Efficient Resource Management in Cloud Environments: A Modified Feeding Birds Algorithm for VM Consolidation

Deafallah Alsadie * and Musleh Alsulami

Posted Date: 7 June 2024

doi: 10.20944/preprints202406.0500.v1

Keywords: Cloud data centers; Virtual machine consolidation; power efficiency; Artificial Feeding Birds Algorithm; Cost management

Article

Efficient Resource Management in Cloud Environments: A Modified Feeding Birds Algorithm for VM Consolidation

Deafallah Alsadie [1,*] and Musleh Alsulami [2]

[1] Department of Computer Science and Artificial Intelligence, College of Computing, Umm Al-Qura University, Makkah 21961, Saudi Arabia
[2] Department of Software Engineering, Umm Al-Qura University, College of Computing, Makkah 21961, Saudi Arabia; mhsulami@uqu.edu.sa
* Correspondence: dbsadie@uqu.edu.sa

Abstract: Cloud data centers play a vital role in modern computing infrastructure, offering scalable resources for diverse applications. However, managing costs and resources efficiently in these centers has become a crucial concern due to the exponential growth of cloud computing. User applications exhibit complex behavior, leading to fluctuations in system performance and increased power usage. To tackle these obstacles, we propose the consolidation of virtual machines (VMs) to optimize resource utilization and reduce operational costs. This paper introduces a novel algorithm, leveraging the Modified Artificial Feeding Birds Algorithm (ModAFBA), to manage cost optimally in cloud data centers by reducing power usage while meeting service level agreements (SLAs). The ModAFBA algorithm, inspired by bird foraging behavior, dynamically adjusts solutions based on environmental changes. Comprehensive experimental evaluation demonstrates the superiority of the proposed algorithm in terms of power usage, migration count, and SLA compliance. Through this research, we aim to advance cost management and resource optimization in cloud data centers, providing a valuable tool for cloud service providers to enhance infrastructure efficiency and achieve cost savings.

Keywords: cloud data centers; virtual machine consolidation; power efficiency; artificial feeding birds algorithm; cost management

1. Introduction

Cloud data centers are the backbone of modern computing infrastructure, providing scalable and on-demand resources to support a myriad of applications and services [1,2]. The rapid expansion of cloud computing utilization across diverse sectors has underscored the vital importance of effectively managing expenses and resource utilization within data centers for cloud service providers. User applications and workloads in cloud environments exhibit highly non-linear behavior, leading to fluctuations in overall system performance, increased power usage, and degradation in QoS [1,3,4]. For instance, across various cloud platforms such as Azure, Alibaba, IBM, Google Cloud Platform (GCP), and Amazon Web Services (AWS), CPU and memory utilization vary significantly [5,6]. However, it is noteworthy that the utilization of CPU and memory in the majority of cloud service providers remains below 60%, suggesting a significant underutilization of available resources [5]. In response, cloud service providers (CSPs) are committed to upholding QoS standards and providing effective services to end-users, thereby emphasizing the importance of avoiding any violations of performance requirements [7,8].

Costly, high-performance computers designed to satisfy demand-driven service requirements are housed in cloud computing data centers [9,10]. Despite this, their power consumption, alongside that of their cooling systems, remains substantial, even during periods of low workload activity [11]. Conversely, in settings characterized by a scarcity of servers but high workloads, physical machines might experience over-utilization due to the demand for VMs. Addressing these challenges involves VM consolidation, a method that dynamically reallocates VMs to optimize resource utilization and curtail operational expenses.

 doi:10.20944/preprints202406.0500.v1

Numerous investigations have proposed that effective resource management could alleviate the overall energy consumption of data centers [1,12]. Various efforts have been undertaken to curtail energy usage by employing efficient resource utilization through techniques such as VM consolidation and placement [13]. The objective of these methods is to consolidate active VMs onto the fewest possible physical machines.

Traditional VM consolidation methods often face challenges in achieving optimal resource allocation due to the dynamic and heterogeneous nature of cloud environments. VM consolidation entails the dynamic relocation of VMs from one physical machine to another, ideally to optimize resource allocation [13]. By enabling inactive physical machines to enter sleep mode, this approach conserves energy. Effective VM consolidation techniques must consider resource utilization, energy consumption, QoS requirements, and strive to optimise VM migration frequency [11].

Conversely, VM placement strategies prioritize dispersing VMs from different applications among a collection of physical machines. Effective VM placement (VMP) strategies aim to minimize energy usage, reduce resource waste in physical machines, and maximize Quality of Service (QoS) for cloud users [14]. To address the limitations of conventional methods, this study proposes a novel hybrid algorithm for VM consolidation that incorporates a modified Artificial Feeding Birds Algorithm (ModAFBA). Inspired by the foraging behavior of birds, the ModAFBA algorithm has demonstrated potential in optimization tasks due to its adaptability to environmental changes. The main objective of the hybrid algorithm is to enhance cost efficiency in cloud data centers by reducing energy usage, ensuring compliance with service level agreements (SLAs), and upholding robust system performance. The ModAFBA approach is designed to enhance the efficiency and effectiveness of VM consolidation processes.

In this paper, we provide a comprehensive description of the proposed ModAFBA algorithm, including its formulation, implementation details, and experimental evaluation. We demonstrate the effectiveness of our approach through extensive simulations conducted in a cloud data center environment. Our experimental findings demonstrate the superior efficacy of the hybrid algorithm when compared to existing VM consolidation methods across various metrics including energy consumption, migration count, and compliance with SLAs within energy usage constraints. This research endeavors to advance cost management and resource optimization in cloud data centers. By presenting the ModAFBA algorithm for VM consolidation, we offer cloud service providers a valuable tool to enhance the efficiency and effectiveness of their infrastructure deployments, ultimately resulting in improved performance and cost savings.

The main contributions of this paper can be distilled as follows:

- Proposal of an energy efficient algorithm for VM consolidation in cloud data centers, leveraging the Modified Artificial Feeding Birds Algorithm (ModAFBA).
- Addressing key challenges faced by traditional VM consolidation techniques, such as achieving optimal resource allocation in dynamic and heterogeneous cloud environments.
- Comprehensive experimental evaluation of the ModAFBA algorithm, including formulation, implementation details, and performance assessment in a simulated cloud data center environment.
- Advancement in cost management and resource optimization in cloud data centers, contributing to enhanced performance and cost savings for cloud service providers.
- " Potential for practical implementation and adoption by cloud service providers, offering a scalable and efficient solution to VM consolidation challenges.

The subsequent sections of this paper are organized as follows: Section 2 provides an extensive review of relevant literature, identifying areas requiring further exploration. Subsequently, Section 3 outlines the problem formulation and the foundational models utilized in our study. In Section 4, we provide a detailed exposition of the customized ModAFBA algorithm developed to optimize VM consolidation, aiming to achieve energy efficiency while upholding SLA standards in computing environments. Following that, Section 5 elaborates on the experimental methodology and configuration.

 doi:10.20944/preprints202406.0500.v1

Section 6 presents the results obtained from the experiments and offers a thorough analysis of the findings. Finally, Section 7 summarizes the key insights derived from this research endeavor.

2. Related Work

Several investigations have delved into strategies for effective resource administration aimed at diminishing the collective energy usage of cloud data centers. Diverse methodologies rooted in VM consolidation and placement have emerged to curtail power consumption through the optimization of resource utilization.

