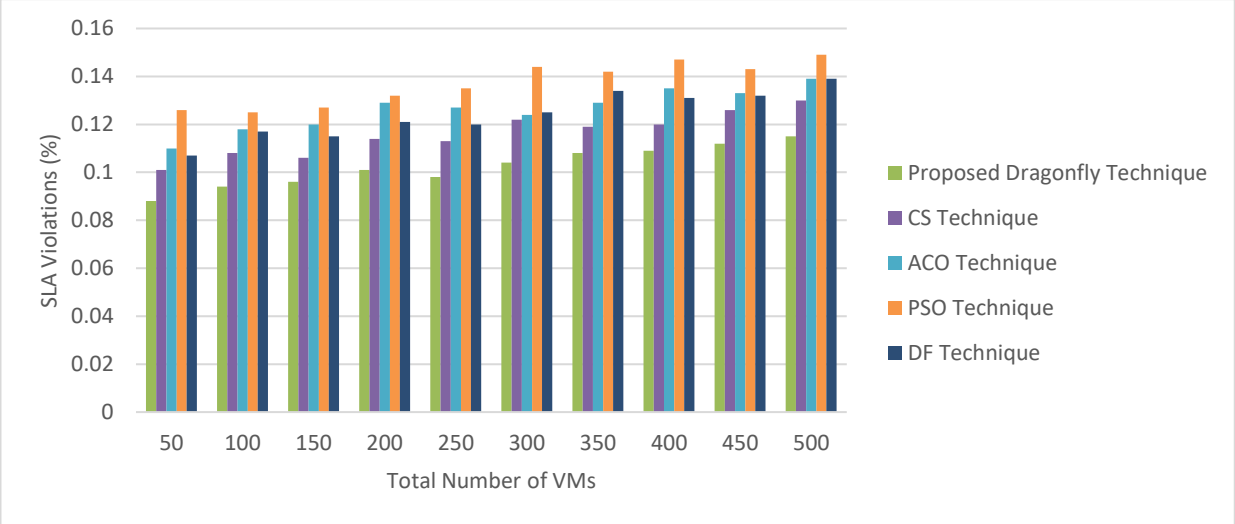

Fig 3 Comparative Analysis for SLA Violation

doi:10.1002/CPE.7636, (2023).

5. Conclusion

The VM allocation and migration problem is presented in this study taking various optimization strategies into consideration. Several optimization algorithms, including dragonfly, CS, ACO, and PSO, are used to allocate virtual machines with the least amount of resource waste possible. Across heterogeneous PM–VM settings, the proposed Dragonfly approach delivers the lowest energy consumption (avg 8.67 kWh), improving over CS, ACO, PSO, and DF by 8.74%, 9.88%, 11.16% and 9.31% respectively. It also achieves the lowest SLA violation (avg 0.103), reducing violations by 11.20%, 18.25%, 24.82%, and 16.94% relative to CS, ACO, PSO, and DF. These consistent gains show that Dragonfly offers a balanced improvement in efficiency and service quality for cloud datacenters. Future work will integrate multiclass meta-heuristics and ML-based workload prediction to further refine placement decisions and QoS.

References

[1] Kaur, T., & Chana, I. (2015). Energy efficiency techniques in cloud computing: A survey and taxonomy. ACM Computing Surveys (CSUR), 48(2), 1–46 (2015).

[2] Askarizade Haghighi, M., Maeen, M., & Haghparast, An energy-efficient dynamic resource management approach based on clustering and meta-heuristic algorithms in cloud computing IaaS platforms. Wireless Personal Communications, 104(4), 1367–1391 (2019).

[3] Masanet, E., Shehabi, A., Lei, N., Smith, S., & Koomey, J., Recalibrating global data center energy-use estimates. Science, 367(6481), 984–986 (2020).

[4] Ferreira, L., Rocha, E. S., Monteiro, K. H. C., Santos, G. L., Silva, F. A., Kelner, J., Sadok, D., et al. (2019). Optimizing resource availability in composable data center infrastructures. In 2019 9th Latin-American Symposium on Dependable Computing (LADC), 1–10. IEEE (2019).

