*applied sciences*

*Article*

# Multi-Objective Hybrid Flower Pollination Resource Consolidation Scheme for Large Cloud Data Centres

**Mohammed Joda Usman [1], Lubna A. Gabralla [2], Ahmed Aliyu [1], Danlami Gabi [3] and Haruna Chiroma [4,*]**

[1] Department of Mathematical Sciences, Bauchi State University Gadau, Bauchi P.O. Box 65, Nigeria
[2] Department of Computer Science and Information Technology, College of Applied, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia
[3] Department of Computer Science, Kebbi State University of Science and Technology, Aliero P.M.B 1144, Nigeria
[4] College of Computer Science and Engineering, University of Hafr Al-Batin, Hafar Al Batin 39524, Saudi Arabia
* Correspondence: chiromaharun@fcetgombe.edu.ng

**Abstract:** Cloud Computing has rapidly emerged as a successful paradigm for providing Information and Communication Technology (ICT) infrastructure. Resource allocation is used to execute user applications in the form of requests for consolidated resources in order to minimize energy consumption and violation of the Service Level Agreement (SLA) for large-scale data centers resource utilization. The energy consumption is usually caused due to local entrapment and violation of SLA during resource assigning and execution. Several researchers have proposed solutions to reduce local entrapments and violations of SLA, to minimize the energy consumption of the entire data center. However, strategies employed in their solutions face entrapment in either local searches or at the global search level with a certain level of SLA violation. In this light, a Multi-Objective Hybrid Flower Pollination Resource Consolidation (MOH-FPRC) scheme for efficient and optimal resource consolidation of data center resources is put forward. The Local Neighborhood Search (LNS) algorithm has been employed for addressing entrapment at the local search level, while the prominent flower pollination algorithm is used to solve the problem of entrapment at the global search level. This, in turn, reduces the energy consumption of the data centers. In addition, clustering strategies have been introduced with a robust migration mechanism to minimize the violation of SLA while also satisfying minimum energy consumption. The simulation results using the MultiRecCloudSim simulator have shown that our proposed MOH-FPRC demonstrates an improved performance on the data center energy consumption, resource utilization, and SLA violation with a 20.5% decrease, 23.9% increase, and 13.5% reduction, respectively, as compared with the benchmarked algorithms. The proposed scheme has proven its efficiency in minimizing energy consumption while at the same time improving the data center resource allocation with minimum SLA violations.

**Keywords:** cloud computing; data centers; energy-efficiency; flower pollination algorithm; consolidation

check for updates

**Citation:** Usman, M.J.; Gabralla, L.A.; Aliyu, A.; Gabi, D.; Chiroma, H. Multi-Objective Hybrid Flower Pollination Resource Consolidation Scheme for Large Cloud Data Centres. *Appl. Sci.* **2022**, *12*, 8516. https://doi.org/10.3390/app12178516

Academic Editors: Eui-Nam Huh and Agostino Forestiero

Received: 29 March 2022
Accepted: 7 July 2022
Published: 25 August 2022



## 1. Introduction

The delivery of services and resources on the Internet, which is packaged as Virtual Machine (VM), is called Cloud Computing (CC). The cloud makes facilities available for use by industries, companies, and organizations on a payment basis, depending on the magnitude of the utilization. This saves much of the cost for enterprises as it allows them to concentrate on the core mandate without building the cloud infrastructure on the organizations' premises. The use of CC as a platform for providing services has increased tremendously and been successfully utilized in recent years. The services provided by the CC platform are in different categories of cloud data center levels, such as Software as a service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS) [1,2].

Generally, Cloud providers offer a heterogeneous set of resources to users with different performances and capacities. Hence, resource management must be formulated as an optimization problem that intends to optimize the allocation and utilization of cloud data center resources for users with different levels of capacity and usage. However, most of the existing optimization algorithms suffer from local optima entrapment and non-optimal resource consolidation, which makes them inefficient in managing data center resources [3,4]. Local optima are defined as the relatively best solutions within a neighbor solution set that is not necessarily optimum. Therefore, local optima entrapment could result to slow convergence and non-optimal resource consolidation. The energy consumption of underutilized Physical Machines (PM) in cloud data centers accounts for a substantial amount of the actual energy used [5,6]. This can be attributed to the entrapment at the local optima resulting in an inefficient resource allocation policy of the data center [7,8].

Simultaneously maintaining energy efficiency and the user's Service Level Agreement (SLA) is sacrosanct for the service providers [9,10]. The cloud service providers require energy-saving strategies such as the VMs consolidation [11]. According to [12,13], one of the main reasons for the resource inefficiency in current data centers is overprovisioning. It leads to inefficient utilization of resources hence, causing poor management of resources that depletes more energy and violation of SLA. For these reasons, energy efficiency and SLA compliance are becoming increasingly important for cloud data centers. The underutilization of Physical Machines (PMs) has also been attributed to the inefficient resource allocation policy of the data center [14–16]. However, central processing units (CPUs) are the most used resources, which fully depends on the PM configuration and the data center set up. CPUs are considered the single largest energy consumer within the Cloud data center. Therefore, it is often assumed that inefficient CPU utilization leads to an underutilized PM of the data center. However, all components are to meet the expected results in reality. Although the CPU is apparently underutilized, other components are already working at their limit. The data center's maintenance is expensive and endangers both humans and the natural environment by producing unwanted substances such as carbon monoxide [17–19]. However, resource management in large-scale data centers is a complex and challenging issue. The demand for large servers, storage, and network infrastructure to accommodate faster processing contributes to the energy consumption at the Cloud data center. Several multi-objective and optimization algorithms for efficient resource allocation have been suggested for minimizing energy consumption and violation of SLA at the data centers.

The existing nature-inspired resource allocation algorithm in Cloud may not fully curtail the problem of energy consumption, thus, creating the need for an ideal solution. Recently, Flower Pollination Algorithm (FPA) has been applied in CC data centers [20–22] and other areas; the results obtained are competitive and, in some situations, perform better than other state-of-the-art nature-inspired algorithms. FPA mimics the idea of flowers' reproduction methods as discussed in our first technique [20]. However, the entrapment issue in local search and *VM* migration across the data center by the resource consolidation algorithms are also the causes of the data center's resource management inefficiency. Local Neighborhood Search (LNS) and Clustering strategies are introduced with a robust migration mechanism to minimize violation of SLA while also satisfying minimum energy consumption to address energy-efficient resource consolidation issues in IaaS Clouds.

- The main contributions of this article are as follows:
- Some mathematical models for the objective functions of energy-efficiency and SLA violation are derived.
- Incorporation of LNS into FPA for addressing entrapment at both the local and global search levels.
- Integration of clustering strategies with robust migration mechanisms into the FPA-LNS to minimize the violation of SLA while satisfying minimum energy consumption.

The remaining part of this article is organized as follow: Section 2 presents the motivation of the study. Section 3 contains a review of the existing literature. Section 4 provides the Multi-Objective Hybrid Flower Pollination Resource Consolidation (MOH-FPRC) scheme and its mathematical modelling. Section 5 presents the performance evaluation and experimental results analysis. Finally, Section 6 concludes the research findings.

## 2. Motivation of the Study

Data center resource consolidation increases IaaS complexity due to resource heterogeneity, the migration of VMs applications, and workloads, which are typically moved to another PM while preserving their status and network connection. Further, Cloud service providers offer on-demand services that are regularly deployed or re-deployed, which triggers frequent allocation/deallocation of VMs. The transfer of workload from one PM to another and frequent allocation/deallocation of VMs contributes to the high energy depletion of the data center. These mentioned challenges require optimization algorithms that can handle both local and global searches for an optimal solution with zero or minimum entrapment. This, in turn, will minimize the energy consumption of the data centers. In most of the existing studies, their solution faces entrapment at either the local search or at the global search with a certain level of SLA violation.