A sophisticated Levy-based Multi-Objective Gray Wolf Optimization (LMOGWO) algorithm was presented by Fatima et al. [15] and specifically designed to tackle the problem of virtual machine placement. Their method combines a grid mechanism to improve the non-dominated virtual machines (VMs) within the archive with an archive for storing and retrieving the real Pareto front. The archive also has a maintenance mechanism included into it. Motivated by the hunting and leadership styles of gray wolves, the suggested algorithm skillfully traverses the multi-objective search space. The algorithm is evaluated on nine popular bi-objective and tri-objective benchmark functions to verify its effectiveness. Comparisons are made with Multi-Objective Particle Swarm Optimization (MOPSO) and Simple Multi-Objective Gray Wolf Optimization (MOGWO). To assess adaptivity, two scenarios are simulated. For the UF1, UF5, UF7, and UF8 benchmarks, LMOGWO in Scenario 1 performs better than MOGWO and MOPSO, but worse than both for UF2. In the second case, MOGWO performs well for UF2 and UF4, whereas LMOGWO performs better for the UF5, UF8, and UF9 benchmarks. Moreover, the suggested algorithm lowers the PM usage rate by 30%, outperforming MOPSO (10%) and MOGWO (11%) in this regard.

Zhou et al. [16] introduced a novel algorithm named the Energy-Efficient VM Cluster (EVCT) algorithm, with the aim of enhancing Quality of Service (QoS) and reducing energy consumption in software-defined data centers, particularly catering to IoT applications like Connected and Autonomous Vehicles (CAVs). Their method involved conceptualizing the VM placement challenge by leveraging VM similarity and principles from maximum flow and minimum cut theory, resulting in the construction of a weighted directed graph. Additionally, factors such as SLA violations and communication bandwidth are also taken into account to ensure energy efficiency and QoS requirements. The research findings illustrated the efficacy of the EVCT algorithm in decreasing energy usage, expenses, and SLA breaches within the software-defined data center context.

A strategy for improving virtual resource consolidation using a tailored particle swarm optimization algorithm to improve QoS and energy efficiency in cloud data centers was provided by Li et al. [17]. In order to effectively distribute resources within cloud data centers, their approach sought to reduce power usage per QoS value. However, as their approach relied on randomly generated workloads that came simultaneously at the start of the simulation, a major flaw in it was the absence of a prediction mechanism to guarantee QoS for subsequent requests. However, a noteworthy feature of their work was how they included a user's degree of satisfaction as a parameter in the QoS model, providing a more accurate depiction of user perception and expectations.

Likewise, Guo et al. [18] presented a shadow routing methodology aimed at optimizing resource allocation within cloud environments. Through intelligent VM-to-PM packing, their approach efficiently mitigates resource wastage, with a particular emphasis on energy and cost reductions. In order to meet the needs of dynamic applications and users, the study also tackles VM placement by implementing VM auto-scaling methodologies. Interestingly, the proposed technique is adaptive, meaning that it doesn't require starting from scratch to solve the optimization problem. The effectiveness of the suggested strategy is shown by the experimental findings. But it's important to understand that VM placement is an NP-hard problem, and given the complexity of cloud data centers, more improvements are required to effectively tackle this difficulty.

A multi-objective genetic algorithm (GA) was presented by Riahi et al. [19] in order to tackle the virtual machine placement problem and lower the physical server utilization rate in cloud environ-

 doi:10.20944/preprints202406.0500.v1

ments. Furthermore, they utilized Bernoulli simulations to evaluate the adaptability of their proposed approach. The primary goal of their work was to mitigate resource wastage within cloud infrastructures, and they proposed a framework to achieve this objective. Through real-time experiments, they successfully optimized the VM placement problem within a targeted company, resulting in reduced operational costs. However, their approach lacked a mechanism to efficiently handle big data, which presents a significant challenge due to its rapid growth rate. Addressing this issue is crucial to ensuring the scalability and effectiveness of cloud-based solutions.

In order to address the VM consolidation problem, the study described in [20] introduces EE-HVMC, a heuristic-based technique. This technique seeks to lessen SLA violations and cut down on power usage in cloud systems. Using a configurable utilization threshold, the EEHVMC technique first divides hosts into three primary groups: Under-Loaded (HUL), Medium-Loaded (HML), and Over-Loaded (HOL) machines. To save energy, virtual machines are then moved from one physical host to another.

Moreover, Zhang et al. [21]have addressed the VM consolidation challenge on cloud platforms. Their study presents a method that uses constraint programming to control the distribution of virtual resources while accounting for the price of virtual cloud resources and the quality of service requirements of users. Nevertheless, results from experiments indicate that the suggested model-which relies on constraint and linear programming-becomes less effective as the number of variables-like the number of VMs and PMs-increases.

The study outlined in [22] introduces an innovative adaptive heuristic-based algorithm designed to uphold SLA requirements while minimizing energy consumption in data centers. This method leverages historical data to analyze VM resource utilization, thereby establishing a dynamic consolidation model. By employing various statistical methods on collected historical data, the approach identifies overloaded and underloaded hosts. In addition, it suggests three policies to manage the placement and selection of virtual machines: Minimum Migration Time, Random Choice, and Maximum Correlation. Finally, it uses a modified version of the Best Fit Decreasing (BFD) algorithm called Power Aware Best Fit Decreasing (PABFD) to allocate hosts with the lowest necessary energy consumption.

In a recent study outlined in [23], an innovative methodology called normalization-based VM consolidation (NVMC) is introduced, with the aim of enhancing the VM consolidation process in cloud platforms. This approach places emphasis on reducing energy consumption, meeting SLA requirements, and minimizing VM migrations. Using a variety of resource metrics from both hosts and virtual machines, NVMC finds overused hosts and determines the best location for virtual machines based on the total available-to-total ratio. Additionally, threshold values are incorporated into the NVMC framework to efficiently detect over-utilized hosts and streamline the consolidation process.

The Hybrid VM Allocation and Placement Policy (HVMAP), created for VM consolidation in cloud data centers, is presented by the research described in [24]. To identify overloaded hosts, this method combines migration control with an improved host overload detection algorithm. In addition, it uses the PEBFD placement algorithm to identify appropriate destination hosts for virtual machine migration. Throughout the consolidation process, lowering energy use while keeping SLA compliance is the major goal. A novel method called utilization-aware VM placement (UAVMP) is proposed in the paper by [25] to improve VM placement in cloud platforms throughout the consolidation process. In summary, the dynamic and varied nature of cloud systems makes it difficult for standard VM consolidation techniques to achieve effective resource allocation. These methods typically involve dynamically migrating VMs between physical machines to achieve efficient resource utilization while ensuring quality of service (QoS) requirements. Conversely, VM placement techniques focus on assigning VMs from different applications to a set of physical machines, aiming to minimize energy consumption and maximize QoS for cloud users. However, existing techniques may not fully address the complex optimization requirements of modern cloud data centers. To overcome these limitations, this paper proposes a modified AFBA algorithm for VM consolidation, inspired by the foraging behavior of birds. This innovative strategy aims to enhance cost management in cloud data centers by

 doi:10.20944/preprints202406.0500.v1

decreasing energy consumption, guaranteeing adherence to SLAs, and maintaining optimal system performance.

Table 1 provides an overview of the discussed studies, outlining their approach, optimization objectives, focus area, performance metrics, strengths, and limitations.

Table 1. Comparative Analysis of VM Consolidation Approaches in Cloud Computing Environments.