[5] Saxena, D., Gupta, I., Kumar, J., & Wen, X. (2021). A Secure and Multi-objective Virtual Machine Placement Framework for Cloud Data Centre. arXiv preprint arXiv:2107.13502 (2021).

[6] Farrag, Aya A. Salah, Mohamad, Safia Abbas, & El Sayed, M. (2019). Swarm intelligent algorithms for solving load balancing in cloud computing. Egyptian Computer Science Journal, 43(1), 45–57 (2019)

[7] Ghasemi, A., Haghighat, A.T.: A multi-objective load balancing algorithm for virtual machine placement in cloud data centers based on machine learning. Computing, https://doi.org/10. 1007/s00607-020-00813-w (2020).

[8] Masdari, M., Nabavi, S. S., & Ahmadi, V.An overview of virtual machine placement schemes in cloud computing. Journal of Network and Computer Applications, 66, 106-127 (2016).

[9] Shukur, H., Zeebaree, S., Zebari, R., Zeebaree, D., Ahmed, O., & Salih, A., Cloud computing virtualization of resources allocation for distributed systems. Journal of Applied Science and Technology Trends, 1(2), 98–105 (2020).

[10] Saadi, Youssef, & El Kafhali, Said. (2020). Energy-efficient strategy for virtual machine consolidation in cloud environment. Soft Computing, 24(19), 14845–14859 (2020).

[11] Zhou, Qiheng, Xu, Minxian, Gill, Sukhpal Singh, Gao, Chengxi, Tian, Wenhong, Xu, Chengzhong, & Buyya, Rajkumar. (2020). Energy efficient algorithms based on VM consolidation for cloud computing: comparisons and evaluations. In 2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID), 489–498. IEEE (2020).

[12] Talwani, S., Singla, J., Mathur, G., Malik, N., Jhanjhi, N. Z., Masud, M., & Aljahdali, S.Machine- Learning-Based Approach for Virtual Machine Allocation and Migration. Electronics, 11(19), 3249 (2022).

[13] Khan, M. S. A., & Santhosh, R. Hybrid Optimization Algorithm for VM Migration in Cloud Computing. Computers and Electrical Engineering, 102,108152 (2022).

[14] Ahmadi, J., Toroghi Haghighat, A., Rahmani, A. M., & Ravanmehr, R. A flexible approach for virtual machine selection in cloud data centers with AHP. Software: Practice and Experience, 52(5), 1216-1241 (2022).

[15] Tarahomi, M., Izadi, M., & Ghobaei-Arani, M.An efficient power-aware VM allocation mechanism in cloud data centers: a micro genetic-based approach. Cluster Computing, 24(2), 919-934 (2021).

[16] Rahman, C. M., & Rashid, T. A. Dragon fly algorithm and its applications in applied science survey. Computational Intelligence and Neuroscience (2019).

[17] Sutar, S. G., Mali, P. J., & More, A. Y. Resource utilization enhancement through live virtual machine migration in cloud using ant colony optimization algorithm. International Journal of Speech Technology, 23(1),79-85 (2020).

[18] Shabeera, T. P., Kumar, S. M., Salam, S. M., & Krishnan, K. M. Optimizing VM allocation and data

placement for data-intensive applications in cloud using ACO metaheuristic algorithm. Engineering Science and Technology, and International Journal, 20(2), 616-628 (2017).

[19] Ruan, X., Chen, H., Tian, Y., & Yin, S.Virtual machine allocation and migration based on performance-to-power ratio in energy-efficient clouds. Future Generation Computer Systems, 100:380-394 (2019).

[20] Wei, C., Hu, Z.H., Wang, Y.G.: Exact algorithms for energy efficient virtual machine placement in data centers. Future Gener. Comput. Syst., 77–91 (2020).

[21] Abohamama, A.S., Hamouda, E.: A hybrid energy-aware virtual machine placement algorithm for cloud environments. Expert Syst. Appl. 150, 113306 (2020).