In addition, most of the current studies often neglected the essential characteristics of CC such as heterogeneity, dynamism, and the stake of various Computing resources (CPU, Memory, Storage), thereby failing to fulfill the Cloud service provider's objective. These resources are bound to high-energy consumption due to performance variations such as resource utilization and *VM* migration, which leads to SLA violation.

Therefore, there is a need to design an algorithm that is based on the Pareto optimization strategy for achieving multiple objectives of both energy and violation of SLA minimization, which are also conflicting. In addition, the algorithm must be able to minimize entrapment at both the local and global search of space, while also satisfying SLA. The following section provides the related works of this study.

## 3. Literature Review

Several researchers have explored the use of Natured-Inspired optimization approaches to solve data center resource consolidation issues. These have paved the way for devising a better approach to solving energy consumption in the Cloud data center. Researchers that have contributed immensely to the use of nature-inspired optimization approaches are explored.

Ref. [23] applied the reassignment algorithm (GeNePi) to reduce the consumption of energy and maximize the use of resources. The GeNePi is found to improve energy efficiency and maximize the utilization of resources. The green cloud data center has been presented [24] using the ACO system. The approach uses dynamic *VM* migration to move the under-loaded PMs to the averagely loaded PMs so that some PMs will switch off, stand by, and/or hibernate while reducing energy consumption and increasing data center performance. The approach uses two regression models to predict the CPU utilization of the PMs to reduce SLA violation and energy usage. This has led to the achievement of near-optimal solutions based on the specified objective function. However, the migration has been done considering CPU utilization while other components are still working, which also consumes energy. The approach produces sub-optimal results in terms of resource utilization and energy consumption. The high demand by users to access resources and process data has contributed to the inefficiency of the algorithms used, which led to higher consumption of energy in the data centers.

There are other research works that used the ACO system to propose solutions for data center consolidation [25,26]. In this case, the nature-inspired algorithm runs around a local solution continuously without any progress, otherwise reaching the iteration limits. Hence, if the value remains constant for a series of runs, then the algorithm traps in a local minimum after some generation. Consolidation led to inefficient resource utilization and energy wastage. In other words, trapping is a serious problem of nature-inspired algorithms in resource management at the cloud data center. Furthermore, the proposed approaches did not take into consideration of the SLA violation due to the migration of VMs. Thus, addressing resource allocation optimization problems using a consolidation technique that avoids local optima entrapment is still an active research area. In addition, a cost-centric ant colony optimization that focuses on resource allocation over cloud infrastructure is proposed for minimizing network cost and execution time [27]. Therefore, the network cost and execution time are considered as the fitness function for achieving improved performance. However, the SLA violation constraint has not been considered.

Another approach, which is based on a gravitational search algorithm improved using fuzzy for resource allocation in CC, has been suggested to handle nonlinear problems due to the exponential time for checking search space [28]. Masses have been generated by incorporating a sequence of tasks assigned to all machines. Thus, each mass position is considered a solution to the problem. Fuzzy logic is employed to determine the number of masses that affect each other during the implementation of the gravitational search algorithm. A resource allocation strategy that employed a hybridized Cuckoo Search Optimization (CSO) and Shuffled Frog Leaping Algorithm (SFLA) has been proposed [29]. The CSO is used to initialize, create, and evaluate fitness function execution. SFLA is used to achieve quick convergence and simpler implementation to achieve lower execution times during task allocation. However, service level agreement has not been considered in achieving less computation time.

A resource allocation concept, which is self-adaptive, considering cloud-based software as a service using the iterative quality of service (QoS) Prediction model has been proposed to minimize high overhead due to frequent machine migration [30]. The model entails feedback loops which go through the developed iterative QoS prediction framework and PSO-centered runtime decision algorithm. However, this model is centered on software as a service paradigm of the CC, without considering the infrastructure as a service aspect. Further, a solution for dynamic resource allocation based on improved power management and optimized task scheduling has been presented for to address the problem of high energy consumption due to inefficient resource scheduling [31]. The solution is based on a dynamic resource table maintenance algorithm and prediction strategy for improved response and task completion time. However, the service level agreement for power management in CC may not have been considered. Ref. [24] proposes ant colony-based virtual machine consolidation (ACS-VMC) for green CC to improve energy efficiency. Similarly, [32] applied a multi-objective CSO (MO-CSOA) to minimize PMs to improve energy efficiency. Ref. [32] used VM Consolidation in cloud data centers based on ACO (VMC-ACO) for the reduction of energy consumption in the cloud data centers. Ref. [33] modified PSO (MPSO) for VM reallocation to improve the energy efficacy of the system. Ref. [34] proposed a prediction based on host selection for energy efficiency for VM. Ref. [35] applied Reinforcement Learning to improve the efficiency of energy utilization in the cloud data center [36].

A Whale optimization algorithm has been employed for handling poor resource usage and high operational cost of the cloud system in the process of task scheduling [37]. Further, a multi-objective optimization model is used to improve the computing resources and performance of the cloud system. Similarly, a quality service-focused cloud scheduler that uses deep reinforcement learning has been proposed for handling large task scheduling [38]. The deep learning approach is used for managing decision-making problems in terms of task allocation to the virtual machine. Further, Ref. [39] suggested a deep adversarial imitation reinforcement learning architecture. The framework is used for scheduling time-constrained cloud tasks. In this, user requests are scheduled in such a way as to maximize task success rate while also reducing task response time. A deep Q-learning strategy for task scheduling in cloud computing setup has been proposed to improve the quality of service [40]. The Q-learning approach is introduced for the purpose of addressing the problem of directed acyclic graph jobs in cloud data centers [41]. An Energy-Efficient Dynamic Resource Management has been proposed to address the issue of energy consumption using a dynamic *VM* migration strategy and k-means clustering technique. However, the technique focuses on quality of service and make-span to reduce the DC energy consumption and resource under-utilization. Similarly, an energy-efficient virtual machine migration approach with SLA conservation in cloud computing has been proposed by [42]. The technique focuses on SLA violations based on the number of *VM* migrations allocated on an individual host [43]. Interference Attentive Genetic Algorithm (IAGA) based VM allocation strategy has been proposed by [44] to reduce SLAV and performance degradation on the IaaS platform. Likewise, resource utilization management has been proposed using an agent-based algorithm in an IoT distributed environment [45]. Furthermore, a review of various numerical and hybridization of bio-inspired optimization techniques has been proposed in order to provide a consolidated platform for future research [46].

The inefficiency of the existing nature-inspired algorithms to meet-up in solving the challenge of resource consolidation calls for a promising ideal solution.

## 4. Multi-Objective Hybrid Resource Consolidation Algorithm

This section presents the design and development of an efficient resource consolidation algorithm that is capable of grouping the PMs using the clustering technique to organize the user requests based on their resource demand in order to reduce high energy consumption and SLA violation. The schematic structure of the proposed system is shown in Figure 1. The algorithm is composed of two main phases: the clustering phase, in which large resource management is carried out, and the resource consolidation phase comprised of VM allocation, which identifies the PMs to consolidate while monitoring the resources that will determine the target PMs. VMs are migrated to the targeted PM, and finally, the transition state decides on the consolidated candidate. A brief description of each phase is presented as follows.