Study	Approach	Optimization Objectives	Focus Area	Performance Metrics	Strengths	Limitations
Fatima et al. [15]	LMOGWO algorithm	VM placement optimization	Optimization of energy efficiency and QoS	Comparative analysis against MOGWO and MOPSO, PM utilization rate minimization	Effective archive mechanism, adaptive algorithm based on gray wolf behavior	Limited scalability due to PM utilization rate minimization
Zhou et al. [16]	EVCT algorithm	QoS and energy consumption optimization	Software-defined data centers, IoT applications	Reduction of energy consumption, costs, and SLA violations	Utilization of similarity modeling and graph theory, effective consideration of communication bandwidth	Complexity in constructing weighted directed graphs
Li et al. [17]	MPSO algorithm	Energy efficiency and QoS enhancement	Cloud data centers	Minimization of power consumption per QoS value	Integration of user satisfaction degree in QoS model	Absence of prediction mechanism for future requests
Guo et al. [18]	Shadow routing approach	Resource allocation optimization	Cloud environments	Mitigation of resource wastage, energy, and cost savings	Adaptive algorithm, incorporation of VM auto-scaling strategies	Complexity due to NP-hard VM placement problem
Riahi et al. [19]	Multi-objective GA	Resource utilization optimization	Cloud environments	Reduction of physical server utilization rate	Real-time optimization in targeted company	Inefficiency in handling big data
Karmakar et al. [20]	EEHVMC mechanism	Power consumption minimization and SLA compliance	Cloud environments	Utilization-based host classification for VM reallocation	Effective resource utilization	Lack of consideration for dynamic workload changes
Zhang et al. [21]	Constraint programming approach	Virtual resource allocation optimization	Cloud platforms	Consideration of QoS requirements and cost	Utilization of constraint programming concepts	Ineffectiveness with increasing number of variables
Beloglazov et al. [22]	Adaptive heuristic-based algorithm	SLA compliance and energy consumption optimization	Data centers	Dynamic consolidation model based on historical data	Effective VM placement policies	Lack of consideration for dynamic workload changes
Wu et al. [23]	VM consolidation strategy	Energy consumption and migration cost minimization	Heterogeneous cloud settings	Incorporation of score function and enhanced genetic algorithm	Effective energy consumption reduction	Limited scalability due to score function complexity
Wu et al. [24], Sonklin et al. [25]	DCGA	VM placement optimization	Cloud environments	Ensuring the integrity of solution quality while simultaneously decreasing the problem's scale and the number of VM migrations	Effective solution quality maintenance	Complexity in solving NP-hard VM placement problem
Ye et al. [26]	EEKnEA	VM placement optimization	Cloud platforms	Optimization of energy efficiency	Effective energy consumption reduction	Lack of consideration for communication network energy consumption
Javadi et al. [6]	NVMC strategy	VM consolidation optimization	Cloud platforms	Energy consumption reduction, SLA compliance, and VM migration minimization	Utilization of resource parameters and threshold values	Dependency on accurate resource parameter estimation
Radi et al. [8]	HVMAP	VM consolidation optimization	Cloud data centers	Energy consumption minimization and SLA compliance	Enhanced host overload detection algorithm	Complexity in VM migration decision-making
Saif et al. [7]	UAVMP technique	VM placement optimization	Cloud platforms	Optimization of VM placement based on resource utilization	Effective resource utilization	Limited scalability due to resource utilization optimization

3. Underlying Models and Problem Formulation

In this section, we present the foundational models and problem formulation concerning VM consolidation within cloud data centers. The efficient allocation of resources in cloud environments

 doi:10.20944/preprints202406.0500.v1

is crucial for optimizing performance and minimizing operational costs. To address this challenge, we delineate the key components of VM consolidation, including resource utilization models, migration policies, and QoS requirements. Additionally, we formulate the VM consolidation problem as an optimization task, aiming to dynamically allocate VMs to physical machines while considering factors such as energy consumption, SLA compliance, and workload fluctuations. By establishing a comprehensive understanding of the underlying models and problem formulation, we pave the way for the development of effective consolidation algorithms and strategies in subsequent sections.

3.1. System Model

In this section, we provide a detailed description of the system model used in our study. The system model encompasses the architecture and components of the cloud data center environment under consideration, including physical hosts, VMs (VMs), and the management software responsible for orchestrating resource allocation, as depicted in in Figure 1.

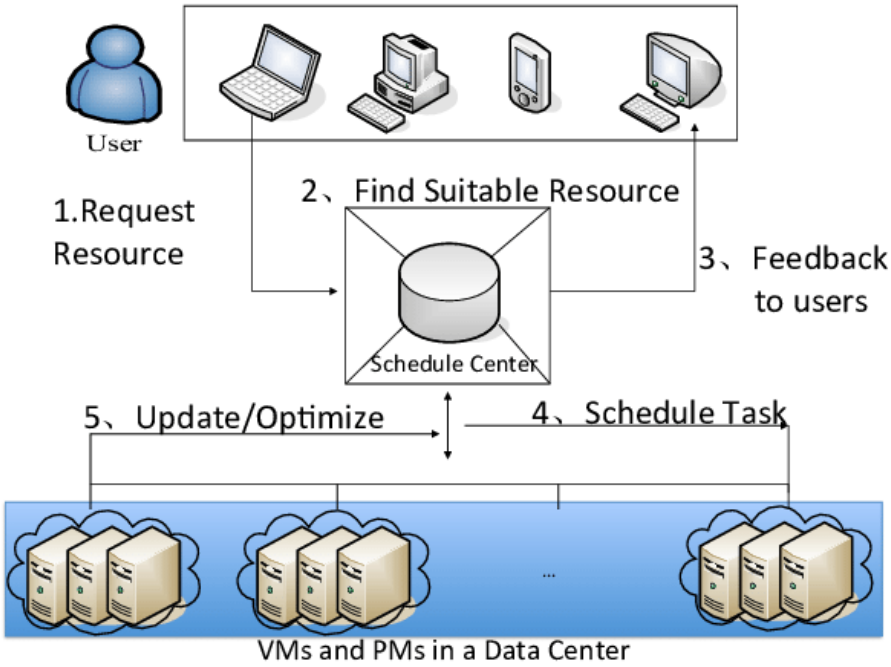

Figure 1. Cloud Computing Model Architecture.

3.1.1. Physical Hosts

The cloud data center comprises a set of physical hosts, which are the underlying hardware resources responsible for hosting VMs. Each physical host is equipped with computational resources such as CPU cores, memory, storage, and network bandwidth. These resources collectively form the basis for virtualization and consolidation activities within the data center.

3.1.2. VMs

VMs represent the virtualized instances of computational resources provisioned to users or applications within the cloud data center. Each VM encapsulates a subset of the physical hostfls resources, including CPU, memory, disk space, and network connectivity. VMs are instantiated, managed, and terminated dynamically based on workload demands and resource availability. On a single physical host, many virtual machines (VMs) can coexist, allowing for effective resource usage through consolidation.

 doi:10.20944/preprints202406.0500.v1

3.1.3. Management Software

The management software layer of the cloud data center system is responsible for coordinating the allocation and provisioning of resources to VMs. This software encompasses various components, including hypervisors, resource schedulers, and monitoring tools. Hypervisors, such as VMware vSphere or KVM (Kernel-based VM), facilitate the creation and management of VMs on physical hosts. Resource schedulers optimize resource allocation decisions based on factors such as workload characteristics, resource constraints, and service-level agreements (SLAs). Monitoring tools continuously collect data on resource utilization, performance metrics, and workload patterns, providing insights for decision-making processes related to VM consolidation and resource optimization.

3.1.4. Interaction and Communication

Communication and interaction among components within the cloud data center ecosystem are facilitated through various protocols and interfaces. Hypervisors communicate with physical hosts to manage VM lifecycle operations, such as creation, migration, and termination. Resource schedulers interact with hypervisors and monitoring tools to gather information on resource availability and workload demands, enabling dynamic resource allocation decisions. Additionally, communication channels exist between VMs for inter-VM communication within the same host or across different hosts in the data center network.

3.1.5. Scalability and Flexibility

The system model is designed to be scalable and flexible, capable of accommodating varying workloads and resource demands over time. New physical hosts can be added to the data center infrastructure to increase capacity, while VMs can be dynamically provisioned or decommissioned based on changing requirements. The management software layer is designed to adapt to fluctuations in workload demand, optimizing resource utilization and cost management objectives through techniques such as load balancing, VM consolidation, and auto-scaling.

Overall, the system model provides a comprehensive framework for understanding the architecture and operation of cloud data centers, laying the foundation for the development and evaluation of cost-effective resource management strategies, such as the hybrid algorithm for VM consolidation proposed in this paper.

3.2. Energy Consumption Model

3.2.1. Energy Consumption of Physical Hosts

Let E_{host} represent the energy consumption of a single physical host over a given time period. The energy consumption can be modeled as a function of the host's utilization, U_{host}, which is typically calculated as the ratio of the total utilized resources to the total available resources on the host. We can use a linear model to represent this relationship:

$$E_{\text{host}} = P_{\text{idle}} + (P_{\text{max}} - P_{\text{idle}}) \times U_{\text{host}}$$

where, P_{max} denotes the maximum power consumption of the host when fully utilized, and P_{idle} represents the idle power consumption of the host.

3.2.2. Energy Consumption of VMs

The energy consumption of a VM is contingent upon the resources it utilizes on the physical host. Let EVM denote the energy consumption of a single VM over the equivalent time period as the physical hosts. We can model this energy consumption as a function of the VMfls resource allocation, including CPU, memory, and disk usage. For simplicity, we can assume that the energy consumption of a VM is proportional to its resource utilization:

 doi:10.20944/preprints202406.0500.v1

$$E_{\text{VM}} = c_{\text{CPU}} \times U_{\text{CPU}} + c_{\text{mem}} \times U_{\text{mem}} + c_{\text{disk}} \times U_{\text{disk}}$$

where:

- U_{CPU}, U_{mem}, and U_{disk} are the CPU, memory, and disk utilization of the VM, respectively.
- c_{CPU}, c_{mem}, and c_{disk} are the coefficients representing the energy consumption per unit of resource utilization for CPU, memory, and disk, respectively.

3.2.3. Total Energy Consumption

The total energy consumption of the cloud data center system can be calculated by summing the energy consumption of all physical hosts and VMs:

$$E_{\text{total}} = \sum_{i=1}^{N_{\text{hosts}}} E_{\text{host}_i} + \sum_{j=1}^{M_{\text{VMs}}} E_{\text{VM}_j}$$

where:

- N_{hosts} is the total number of physical hosts in the data center.
- M_{VMs} is the total number of VMs in the data center.

3.3. Workload Modeling

Modeling workloads is essential for comprehending the resource requirements imposed on the cloud data center system. In this section, we present a detailed description of the workload model used in our study.

3.3.1. Types of Workloads

Workloads in a cloud data center can be categorized into various types based on their characteristics and resource requirements. Common workload types include:

- **CPU-bound Workloads**: These workloads primarily utilize CPU resources and involve intensive computational tasks such as data processing, mathematical computations, and simulations.
- **Memory-bound Workloads**: Workloads that predominantly require memory resources, such as in-memory databases, caching applications, and large-scale data analytics.
- **I/O-bound Workloads**: Workloads characterized by high I/O (input/output) operations, including database transactions, file processing, and network communication.
- **Mixed Workloads**: Workloads that exhibit a combination of CPU, memory, and I/O resource utilization, representing diverse application scenarios and usage patterns.

3.3.2. Workload Generation Model

We model the workload generation process using a stochastic approach, where the arrival of new workloads and their resource demands follow probabilistic distributions. Let W_i represent a workload instance, and t_i denote the arrival time of the i-th workload. The workload arrival process can be modeled as a Poisson process with rate parameter λ, representing the average arrival rate of workloads.

The resource demands of individual workloads are modeled using probability density functions (PDFs) or probability mass functions (PMFs) depending on the workload type. For instance, for a CPU-bound workload, the CPU resource demand D_{CPU} of workload W_i can be modeled using a normal distribution:

$$D_{\text{CPU}}^{(i)} \sim \mathcal{N}(\mu_{\text{CPU}}, \sigma_{\text{CPU}}^2)$$

where μ_{CPU} and σ_{CPU} are the mean and standard deviation of CPU resource demands, respectively.

 doi:10.20944/preprints202406.0500.v1

Similarly, for memory-bound and I/O-bound workloads, appropriate probability distributions (e.g., exponential, log-normal) can be used to model the resource demands of memory (D_{mem}) and I/O ($D_{\text{I/O}}$), respectively.

3.3.3. Total Workload Resource Demand

The total resource demand of the cloud data center system at any given time is the cumulative resource demand of all active workloads. Let $N_{\text{active}}(t)$ denote the number of active workloads at time t, and $D_{\text{total}}(t)$ represent the total resource demand vector at time t. Then, the total workload resource demand can be calculated as:

$$D_{\text{total}}(t) = \sum_{i=1}^{N_{\text{active}}(t)} D_i$$

where D_i is the resource demand vector of the i-th workload.

3.4. SLA and VM Placement Constraints

SLAs define the quality of service parameters guaranteed by the cloud provider to its customers. These SLAs often include requirements related to performance, availability, and resource allocation. In this subsection, we discuss the formulation of SLAs and VM placement constraints within the cloud data center environment.

3.4.1. SLA Formulation

Let S denote the set of SLAs specified by the customers of the cloud data center. Each SLA $s \in S$ can be represented as a tuple (T_s, U_s, R_s), where:

- T_s represents the maximum acceptable response time for requests associated with SLA s.
- U_s denotes the minimum acceptable level of resource utilization for the VMs allocated under SLA s.
- R_s specifies the reliability requirement, such as the minimum uptime or availability percentage.

These SLAs impose constraints on the resource allocation and placement of VMs within the cloud data center to ensure compliance with the specified service requirements.

3.4.2. VM Placement Constraint

To satisfy the SLAs and ensure optimal resource utilization, VM placement decisions must adhere to certain constraints. Let P denote the set of physical hosts in the data center, and M represent the set of VMs to be placed. The VM placement constraint can be formulated as follows:

$$\sum_{j \in P} x_{ij} = 1 \quad \forall i \in M$$

where:

- x_{ij} is a binary decision variable indicating whether VM i is placed on physical host j.

This constraint ensures that each VM is assigned to exactly one physical host, thereby meeting the resource allocation requirements specified by the SLAs.

3.5. Problem Formulation

In this section, we formally define the optimization problem aimed at minimizing costs through VM consolidation while satisfying SLAs and VM placement constraints within the cloud data center environment.

 doi:10.20944/preprints202406.0500.v1

3.5.1. Objective Function

The aim is to reduce the overall operational expenses of the cloud data center, encompassing energy usage and penalties incurred due to SLA breaches. Let C_{op} represent the total operational cost. The objective function can be formulated as follows:

$$\min_{x_{ij}} C_{\text{op}} = \sum_{i=1}^{M} \sum_{j=1}^{N} (E_{\text{host}_j} + E_{\text{VM}_i}) + \sum_{s \in S} \text{Penalty}_s$$

where:

- M is the total number of VMs.
- N is the total number of physical hosts.
- E_{host_j} represents host j's energy consumption.
- E_{VM_i} is the energy consumption of VM i.
- Penalty_s represents the penalty cost incurred for SLA violation of SLA s.

3.5.2. Constraints

The optimization problem is subject to the following constraints:

- Resource Allocation Constraint: The resource demand of each VM should not exceed the capacity of the physical host it is placed on. Let D_i denote the resource demand vector of VM i, and U_{host_j} represent the utilization level of physical host j. This constraint can be expressed as:

$$\sum_{i=1}^{M} D_i \leq U_{\text{host}_j} \quad \forall j$$

- VM Placement Constraint: Each VM must be assigned to exactly one physical host. This constraint ensures that VMs are placed optimally to utilize the available resources effectively. It can be represented as:

$$\sum_{j=1}^{N} x_{ij} = 1 \quad \forall i$$

  where x_{ij} is a binary decision variable indicating whether VM i is placed on physical host j.
- SLA Compliance Constraint:

  The resource allocation and VM placement decisions must adhere to the SLAs defined by the customers. Let U_s represent the minimum acceptable level of resource utilization specified in SLA s. This constraint ensures that the resource utilization meets the SLA requirements:

$$U_{\text{host}_j} \geq U_s \quad \forall j, \forall s \in S$$

4. The Proposed ModAFBA Algorithm

The modified Artificial Feeding Birds Algorithm (ModAFBA) is a nature-inspired metaheuristic optimization algorithm based on the feeding behavior of birds. It simulates the interactions between birds searching for food to find optimal solutions to optimization problems. In this work, we employ the AFBA technique to tackle the VM consolidation issue within cloud data centers, with the objective of reducing operational expenditures while adhering to SLAs and VM placement criteria. The proposed ModAFBA algorithm is presented in Algorithm 1 and described as below.