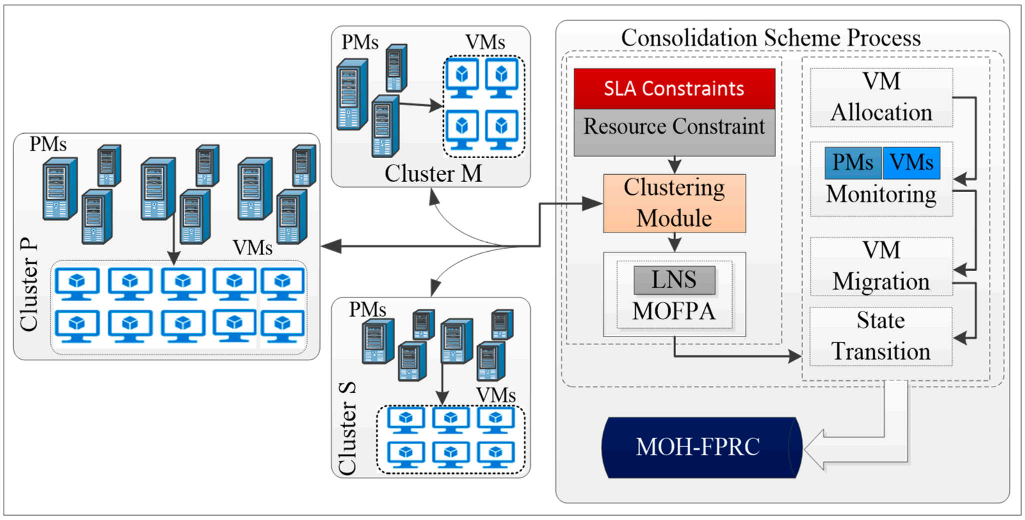


**Figure 1.** Schematic Diagram of MOH-FPRC.

The first phase is the Dynamic Clustering (DC) strategy, which groups the VMs requests into sub-groups based on user demand. Requests with similar requirements are placed on the PMs that are in the same cluster to reduce the need for resource migration, which also leads to energy consumption. This strategy determines where the user request will be executed based on the current resource utilization of a cluster type. The clusters are divided into three groups, either CPU intensive, Memory intensive, or Storage intensive. The second phase is VM resource consolidation, which uses LNS and the migration strategy. The LNS balance the local and global search procedure that avoids entrapment, while migration from overloaded PMs is used to avoid SLA violation. At the final stage of this phase, a monitoring module is employed to provide information on resource availability and clusters to determine the VMs that need to be migrated, those that need to be switched off/on by the transition state to increase resource utilization, or moved to a shutdown pool so they can be reinitialized later if user requests increase. In this way, the data center resource management in terms of energy efficiency and SLA violation could be improved.

*4.1. Mathematical Modelling of Objectives Functions*

This section incorporated the mathematical model of the SLA violation due to *VM* migration. Additionally, the energy model of PM is calculated based on the energy consumption model presented [47,48] using the ATTO Disk benchmark software solution. This is to ensure that the model addresses the conflicting objectives as addressed by the proposed algorithm. The Pareto optimization strategy was adopted to address this multi-objective problem. Obviously, resource consolidation decisions that result in fewer SLA violations can be a better choice. Therefore, the SLA violation due to the *VM* migration and the multi-objective consolidation are modelled as discussed in the following sub-section.

4.1.1. Service Level Agreement Model

The SLA violations due to the *VM* migration in Cloud data center environments are adapted from [49]. However, the authors only considered the migration of one resource. This research, on the other hand, considered not only the CPU but other resource components such as memory and storage. Furthermore, the SLA violation due to *VM* migration is divided into service violation per active PM, which is the percentage of time active PM experiencing 100% utilization of its resource components that lead to SLA violation. Another potential situation is when the active PM performance is below the expectation due to the *VM* migration, known as Performance Degradation (PD). SLA violation ($SLAV$) is modelled as in Equations (1)–(4),

$$SLAV(PD) = \frac{1}{M}\sum\nolimits_{j=1}^{M} \frac{\nabla_{VM_i}}{\Delta VM_s} \tag{1}$$

where $M$ is the number of $VMs$, $\nabla_{VM_i}$ is the estimate of $PD$ of $VM_i$ due to migration, and $\Delta VM_s$ is the total resource capacity of $VM_i$ during its execution time.

$$SLAV(PM_{active}) = \frac{1}{N}\sum\nolimits_{i=1}^{N} \frac{T_{PM_j}}{T_{PM_{j_{iactive}}}} \tag{2}$$

where $N$ is the number of PMs, $T_{PM_j}$ is the total time $PM_j$ reached 100% resource utilization, and $T_{PM_{j_{iactive}}}$ is the total time of $PM_j$ considered active for serving user request in form of $VM$.

Given the above estimation models for single *VM* migration overhead factors, the overall *SLAV* migration overhead for a particular *VM* consolidation plan for a cluster of PMs is modeled as the accumulated overhead of all the necessary *VM* migrations within the cluster:

$$SLAV = \left(\frac{1}{M}\sum\nolimits_{j=1}^{M} \frac{\nabla_{VM_i}}{\Delta VM_s}\right) + \left(\frac{1}{N}\sum\nolimits_{i=1}^{N} \frac{T_{PM_j}}{T_{PM_{j_{iactive}}}}\right) \tag{3}$$

Here, the sum of all time needed for the resource utilization for all *VMs* in data centers against active VMs is calculated for near average results.

$$SLAV = \frac{Total\ VM_i\ Request - Total\ AllocatedVM_i}{Total_{PM_{j_{active}}}} \tag{4}$$

4.1.2. Resource Consolidation Model

The aim of multi-objective resource consolidation is to simultaneously minimize energy consumption and SLA violation because of migrating VMs from one PM to another. Therefore, an index has been assigned to the consolidation technique as the evaluation parameters as follows:

Y is a consolidation parameter and $1 - $Y is the lack of a consolidation parameter.

The index shows the amount of resource energy consumption $(EU(t)_j)$ and SLA violation $\left(SVM\left(A_i, B_j\right)\right)$ due to resource consolidation $(R^c)$. Therefore, the resource consolidation index in the CC environment is as defined above, in which their coefficients ($\Phi$, $\Omega$, $\Psi$) are subject to change. According to consolidation importance, Y has the higher value and coefficients, $\Omega$ and $\Psi$, according to the type of system, and the importance of energy consumption and SLA violation will change. Eventually, the aim is to minimize the index $R^c$. When the number of $UR^n$ increases, the complexity rises, and while the allocated resources reach to their maximum utilization a lot of resources need to be relocated to improve performance. Consequently, $EU(t)_j$ and $SVM\left(A_i, B_j\right)$ increase. To overcome this limitation, the number of active PM resources is reduced by consolidating VMs, and $EU(t)_j$ is reduced drastically in the Cloud environment. Consolidation causes the reduction of $SVM\left(A_i, B_j\right)$ and then existing resources by allocation techniques as much as possible. Therefore, in principle, data centers' resource management requires consolidation in resource allocation to improve the overall data center resource utilization that reduces energy consumption and SLA violation.

The combined objectives' function of resource consolidation denoted as $R^c$, which minimizes energy consumption and SLA violation of the Cloud data center, can be mathematically formulated as follows in Equation (5).

$$R^c = \Phi(1 - conflict) + \Omega EU(t)_j + \Psi\ SVM\left(A_i, B_j\right) \tag{5}$$

where $\Phi$, $\Omega$, and $\Psi$ are the coefficient of the consolidation with the conflicting objectives, while $(1 - conflict)$ signifies that there is no consolidation.

$$\mathrm{Min}\left(R^c = \Phi\left(1 - conflict_{Bj}\right) + \Omega EU(t)_{Bj} + \Psi\ SVM_{Bj}\right) \tag{6}$$

Min $R^c$ is the objective function to minimize the overall energy consumption and SLA violation.

where:

$$EU(t)_j = \begin{cases} 1\ if\ UR\ assigned\ to\ A_i \\ 0\ \ else \end{cases}$$

$$SVM_{Bj} = \begin{cases} 1\ if\ VM_i\ migrate\ to\ PM_j \\ 0\ \ else \end{cases}$$

### *4.2. Hybrid Flower Pollination Resource Consolidation Algorithm*

The proposed MOH-FPRC scheme is designed and developed by improving the FPA Nature-Inspired algorithm. The algorithm uses the Dynamic Clustering (DC) algorithm to create clusters of VM requests based on the type of resources provided by the PMs through the Cloud management system. Additionally, the MOH-FPRC algorithm also employs the LNS strategy in order to avoid local optima entrapment in a local search, the *VM* migration strategy is employed, leading to better performance. Before presenting the LNS strategy, there is a need to introduce the existing FPA algorithm, which is very strong in terms of a global search strategy. This will provide the necessary background required to trail and understand the proposed hybrid solution.