- Initialization: We initialize the population of artificial birds representing potential solutions to the optimization problem. Each bird corresponds to a potential solution, which consists of a placement configuration for the VMs on the physical hosts.

 doi:10.20944/preprints202406.0500.v1

- Feeding Behavior: The feeding behavior of birds is simulated using the concept of food source exploitation. In the context of the optimization problem, food sources represent potential solutions, and their quality is determined by the objective function value.
    1. Exploration Phase: During the exploration phase, birds search for new food sources by exploring the solution space. This is achieved through random perturbations or local search operators applied to the current solutions. In our problem, this corresponds to exploring neighboring VM placement configurations.
    2. Exploitation Phase: During the exploitation phase, birds exploit the best food sources found so far by intensifying the search around promising regions of the solution space. This involves refining and improving the quality of the solutions through local search or optimization techniques.
- Update Rules: The update rules in AFBA govern how birds adjust their behaviors based on the quality of the food sources encountered. In our problem, the update rules determine how birds adapt their VM placement configurations based on the objective function value and constraints.
    1. Movement: The movement of birds is guided by the quality of the food sources. Birds tend to move towards better food sources while avoiding poor-quality ones. This movement is represented by adjustments to the VM placement configurations.
    2. Selection: During the selection process, birds choose which food sources to exploit based on their quality. High-quality food sources are prioritized for exploitation, while low-quality ones may be abandoned or subject to further exploration.
- Termination Criteria: The optimization process continues iteratively until a termination criterion is met. Common termination criteria include reaching a maximum number of iterations, convergence of the objective function value, or a predefined computational budget.
- Mathematical Formulation: The application of AFBA to the VM consolidation problem involves translating the feeding behavior of birds into mathematical operations. Let X represent the population of bird solutions, $f(X)$ denote the objective function value, and $C(X)$ represent the constraints.

  The update rules in AFBA can be expressed as follows:

$$X(t+1) = X(t) + \Delta X(t)$$

  where $\Delta X(t)$ represents the adjustment to the VM placement configurations based on the exploration and exploitation phases.

  The termination criterion can be formulated as:

$$\text{Termination\_Criteria}(t) = \begin{cases} 1, & \text{if criterion is satisfied} \\ 0, & \text{otherwise} \end{cases}$$

  where t is the current iteration.

 doi:10.20944/preprints202406.0500.v1

Algorithm 1 The proposed ModAFBA for VM Consolidation

```
Initialize population of bird solutions X randomly
Evaluate objective function f(X) and constraints C(X) for each bird
while termination criterion not met do
    for each bird x_i in population X do
        Explore neighboring solutions to x_i to find Δx_i
        Update x_i with Δx_i
        Evaluate objective function f(x_i) and constraints C(x_i)
    end for
    Select best-performing birds for exploitation
    Exploit best-performing birds to refine solutions
end while
```

In the ModAFBA algorithm, there are distinct bird sizes, namely small and big, each with its set of movements, which include walk, fly, revise, and join. These movements enable birds to navigate and search for optimal solutions in different ways. Specifically:

- A bird has the ability to perform a local search by moving to a different location nearby, which is referred to as a "walk."
- For random exploration, a bird can execute the *fly* movement, wherein it lies and lands at a semi-random location.
- Another action, termed as "revise," entails the bird pausing and returning to the best location it remembers, allowing it to revise its course based on past experiences.
- The *join* movement, exclusive to big birds, allows them to join another bird by lying to its location and potentially copying its optimal solution.

Big birds have the capability to perform move 4, enabling them to join another bird, regardless of its size, through the *join* movement [27].

Two main inputs are required by the VM placement algorithm: (1) a list of virtual machines (VMs) that are marked for relocation, and (2) a pool of PMs that are willing to host the specified VMs. The hosting PM is indicated by each element of a vector that is the same length as the input virtual machines (VMs) that represents each bird. The activities of walk and fly are affected by the optimization problem, whereas the actions of revise and join function without it. Usually, the fly function yields random placements. But we've improved the random placement procedure in our updated strategy to lower the overall count of active PMs. In particular, the input parameter with a probability of prb directs the random distribution of the input VMs across a selection of randomly created PMs. As an alternative, a probability of 1 - prb is used to randomly assign the input VMs to the set of input PMs. By giving virtual machines (VMs) the opportunity to be located on a subset of PMs with a specific probability prb, as opposed to all PMs, this modification seeks to diversity the investigated options.

However, to prevent the possibility of overlooking superior solutions beyond the subset, the algorithm doesn't always utilize the subset of PMs. As a result, the algorithm uses the probability prb to randomly decide whether to use the subset. If the subset is used, each virtual machine (VM) is then placed on a host that is not part of the excluded subset after a subset of the available physical machines is randomly chosen to be excluded for placement. Every virtual machine is assigned at random to any host that is accessible if the subset is not used. The placement step, where nVMs is the number of virtual machines and m_{PMs} is the number of accessible physical machines, is the main cause of the fly algorithm's O($n_{VMs} \times m_{PMs}$) time complexity. Every bird in the population and every iteration of the AFBA algorithm is subjected to this algorithm.

Stochastic modifications to the VM location of a bird are introduced via the walk operation in the ModAFBA method. Its goal is to make little changes to the existing solution and see if that raises the fitness value. From the input bird, the algorithm chooses a host at random and then allocates each virtual machine (VM) to either the chosen host or another one that is available. Through the use of

 doi:10.20944/preprints202406.0500.v1

this method, the algorithm is able to investigate the immediate area around the existing answer and maybe pinpoint improvements. Like the fly method, the walk algorithm's time complexity may be traced mostly to the placement step, with an O($n_{VMs} \times m_{PMs}$)time complexity. This algorithm is used for every iteration of the AFBA algorithm and for a subset of the population of birds.

Fly, walk, revise, and join are the four primary phases of the ModAFBA algorithm. The temporal complexity of the fly and walk steps, for a single bird in a single loop, is O($n_{VMs} \times m_{PMs}$). Every bird in the population goes through the fly phase, while only a portion of the population goes through the walk step. Therefore, O($pop \times n_{VMs} \times m_{PMs} \times itr$) and O($b_{selected} \times n_{VMs} \times m_{PMs} \times itr$)is the worst-case time complexity of the fly and walk steps, respectively. In this case, pop stands for population size, $b_{selected}$ for number of selected birds, n_{VMs} for number of virtual machines, m_{PMs} for number of physical machines, and itr for maximum number of iterations.

In the rewrite step, the bird copies values from memory to obtain the best spot it has identified so far. This is a straightforward value assignment that takes O(n_{VMs}) of time for a single bird in a single iteration. The revision process is applied to a portion of the population's birds. As a result, O($b_{revised} \times n_{VMs} \times itr$)is the worst-case time complexity of the revision step, where $b_{revised}$ is the number of revised birds.

In the join phase, a large bird can fly to the location of another bird, no matter how big or small, and mimic its positioning. This is a straightforward value assignment that takes O(n_{VMs})of time for a single bird in a single iteration. A subset of the population's large birds are used for the join step. Consequently, O($b_{joined} \times n_{VMs} \times itr$), is the worst-case time complexity of the join step, where b_{joined} is the number of birds joined.