#### 4.2.1. Flower Pollination Algorithm

One of the fascinating processes of reproduction in the natural world is flower pollination, and its evolutionary features are employed in the design of a unique optimization algorithm by [37]. It is a global optimization algorithm that is centered on population. The following steps summarize the flower pollination procedures.

*Procedure 1*: cross-pollination, which is termed as biotic, is considered global pollination carried out by pollinators called pollen vectors. The pollen vectors could be bats, birds, and bees. The pollen vector that carries pollen grain performs Levy flights.

*Procedure 2*: self-pollination is considered abiotic, which is a local pollination process. This pollination occurs on the same flower or nearby flowers.

*Procedure 3*: flower constancy is passive in terms of the reproduction chances, and is directly proportional to the resemblance of two flowers involved.

*Procedure 4*: the switch probability $p \in [0, 1]$ controls both the global pollination and local pollination. Considering the real-life proximity and other factors such as wind, a significant fraction $p$ in the whole pollination process can be achieved by local pollination.

Considering the above procedures, the FPA comprises local and global pollination operators. From the FPA, each pollen grain is handled as a solution, $S_l X_i$, and the solutions are set with non-uniform vectors in the viable search space. The preliminary formula is expressed in Equation (7) follows:

$$S_l X_i = lower + RD_v(upper - lower) \tag{7}$$

Such that $i \in \{1, \ldots, NP\}$, the population size, is represented as NP, and $RD_v$ is the random vector between $[0, 1]^D$ with D-dimension. The lower boundary of the search space is $lower = l_1, \ldots, l_d, \ldots, l_D$ and the upper boundary is $upper = u_1, \ldots, u_d, \ldots, u_D$.

Considering the global pollination process, pollen vectors such as birds have a comparatively large movement coverage and could carry pollens over a lengthy distance. Hence, procedures 1 and 3 are expressed as in Equation (8).

$$S_l X_i^{t+1} = S_l X_i^t + \propto L(\beta)\left(gbest - S_l X_i^t\right) \tag{8}$$

where $S_l X_i^t$ is the solution $i$ at the cycle of execution $t$, the present global best solution is termed as gbest, $\beta$ represents the step factor, the flight characteristics of a bird could be numerically mimicked by a levy distribution symbolized by $L(\beta)$ in Equation (3), and it can be perceived as a varying step factor for measuring the intensity of pollination. The levy distribution when $L > 0$ can be expressed as in Equation (9).

$$L \sim \frac{\beta \varnothing(\beta) \sin(\ \pi\beta/2)}{\pi} \cdot \frac{1}{S^{1+\beta}} \tag{9}$$

$$s \gg s_0 > 0,\ s_0 = 0.1\ ,$$

where $\varnothing(\beta)$ represents the standard gamma function with $\beta = 1.5$. In addition, $s$ is obtained with 2 Gaussian distribution $V$ and $W$ as follows in Equation (10).

$$s = \frac{V}{|V|^{\frac{1}{\beta}}},\ V \sim N\left(0, \mu^2\right),\ W \sim N(0, 1),\ \mu^2 = \left\{\frac{\varnothing(1+\beta)}{\beta\varnothing[(1+\beta)/2]} \cdot \frac{\sin(\pi\beta/2)}{2^{(\beta-1)/2}}\right\}^{1/\beta} \tag{10}$$

where $N\left(0, \mu^2\right)$ represents the normal distribution attached with the mean score 0 and variance $\mu^2$; $N(0,\ 1)$ represents the standard normal distribution.

In the case where the pollination process involves local pollination, pollen grains are spread to a local neighbor, and the mathematical model can be formulated considering procedures 2 and 3 as in Equation (11).

$$S_l X_i^{t+1} = S_l X_i^t + \vartheta\left(S_l X_j^t - S_l X_k^t\right) \tag{11}$$

where $S_l X_j^t$ and $S_l X_k^t$ are the pollen grain, which are non-uniformly selected from different flowers in the same plant, such that $j\ and\ k\ \in\ \{1,\ \ldots,\ NP\}$ and $\vartheta$ represent D-dimensional non-uniform vector in $[0,\ 1]^D$. Moreover, considering procedure 4, the two-pollination process happens non-uniformly and is obtained using a probability $p$. Thus, when a non-uniform value *rand* in [0, 1] is less than the value of $p$, global pollination is carried out. However, if the random value is higher than $p$, then global pollination is not carried out. The whole concept of the FPA is integrated with the LNS strategy as discussed further in Section 4.2.2. The following sections explain the phases of the proposed MOH-FPRC algorithm.

4.2.2. Local Neighborhood Search Strategy Phase

Previous neighborhood strategies were static and did not consider the whole search space that we integrated into the FPA. In this research, the LNS is dynamic, and each solution has a knowledge of the current and previous solutions. This reduces the time taken to find a solution. The LNS contains an $X$ and $Y$ axis that divides the search space into a fully connected neighborhood. The square contains vectors in which the structure explores to find other solutions instead of the whole population size. The search is carried out in a clockwise movement from the positive Y to the negative X direction within the neighborhood radius in which each neighbor shares their best solution. It is assumed that there exists a differential evolutionary population $P_G = [X_{1,S}, X_{2,S}, \ldots, X_{i+1,S}]$, and each $X_{i,S}(i = 1, 2, 3, \ldots, HQ)$ is a parameter vector with D-dimension. Each vector subscript index is randomly divided to ensure the diversity of each neighborhood. Figure 2 illustrates the structure of the LNS strategy of the MOH-FPRC algorithm, where every solution can be compared with each other.

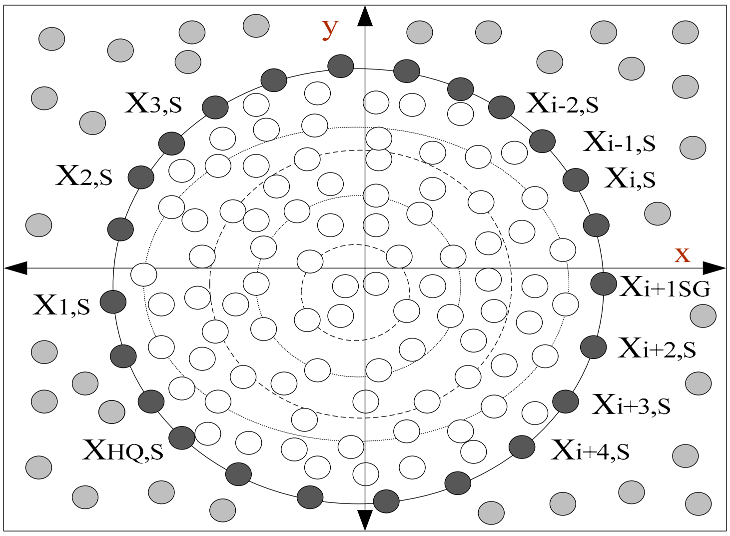


**Figure 2.** Local Neighborhood Search Structure.

For each vector $X_{i,S}$, which defines a neighborhood, and the radius $k$ $(2k + 1 < HQ)$. The neighborhood consists of vector $X_{i-k,S}, X_{i,S}, \ldots, X_{i+k,S}$. The assumption is that the vectors are in accord with the subscript indices in a loop structure formed within the search space. Hence, $X_{HQ,S}$ and $X_{2,S}$ are two direct neighbors of $X_{1,S}$. The neighborhood structure is dynamic and determined by the collection of vector subscripts. Thus, the LNS strategy can be expressed in the Equation (12).

$$X_{i+1,S} = X_{i,S} + \varnothing\left(X_{n-best_i S}\right) + \vartheta\left(X_{p,S} - X_{q,S}\right) \tag{12}$$

where $X_{n-best_i S}$ is the best vector of the $X_{i,S}$ neighborhood, $h, q \in [i-k, i+k](p \neq q \neq i)$, and $\varnothing$, $\vartheta$ are two scale factors.