Hence, the upper bound of the time complexity of the modified AFBA can be represented as depicted in Equation 1.

$$\begin{aligned} Complexity = O(&pop \times n_{VMs} \times m_{PMs} \times itr \\ &+ b_{selected} \times n_{VMs} \times m_{PMs} \times itr \\ &+ b_{revised} \times n_{VMs} \times itr \\ &+ b_{joined} \times n_{VMs} \times itr) \end{aligned} \tag{1}$$

5. Experimental Methodology

In this section, we describe the experimental methodology utilized to assess the effectiveness of the proposed ModAFBA algorithm for VM consolidation in cloud data centers. The experimental setup encompasses the configuration of the simulated cloud data center environment, including the specifications of PMs and VMs, as well as the deployment of benchmark algorithms for comparative analysis. Furthermore, the process of selecting and defining performance metrics, which include energy consumption, SLA adherence, and VM migration count, is elaborated to evaluate the efficacy of the ModAFBA approach. Furthermore, other experimental configurations, such as the duration of experiments and the number of repetitions, are specified to ensure robustness and reliability in the evaluation process.

5.1. Experimental Setup

In this study, we conducted simulations to evaluate the performance of our proposed hybrid system in comparison to state-of-the-art methods within the field. These simulations were performed using several benchmark datasets. The identified methods for comparison included the studies in TOPSIS [28], PVMP [22], and SVMP [29].

While all reference methods utilize the MMT policy for VM selection, their approaches vary regarding VM placement and underload detection algorithms. Notably, PVMP utilizes a straightforward technique for identifying underloaded PMs, while the TOPSIS method implements the TACND approach, and SVMP relies on stochastic process-based methodologies for underloaded PM detection. To establish a robust data usage model, the initial training phase spans 35 time steps, during which

 doi:10.20944/preprints202406.0500.v1

optimization is halted. Following this initial training phase, the consolidation process begins in the experiments. The ModAFBA method is then employed with three distinct QoS targets ($\theta = 0.15, 0.25, 0.35$) to assess its impact on performance metrics.

For the sake of experiment reproducibility, we employed the CloudSim toolkit [30] extension to establish a uniform experimental environment. The performance assessment of our proposed system occurs within a data center comprising 800 heterogeneous servers. These servers are classified into two configurations, as illustrated in Table 2.

Table 2. Server Configurations.

Server Type	Number of Servers	Specifications
HP ProLiant ML110 G5	400	2 cores at 2660 MHz
HP ProLiant ML110 G4	400	2 cores at 1860 MHz

The workload data utilized in our experiments was obtained from a dataset exhibiting Markov properties, as provided in [22], which forms a component of the CoMon project [31]. To ensure compatibility with the CloudSim toolkit, we processed the dataset beforehand. Out of the pool containing 100 various sets of VMs available in the dataset, we randomly chose 10 unique sets for our experimentation.

5.2. Metrics

Performance metrics are essential tools for evaluating the effectiveness and efficiency of VM placement algorithms in cloud data centers. These metrics provide quantitative measures of various aspects such as resource utilization, energy consumption, SLA compliance, migration overhead, and scalability [15,29]. Resource utilization metrics assess the effective allocation of physical machine resources to VMs, while energy consumption metrics quantify the total energy consumed by the data center infrastructure. SLA compliance metrics measure the extent to which VMs meet their specified SLAs, while migration overhead metrics assess the cost of VM migration. Scalability metrics evaluate the ability of VM placement algorithms to handle larger data centers and workloads. Together, these metrics enable stakeholders to assess and compare the performance of VM placement algorithms, identify areas for improvement, and make informed decisions to optimize cloud data center operations [1].

The metrics used in this work can be defined as below.

1. SLAV_TPM (SLA violation Time Per active PM) is utilized as a metric to assess the degree of SLA violation, adhering to the definition of overload time fraction (OTFr). It is calculated by dividing the total time that the value of OTFr on a specific PM surpasses the agreed-upon threshold by the total active time of that PM. The formula for SLAV_TPM is represented according to Eq. 2.

$$\text{SLA\_TPM} = \frac{\sum_{j=1}^{m_{PMs}} T_j}{T_{OTFr_j}} \tag{2}$$

   where m_{PMs} denotes the count of PMs, T_j is the cumulative time the OTFr value on PM j exceeds the agreed threshold, and T_{OTFr_j} represents the total active time of PM j.
2. SLAV_EC (Energy Constrained SLA Violation) serves as a comprehensive metric to evaluate both energy efficiency and adherence to SLA requirements. It combines the SLAV_TPM metric with energy usage, represented as per following Eq. 3. This metric provides a holistic view of system performance, considering both SLA compliance and energy usage simultaneously.

$$\text{SLAV\_EC} = \text{SLAV\_TPM} \times \text{E\_Usage} \tag{3}$$

 doi:10.20944/preprints202406.0500.v1

3. SLAV_EC_M (Energy-Constrained SLA Violation and Migration) serves to simultaneously minimize SLA violations, energy usage, and VM migration count (MC). It is represented as per Eq. 4.

$$\text{SLAV\_EC\_M} = \text{SLAV\_EC} \times \text{MC} \tag{4}$$

6. Results and Analysis

Experimental results were obtained by conducting each set of experiments 10 times, and the averaged outcomes are detailed in this section, adhering to the experimental setup delineated in Section 5.1. The performance assessment of the proposed ModAFBA method was executed under three scenarios, denoted as ModAFBA 1, ModAFBA 2, and ModAFBA 3, corresponding to $\theta = 0.15, 0.25$, and 0.35, respectively. These were juxtaposed against established methods including TOPSIS, SVMP, and PVMP, in accordance with the identified performance metrics described in Section 5.2.

As illustrated in Figure 2, the ModAFBA 3 method has exhibited an enhancement in energy usage by 96.69%, 126.05%, and 186.79% compared to TOPSIS, SVMP, and PVMP methods, respectively. This indicates that the proposed ModAFBA 3 surpasses competing approaches in minimizing energy consumption. By permitting specific VM placements that other methods might consider overloads, the QoS-aware VM placement technique proposed in this research can optimize VM allocations more effectively in relation to energy consumption. Moreover, the ModAFBA 3 method contributes to the enhancement in energy usage by reducing the number of active PMs.

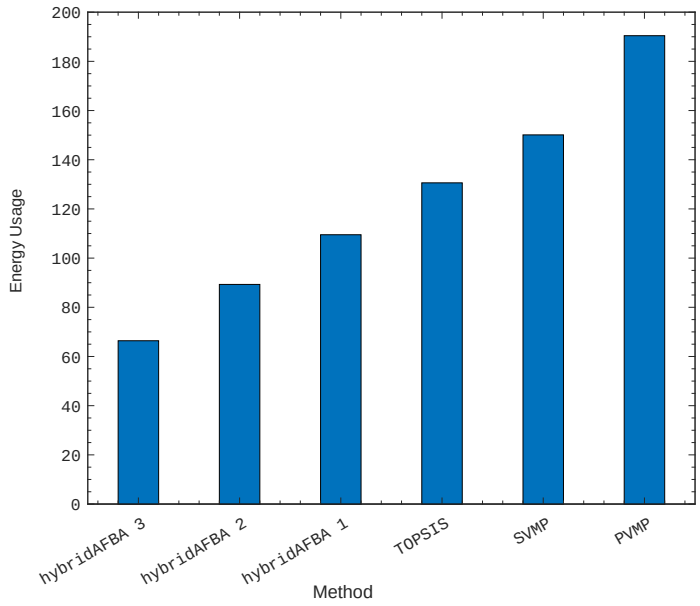

Figure 2. Comparison of energy usage by ModAFBA method against TOPSIS, SVMP, and PVMP methods.