4.2.3. Clustering Phase

Resource clustering is the group of PMs in the Cloud data center joined together and managed as a single resource pool, reserved for efficient infrastructure resource management and consolidation. The main aim of the Dynamic Clustering (DC) algorithm is to reduce the number of migrations, find the best cluster for $VM$ migration, and to support resource consolidation by identifying user requests with the same characteristics pattern. Figure 3 shows the conceptual framework of the clustering algorithm.

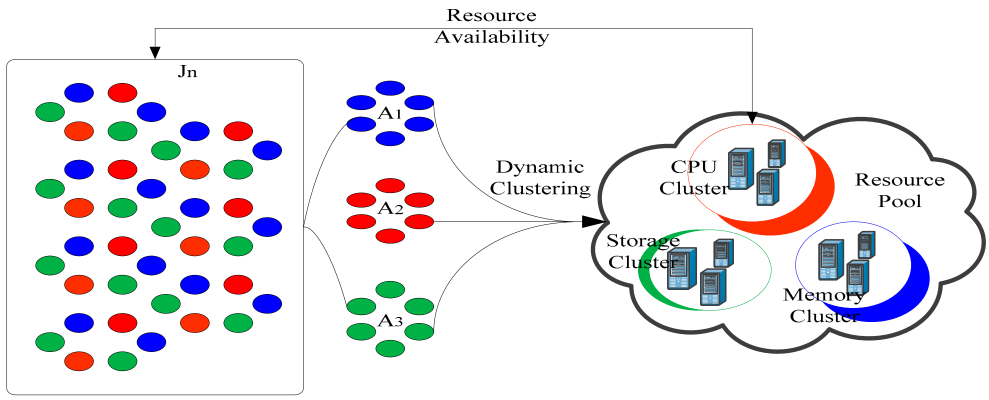


**Figure 3.** Dynamic clustering.

The clusters make independent and local decisions on intra/inter cluster migration in case a PM within a cluster should attain the utilization threshold. For example, consider a set of user requests $A_i \in \beth_n$ representing a partition $\beth_n = \{A_1 + A_2 + A_3 + A_n\}$ of $A_i$ into $n$ clusters. Therefore,

$$A_i \neq \complement\, i = \{1,\ 2,\ 3, \ldots,\ n\} \tag{13}$$

Here, the user request is not the same as the current type of request being executed for resource allocation.

$$A_i \cap\ \mathcal{L}_i = \complement\, i, \mathcal{L} = 1,\ 2,\ 3, \ldots,\ n,\ i \neq \mathcal{L}; \tag{14}$$

Therefore, the user request does not intersect and has no relationship with the current request,

$$\mathcal{H}_{i=1}^{n} A_i = J \tag{15}$$

Thus, these user requests are clustered into a group called $J$, where $\complement\, i = 1,\ 2,\ 3, \ldots,\ n$; These are denoted as different kind of requests. $\mathcal{L}_i$ is the distinct cluster group and $\mathcal{H}_{i=1}^{n}$ is the total distinct group within each cluster.

The DC strategy phase groups the VMs request based on user demand into groups of clusters using the Cloud management system. Requests with similar requirements are placed on the PMs that are on the same cluster to reduce network communication overhead that also leads to energy consumption. This strategy determines where the user request will be executed based on the current resource utilization of a cluster type. The clusters are divided into three groups, including the CPU intensive, Memory intensive, and Storage intensive clusters. Conversely, because of the dynamic characteristic of both user requests and resources, the volume of a cluster is tentative for more resourceful usage. At each point of the consolidation, the Cloud management system is used to determine the current number of available PMs in order to dynamically partition the VM request in accordance with the current number of available resources. Algorithm 1 shows the DC pseudo-code.

### 4.2.4. Virtual Machine Migration Phase

The *VM* migration technologies have proven to be a very effective tool for data center resource management in a non-disruptive manner, and their migration incurs SLA violations. Based on the above modeling analysis, the *VM* migration pseudocode has been presented as Algorithm 2. The proposed MOH-FPRC algorithm uses a *VM* migration strategy with a minimum number of migrations to reduce energy consumption and minimize the number of SLA violations. The main idea of the *VM* migration algorithm, after a request has been received, is to identify a target resource for in-cluster scaling. Then, the two PMs, one sending the request and the other accepting to be the target, negotiate the *VM* migration. Once an agreement has been reached, the two PMs carry out the operation without disrupting the power of the former for efficient resource utilization. Some of the main improvements introduced into this new approach to achieving the MOH-FPRC algorithm include (i) the introduction of the DC technique that helps to reflect on the most accurate number of available resources in the Cloud system. It also helps to divide the submitted request $\beth_n$ into an equal number of clusters as the number $n$ of available resources, and (ii) with the incorporation of the LNS strategy to balance the global and local search procedure for efficient optimization. In addition, (iii) the algorithm integrates the resource migration strategy in case of resource over-utilization during the consolidation. The proposed algorithm migrates resources to available target resources to reduce the number of migrations that results in fewer SLA violations.

**Algorithm 1** *Dynamic Clustering Algorithm*

**Require:** Combination of resource request
**Ensure:** CPU Cluster, Memory Cluster, and Storage Cluster
**Initialization**
**Get** $n$ from the Cloud management system
$\beth_n = \{A_1 + A_2 + A_3 + \ldots\ldots\ldots A_n\}$
$A_i = \{J\}\ i = 1, 2, 3, \ldots, n;$
$k = 0$
Let $\beth_n = \beth_{n-k/(n-k)}$
**Current Step:**
**While** $(n - k \geq 1)$ *or* $(\beth_n \neq \complement)$. *Do*
*Select* $A_i \mathcal{L}_i \in \beth_{n-k}$
$\beth_{n-k} = (\beth_{n-k} \cup \mathcal{L}_i)/(n-k)$
$k = k + 1$
**Get** current $n$ from Cloud management system
**End While**

**Algorithm 2** *VM Migration Algorithm*

**Require:** Active PMs in $n$ cluster, $VM$ migration $VM_m$, Migration time, Migration data
**Ensure:** *SLA violation due to PM migration*
1: **Initialization**
2: **For** each PM in PM list do
3: **For** each data of PM component (CPU, memory, storage) do
4: Select best *VM migration* strategy
5: Estimate the $SLAV = (PDM) + (PM_{active})$
6: Compute the predicted resource utilization of $PM$
7: **If** utilization $< 1$ **then**
8: **Repeat** step 2–step 7 **Else**
9: *VM is migrated*
10: **Migration time** $\longleftarrow$ **VM** get allocated
11: $VM$ started on targeted $PM$
12: $PM$ state change according to current utilization
13: **End**
14: **End**
15: **Return** (Number of VMs migration)
16: **SLAV**

#### 4.2.5. Implementation of Hybrid Resource Consolidation Algorithm

The proposed MOH-FPRC algorithm for the Cloud data center resource consolidation is presented as Algorithm 3 with the flowchart of the algorithm shown in Figure 4. The shapes represented with a white background indicate the existing algorithm, while the colors and dark gray indicate the improved part of the algorithm which enhances it. The MOH-FPRC algorithm used the DC and migration strategies to improve the resource management of the Cloud data center, satisfying conflicting objectives that are simultaneously optimized. The DC strategy creates clusters that map VM requests based on their application requirement. The multi-objective FPA optimizes the number of active PMs by finding a migration plan with a fewer number of VMs and initializing the best global migration plan at the global search space. The solution iterates to generate a new solution of flowers and switches to the local search using the switching probability. Each flower pollen at the local search chooses a solution appropriate to the cluster. If the solutions cannot be handled by the cluster, then the impact of the migration is computed based on SLA constraint. This step ensures that only high-quality solutions remain. The MOH-FPRC iterates until the stopping criterion is met and returns the best consolidation plan with lower migration and SLA violation.

**Algorithm 3** *Multi-Objective Hybrid Flower Pollination Algorithm*

**Require:** Set of population of *n flowers/pollen* gametes with random solutions
Find the best solution $g*$ in the initial population
**Ensure:** Define a switch probability $P = 0.6 - 1.0\ X\ \frac{(Max_{iteration} - t)}{Max_{iteration}}$
1: **Initializing:** use Algorithm 1 // Resource are clustered into CPU, memory, storage
2: Each Cluster is a single resource demand
3: *VMs* are classified based on requirement
4: **Input:** *PM list*, *VM*, set of parameters
5: **Migration Strategy:** use Algorithm 2 // *Resource and SLA violation constraint*
6: **Output:** *Consolidation*
7: $Objective\ Min\ \left(R^c = \Phi\left(1 - \text{conflict}_{Bj}\right) + \Omega \text{EU(t)}_{\text{Bj}} + \Psi\ SVM_{Bj}\right)$ //Equation (5)
8: **Initialize:** a population of *n flowers/pollen gametes* with random solutions
9: Find the best solution $g_*$ in the initial population
10: Define a switch probability $P$
11: **While** (t < *MaxGeneration*)
12: **For** $i = 1 : n\ (all\ n\ flowers\ in\ the\ population)$
13: **If** $rand < p$,
14: $Draw\ a\ (d - dimentional)\ step\ vector\ L\ which\ obeys\ a\ Levy\ \ distribution$
15: $Global\ pollination\ via\ X_i^t = X_i^t + L\left(g_* - X_i^t\right)$
16: **Else**
17: $Draw\ \in from\ a\ uniform\ distribution\ in\ [0,\ 1]$
18: $Randomly\ choose\ j\ and\ k\ among\ all\ the\ solutions$
19: **Do** $local\ pollination\ via\ X_{i+1,G} =$ $X_{i,G} + \varnothing\left(X_{n-best_i G}\right) + \quad \vartheta\left(X_{p,G} - \quad X_{q,G}\right) \quad where\ \varnothing = \vartheta = \in;$
20: **end if**
21: *Evaluate new solutions*
22: *If new solutions are better, update them in the population*
23: **end for**
24: Find the current best solution $g_*$
25: **End while**
25: **Termination criteria:** If the stopping criterion is satisfied, then output the content of archive as the optimal solutions otherwise Move to line 8.

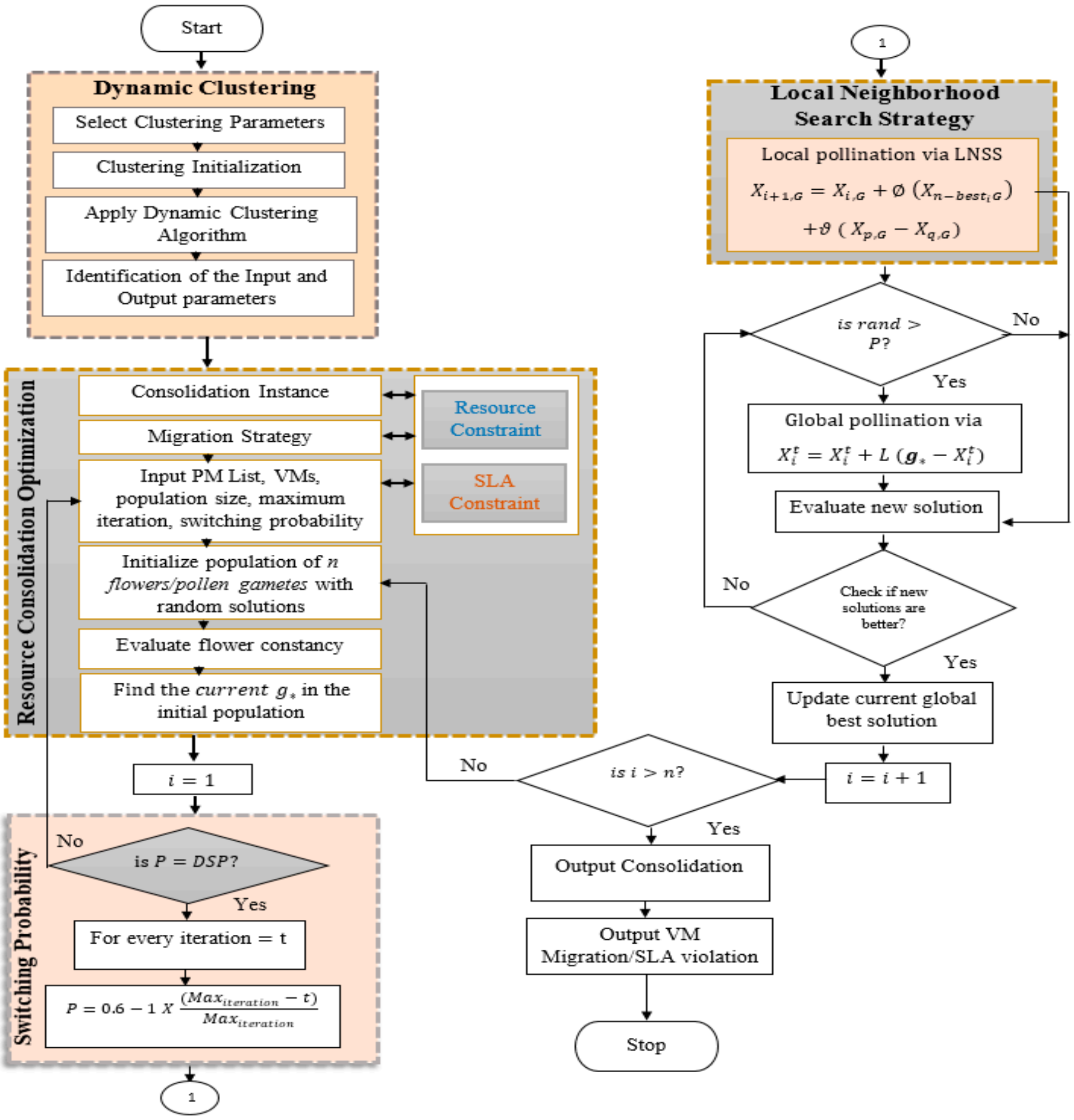


**Figure 4.** Proposed MOH-FPRC algorithm.

## 5. Performance Evaluation

Extensive simulation experiments were carried out in MultiRecCloudSim with the IntelliJ IDEA release version 3.4 to evaluate the performance of the MOH-FPRC algorithm. The general description of the simulation setup is presented in Table 1. The use of the DC and *VM* migration strategies in the proposed algorithm provide global optimum solutions for allocating VMs to PMs, minimizing the data center energy consumption. The proposed algorithm is different from other existing approaches in the way that it selects and migrates VMs by considering not only CPU as a metric but also the considered storage and memory. Therefore, the strategy used in the proposed algorithm of selecting the destination PM based on DC with a minimum number of *VM* migrations in this research is more energy-efficient than the existing approaches. The performance comparison of the MOH-FPRC algorithm is estimated in terms of *VM* migrations, SLA violations, and energy consumption. The experiments are performed by a varying number of PMs and VMs requests to evaluate the scalability of the MOH-FPRC algorithm. In the following sub-section, the experimental results of the proposed algorithm are discussed and analyzed. The simulation setup is presented in Table 1.