The SLA_TPM metric was employed to assess SLA violations within the system. As illustrated in Figure 3, the SLA_TPM value decreases with the implementation of the proposed ModAFBA policy as the desired level of QoS is lowered, yielding values below those of benchmark approaches when $\theta = 0.35$. This is likely attributed to the decreased occurrence of SLA violations over an extended duration after a PM becomes overloaded.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 7 June 2024 doi:10.20944/preprints202406.0500.v1

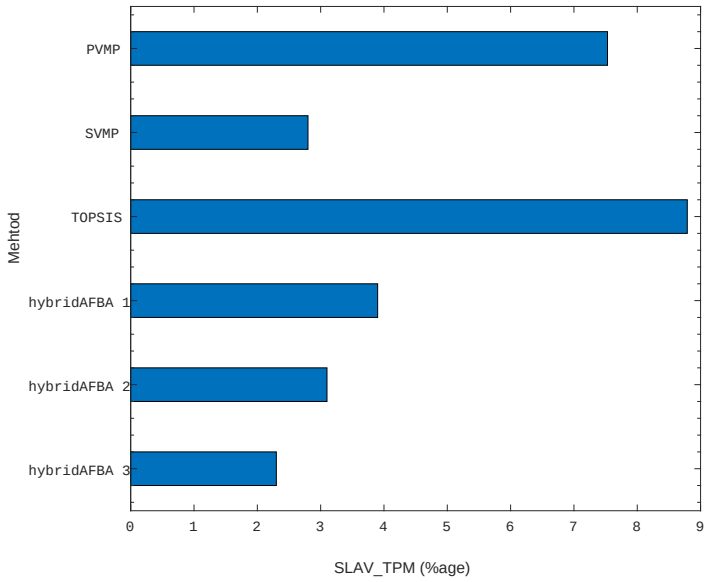

Figure 3. Comparison of SLAV_TPM by ModAFBA method against TOPSIS, SVMP, and PVMP methods.

As shown in the simulation results depicted in Figure 4, the ModAFBA approach exhibited a decrease in the number of VM migrations compared to the performance evaluation methods, including TOPSIS, SVMP, and PVMP. The reduction was particularly pronounced, approximately 517.24%, 29.29%, and 882.08% compared to TOPSIS, SVMP, and PVMP methods, respectively, when the θ value was set to 0.35.

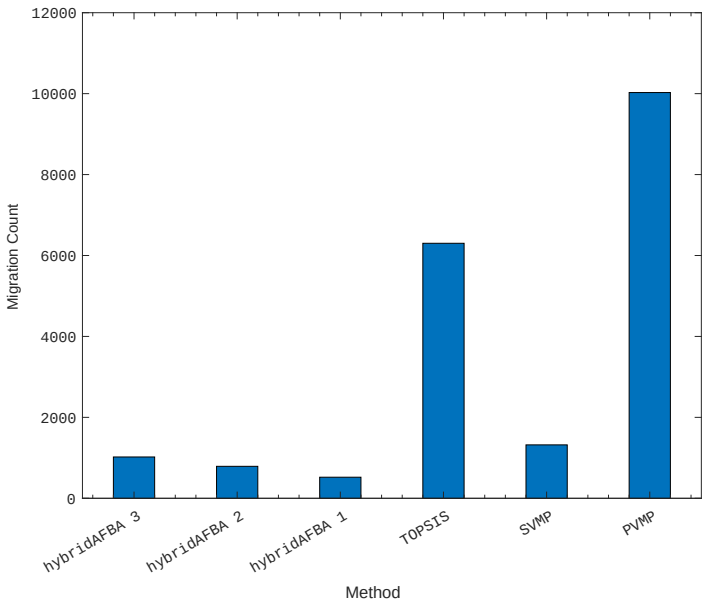

Figure 4. Comparison of migration count by ModAFBA method against TOPSIS, SVMP, and PVMP methods.

In this conclusion, it is stated that a thorough comparison was made between the proposed ModAFBA method and other identified methods using two metrics: SLAV_EC and SLAV_EC_M. The conclusion affirms that the developed system outperformed the identified methods across all values of the QoS objective, as evidenced by the results depicted in Figures 5 and 6.

Further elaboration is provided based on the observation from Figure 5, where it is highlighted that the ModAFBA method achieved significant improvements over TOPSIS, SVMP, and PVMP

 doi:10.20944/preprints202406.0500.v1

methods by 242.78%, 72.13%, and 182.04%, respectively, in terms of the SLAV_EC metric. Similarly, the conclusion emphasizes the improvements demonstrated by the ModAFBA method over the same benchmark methods by 221.03%, 64.47%, and 167.81%, respectively, in terms of the SLAV_EC_M metric, as depicted in Figure 6.

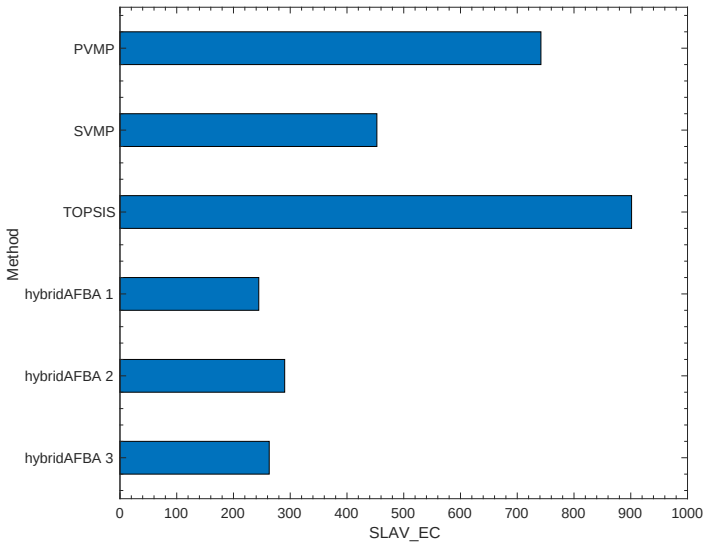

Figure 5. Comparison of SLA_EC by ModAFBA method against TOPSIS, SVMP, and PVMP methods.

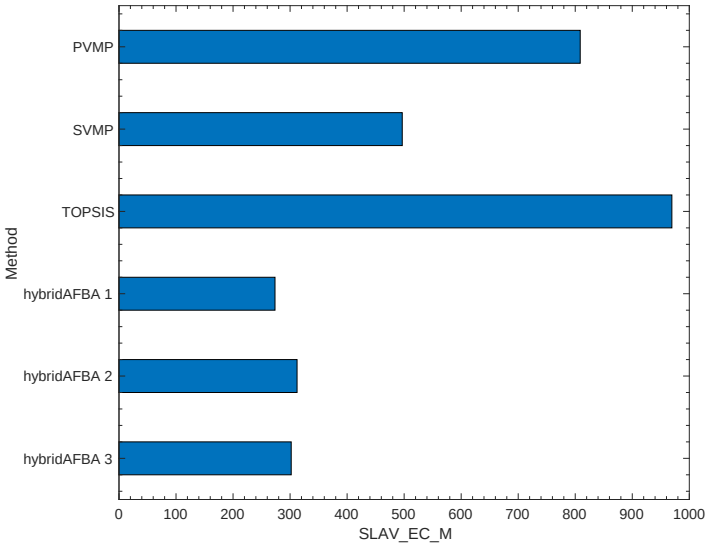

Figure 6. Comparison of SLA_EC_M by ModAFBA method against TOPSIS, SVMP, and PVMP methods.

7. Conclusion

In summary, this paper presented the Modified Artificial Feeding Birds Algorithm (ModAFBA) as a novel approach for VM consolidation. The ModAFBA algorithm was designed to optimize cost management by minimizing energy consumption while meeting SLAs and maintaining high system performance. Through comprehensive experimental evaluation, the effectiveness of the ModAFBA approach was demonstrated, showcasing superior performance compared to existing VM consolidation techniques. The experimental findings demonstrated substantial improvements in key performance metrics. Specifically, the ModAFBA method exhibited significant enhancements in energy usage, SLA compliance, and the number of VM migrations compared to benchmark algorithms such as TOPSIS, SVMP, and PVMP methods. Notably, the ModAFBA 3 method achieved a reduction in energy usage by

 doi:10.20944/preprints202406.0500.v1

49.16%, 55.76%, and 65.13% compared to TOPSIS, SVMP, and PVMP methods, respectively. Moreover, the ModAFBA method resulted in decreases of around 83.80%, 22.65%, and 89.82% in the quantity of VM migrations in contrast to the aforementioned benchmark techniques.

These results underscore the effectiveness of the ModAFBA algorithm in enhancing VM consolidation processes and optimizing resource utilization within cloud data centers. Through the reduction of mitigation of SLA violations, energy consumption, and optimization of VM placements, the ModAFBA approach presents a scalable and efficient solution to the challenges encountered by cloud service providers. In future research endeavors, further improvements and extensions to the ModAFBA method can be explored. One potential avenue is the incorporation of machine learning techniques to enhance adaptability and scalability. Additionally, conducting experiments in real-world cloud data center environments would provide valuable insights into the practical applicability of the ModAFBA approach and its performance under diverse operational conditions.

References

1. Radi, M.; Alwan, A.A.; Gulzar, Y. Genetic-Based Virtual Machines Consolidation Strategy With Efficient Energy Consumption in Cloud Environment. *IEEE Access* **2023**.
2. Monshizadeh Naeen, M.A.; Ghaffari, H.R.; Monshizadeh Naeen, H. Cloud data center cost management using virtual machine consolidation with an improved artificial feeding birds algorithm. *Computing* **2024**, pp. 1–29.
3. Khan, T.; Tian, W.; Ilager, S.; Buyya, R. Workload forecasting and energy state estimation in cloud data centres: ML-centric approach. *Future Generation Computer Systems* **2022**, *128*, 320–332.
4. Dogani, J.; Khunjush, F.; Seydali, M. Host load prediction in cloud computing with discrete wavelet transformation (dwt) and bidirectional gated recurrent unit (bigru) network. *Computer Communications* **2023**, *198*, 157–174.
5. Avula, R.N.; Zou, C. Performance evaluation of TPC-C benchmark on various cloud providers. 2020 11th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON). IEEE, 2020, pp. 0226–0233.
6. Javadi, S.A.; Suresh, A.; Wajahat, M.; Gandhi, A. Scavenger: A black-box batch workload resource manager for improving utilization in cloud environments. Proceedings of the ACM symposium on cloud computing, 2019, pp. 272–285.
7. Saif, F.A.; Latip, R.; Derahman, M.; Alwan, A.A. Hybrid meta-heuristic genetic algorithm: Differential evolution algorithms for scientific workflow scheduling in heterogeneous cloud environment. Proceedings of the future technologies conference. Springer International Publishing Cham, 2022, pp. 16–43.
8. Radi, M.; Alwan, A.; Abualkishik, A.; Marks, A.; Gulzar, Y. Efficient and Cost-effective Service Broker Policy Based on User Priority in VIKOR for Cloud Computing. *Sci. J. King Faisal Univ. Basic Appl. Sci* **2021**, *22*, 1–8.
9. Saif, F.A.; Derahman, M.; Alwan, A.A.; Latip, R. Performance Evaluation of Task Scheduling using Hybrid Meta-heuristic in Heterogeneous Cloud Environment. *International Journal of Advanced Trends in Computer Science and Engineering* **2019**, *8*, 3249–3257.
10. Alshammari, M.M.; Alwan, A.A.; Nordin, A.; Abualkishik, A.Z. Data backup and recovery with a minimum replica plan in a multi-cloud environment. *International Journal of Grid and High Performance Computing (IJGHPC)* **2020**, *12*, 102–120.
11. Khan, M.A. An efficient energy-aware approach for dynamic VM consolidation on cloud platforms. *Cluster Computing* **2021**, *24*, 3293–3310.
12. Dabhi, D.; Thakor, D. Hybrid VM allocation and placement policy for VM consolidation process in cloud data centres. *International Journal of Grid and Utility Computing* **2022**, *13*, 459–471.
13. Dabhi, D.; Thakor, D. Utilisation-aware VM placement policy for workload consolidation in cloud data centres. *International Journal of Communication Networks and Distributed Systems* **2022**, *28*, 704–726.
14. Omer, S.; Azizi, S.; Shojafar, M.; Tafazolli, R. A priority, power and traffic-aware virtual machine placement of IoT applications in cloud data centers. *Journal of systems architecture* **2021**, *115*, 101996.

 doi:10.20944/preprints202406.0500.v1

15. Fatima, A.; Javaid, N.; Anjum Butt, A.; Sultana, T.; Hussain, W.; Bilal, M.; Hashmi, M.A.u.R.; Akbar, M.; Ilahi, M. An enhanced multi-objective gray wolf optimization for virtual machine placement in cloud data centers. *Electronics* **2019**, *8*, 218.
16. Zhou, Z.; Shojafar, M.; Li, R.; Tafazolli, R. EVCT: An efficient VM deployment algorithm for a software-defined data center in a connected and autonomous vehicle environment. *IEEE Transactions on Green Communications and Networking* **2022**, *6*, 1532–1542.
17. Li, H.; Zhu, G.; Zhao, Y.; Dai, Y.; Tian, W. Energy-efficient and QoS-aware model based resource consolidation in cloud data centers. *Cluster Computing* **2017**, *20*, 2793–2803.
18. Guo, Y.; Stolyar, A.L.; Walid, A. Online VM auto-scaling algorithms for application hosting in a cloud. *IEEE Transactions on Cloud Computing* **2018**, *8*, 889–898.
19. Riahi, M.; Krichen, S. A multi-objective decision support framework for virtual machine placement in cloud data centers: a real case study. *The Journal of Supercomputing* **2018**, *74*, 2984–3015.
20. Karmakar, K.; Banerjee, S.; Das, R.K.; Khatua, S. Utilization aware and network I/O intensive virtual machine placement policies for cloud data center. *Journal of Network and Computer Applications* **2022**, *205*, 103442.
21. Zhang, L.; Zhuang, Y.; Zhu, W. Constraint programming based virtual cloud resources allocation model. *International Journal of Hybrid Information Technology* **2013**, *6*, 333–344.
22. Beloglazov, A.; Buyya, R. Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. *Concurrency and Computation: Practice and Experience* **2012**, *24*, 1397–1420.
23. Wu, Q.; Ishikawa, F.; Zhu, Q.; Xia, Y. Energy and migration cost-aware dynamic virtual machine consolidation in heterogeneous cloud datacenters. *IEEE transactions on Services Computing* **2016**, *12*, 550–563.
24. Wu, G.; Tang, M.; Tian, Y.C.; Li, W. Energy-efficient virtual machine placement in data centers by genetic algorithm. Neural Information Processing: 19th International Conference, ICONIP 2012, Doha, Qatar, November 12-15, 2012, Proceedings, Part III 19. Springer, 2012, pp. 315–323.
25. Sonklin, C.; Tang, M.; Tian, Y.C. A decrease-and-conquer genetic algorithm for energy efficient virtual machine placement in data centers. 2017 IEEE 15th International Conference on Industrial Informatics (INDIN). IEEE, 2017, pp. 135–140.
26. Ye, X.; Yin, Y.; Lan, L. Energy-efficient many-objective virtual machine placement optimization in a cloud computing environment. *IEEE access* **2017**, *5*, 16006–16020.
27. Abadi, R.M.B.; Rahmani, A.M.; Alizadeh, S.H. Challenges of server consolidation in virtualized data centers and open research issues: a systematic literature review. *The Journal of Supercomputing* **2020**, *76*, 2876–2927.
28. Arianyan, E.; Taheri, H.; Sharifian, S. Novel energy and SLA efficient resource management heuristics for consolidation of virtual machines in cloud data centers. *Computers & Electrical Engineering* **2015**, *47*, 222–240.
29. Monshizadeh Naeen, H.; Zeinali, E.; Toroghi Haghighat, A. A stochastic process-based server consolidation approach for dynamic workloads in cloud data centers. *The Journal of Supercomputing* **2020**, *76*, 1903–1930.
30. Calheiros, R.N.; Ranjan, R.; Beloglazov, A.; De Rose, C.A.; Buyya, R. CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and experience* **2011**, *41*, 23–50.
31. Park, K.; Pai, V.S. CoMon: a mostly-scalable monitoring system for PlanetLab. *ACM SIGOPS Operating Systems Review* **2006**, *40*, 65–74.