**Table 1.** Simulation settings: (**a**) PM and VM Parameters setting (**b**) Parameter settings of the compared Algorithm.

**(a)**

| Cloud Entity | Parameter | Value |
|---|---|---|
| Datacenter | Number | 1 |
| PM | RAM | 2,048,000 MB |
| | Disk | 10,000,000 MB |
| | Operating System | Linux |
| | Bandwidth | 1,000,000,000 MB |
| | Architecture | x86 |
| | VM Manager | Xen |
| | CPU Power Model | PowerModelSpecPowerX3550XeonX5675 |
| | Storage Power Model | PowerModelStorageSimple |
| | MemoryPower Model | PowerModelMemorySimple |
| VM | RAM | 2,048,000 MB |
| | Bandwidth | 0.1 GB/s |
| | MIPS | 367 MHz |
| | Storage | 1,000,000 MB |

**(b)**

| Algorithms | Parameter | Value |
|---|---|---|
| MOH-FPRC | Population size | 50, 100, 150, 200 |
| | Standard gamma function $\beta$ | 1.5 |
| | Random walk L | $\in$ [0, 1] |
| | Switching Probability $p$ [0, 1] | 0.6–1.0 |
| | Maximum iteration | 1000 |
| FPA | Population size | 50, 100, 150, 200 |
| | Standard gamma function $\beta$ | 1.5 |
| | Random walk L | $\in$ [0, 1] |
| | Switching Probability $p$ [0, 1] | 0.9 |
| | Maximum iteration | 1000 |
| MOACS/ACS-VMC | Population size | 50, 100, 150, 200 |
| | Crossover rate | 0.5 |
| | Pheromone tracking weight $\alpha$ | 0.3 |
| | Heuristic information weight $\beta$ | 1 |
| | Pheromone updating constant $Q$ | 100 |
| | Maximum iteration | 1000 |

*Result Analysis of MOH-FPRC*

This section presents the experimental results and analysis of the MOH-FPRC algorithm. The proposed algorithm was designed and developed to address resource allocation problems in a CC environment. This was achieved by considering energy consumption and SLA violation as the consolidation parameters. The experiments were repeated ten times for each of the algorithms, and the averages of *VM* migration, energy consumption, and the SLA violation rate (in percentage) were observed. The performance of the proposed algorithm, together with that of the benchmarked algorithms, were elaborated based on two multi-objective parameters, namely energy consumption and SLA violation. Figure 5 illustrates the trend of the performance of the consolidation algorithms.

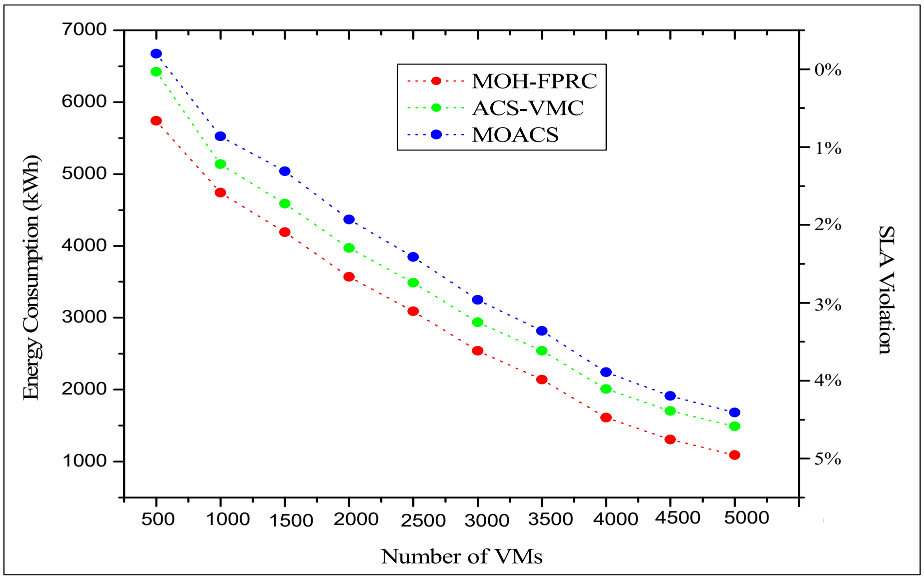


**Figure 5.** Comparison of algorithms Energy consumptions and SLA violation on VMs allocation.

As shown in Figure 5, a VM user request of 5000 VMs at the interval of 500 from 500–1000 and 1500–2000 is presented. All consolidation algorithms started at a promising point, with low energy consumption and SLA violations. However, as the number of VMs continued to increase, the energy consumption level on all consolidation algorithms increased significantly, together with the SLA violation. In all the consolidation algorithms, the MOACS algorithm tended to have the highest energy consumption at each increasing VM interval, as shown in Table 2. The ACS-VMC algorithm consumed less energy at the initial VM placement as compared with the MOACS algorithm. The differences can be seen in the results between the ACS-VMC and MOACS algorithms. However, the algorithm consumed more energy than the MOH-FPRC algorithm. This is because in the MOH-FPRC algorithm, VM placement takes into consideration the user request and places the request on the appropriate cluster of resources. This observation can be clearly described by the fact that the MOH-FPRC algorithm uses hybridization, which notably improves the output result as compared with benchmarking algorithms.

**Table 2.** Average energy consumption and SLA violation with different VM request.

| Algorithm | MOH-FPRC | | ACS-VMC | | MOACS | |
|---|---|---|---|---|---|---|
| VM Request | Energy Consumption (kWh) | SLA Violation % | Energy Consumption (kWh) | SLA Violation % | Energy Consumption (kWh) | SLA Violation % |
| 500 | 1089.25 | 0.00 | 1489.35 | 0.201 | 1589.45 | 0.220 |
| 1000 | 1304.05 | 0.00 | 1704.18 | 0.252 | 1804.11 | 0.251 |
| 1500 | 1609.32 | 0.15 | 2009.97 | 0.304 | 2109.76 | 0.312 |
| 2000 | 2139.84 | 0.20 | 2539.47 | 0.305 | 2639.34 | 0.325 |
| 2500 | 2539.21 | 0.25 | 2939.53 | 0.308 | 3039.62 | 0.339 |
| 3000 | 3089.59 | 0.30 | 3489.48 | 0.319 | 3589.46 | 0.339 |
| 3500 | 3569.35 | 0.35 | 3969.36 | 0.301 | 4069.64 | 0.342 |
| 4000 | 4189.87 | 0.40 | 4589.95 | 0.318 | 4689.89 | 0.342 |
| 4500 | 4739.65 | 0.42 | 5139.54 | 0.41 | 5139.41 | 0.401 |
| 5000 | 5739.14 | 0.48 | 6420.25 | 0.50 | 6200.96 | 0.50 |

Furthermore, as depicted in Table 3, the MOH-FPRC algorithm shows improvement in terms of PM utilization which leads to minimum energy consumption and less SLA violation after consolidation. In the table, the values indicate the energy consumption and SLA violation of the PM utilization of the tested workload. The data center energy consumption and the SLA violation of different algorithms is shown in Figure 6. The MOACS algorithm consumes higher energy with greater SLA violation compared with the ACS-VMC algorithm. However, the ACS-VMC consolidation algorithm seems more promising than the MOACS in terms of SLA violation and the amount of energy consumption. Moreover, the performance of the MOH-FPRC consolidation algorithm is outstanding at each increasing PM interval. The obtained energy consumption level by the MOH-FPRC is minimal compared to the benchmarked algorithms. This can also be seen in Table 4, which depicts the performance improvement (PI) of the MOH-FPRC algorithm as 14.54% and 29.48% better than the MOACS and ACS-VMC algorithms, respectively. The MOH-FPRC algorithm achieved the optimal balance in multiple conflicting objectives using the positive feedback mechanism and the constantly updated monitoring of resources, which effectively reduced the energy consumption and minimized violation of SLA. Likewise, the table shows a performance improvement of 13.57% for both MOACS and ACS-VMC algorithms in terms of SLA violation.

**Table 3.** Average energy consumption and SLA violation with different PM utilization.

| **Algorithm** | **MOH-FPRC** | | **ACS-VMC** | | **MOACS** | |
|---|---|---|---|---|---|---|
| **PM Utilization** | **Energy Consumption (kWh)** | **SLA Violation %** | **Energy Consumption (kWh)** | **SLA Violation %** | **Energy Consumption (kWh)** | **SLA Violation %** |
| 10 | 900.021 | 0.00 | 1275.25 | 0.21 | 1370.59 | 0.22 |
| 20 | 1115.50 | 0.01 | 1427.05 | 0.25 | 1522.36 | 0.25 |
| 30 | 1420.11 | 0.15 | 1845.32 | 0.34 | 1940.21 | 0.32 |
| 40 | 1950.01 | 0.20 | 2375.84 | 0.35 | 2470.58 | 0.35 |
| 50 | 2350.27 | 0.25 | 2775.21 | 0.38 | 2870.98 | 0.39 |
| 60 | 2900.14 | 0.31 | 3325.59 | 0.31 | 3420.74 | 0.39 |
| 70 | 3380.89 | 0.33 | 3805.11 | 0.38 | 3900.51 | 0.32 |
| 80 | 4010.22 | 0.37 | 4425.23 | 0.38 | 4520.45 | 0.32 |
| 90 | 4550.82 | 0.40 | 5075.87 | 0.41 | 5170.67 | 0.41 |
| 100 | 5150.42 | 0.43 | 5925.64 | 0.50 | 6100.07 | 0.50 |

Similarly, MOH-FPRC provides minimum SLA violation compared to the ACS-VMC and MOACS consolidation algorithms, as stated above. The reason is that the incorporation of the Pareto Optimization strategy provides better options for selecting PMs with less energy consumption within the proposed algorithm, helping to reduce the level of energy consumption and avoid high SLA violation. The VM selection and PM allocation policies within the proposed algorithm help facilitate the choosing of VMs for migration out from the overloaded PMs and selecting new PMs for the VMs to be migrated as shown in Table 3. The migration strategies used in this research by the MOH-FPRC and benchmarking algorithms are Static Threshold (ST) and Inter Quartile Range (IQR), and the strategy that returned the same performance in terms of the number of migrations and dynamic VM allocation strategies based on Median Absolute Deviation (MAD).

Table 5 shows the results generated due to the migration of VM resources using different strategies with different numbers of VMs at the interval of 625 VMs. The table reveals more details of the performance of the proposed algorithm on *VM* migration.

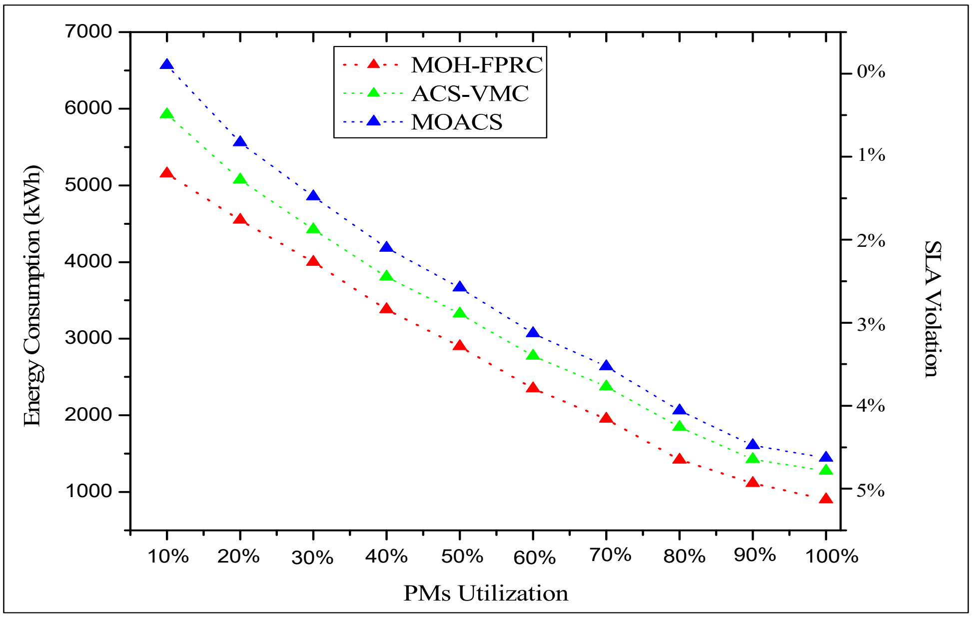


**Figure 6.** Performance comparison of PM Energy consumption, SLA violation and utilization.

**Table 4.** MOH-FPRC performance improvement on energy consumption and SLA violation.

| | MOH-FPRC | ACS-VMC | MOACS |
|---|---|---|---|
| Total average energy consumption and SLA violation | 5150.42 | 5925.64 | 6100.07 |
| | 0.43 | 0.50 | 0.50 |
| PI over ACS-VMC and MOACS (kWh) | – | 14.54% | 29.48% |
| PI over ACS-VMC and MOACS (SLA violation) | – | 13.57% | 13.57% |

**Table 5.** Average energy consumption and SLA violation with different number of PMs.

| Algorithms | | MOH-FPRC | | | ACS-VMC | | | MOACS | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Number of PM | Number of VM | MAD Number of Migration | ST Number of Migration | IQR Number of Migration | MAD Number of Migration | ST Number of Migration | IQR Number of Migration | MAD Number of Migration | IQR Number of Migration | ST Number of Migration |
| 100 | 625 | 287 | 69 | 122 | 300 | 210 | 420 | 323 | 400 | 239 |
| 200 | 1250 | 520 | 114 | 145 | 535 | 458 | 810 | 540 | 800 | 466 |
| 300 | 1875 | 754 | 128 | 170 | 760 | 513 | 835 | 761 | 850 | 520 |
| 400 | 2500 | 805 | 215 | 310 | 810 | 621 | 855 | 810 | 970 | 624 |
| 500 | 3125 | 973 | 248 | 384 | 985 | 785 | 1100 | 982 | 1140 | 786 |
| 600 | 3750 | 1222 | 291 | 390 | 1240 | 986 | 1225 | 1238 | 1220 | 997 |
| 700 | 4375 | 1298 | 315 | 410 | 1310 | 1125 | 1324 | 1312 | 1320 | 1136 |
| 800 | 5000 | 1356 | 357 | 456 | 1400 | 1265 | 1368 | 1405 | 1420 | 1329 |

The MOH-FPRC algorithm incorporated *VM* migration from overloaded PMs to avoid SLA violation as shown in Figures 7–9 under different migration strategies. The results produced by the benchmarking algorithms, namely MOACS and ACS-VMC, caused a higher number of migrations for resource consolidation as compared with the proposed MOH-FPRC algorithm. However, MOACS migrated VMs 1405 times, while the ACS-VMC did so 1400 times compared with the proposed MOH-FPRC algorithm.

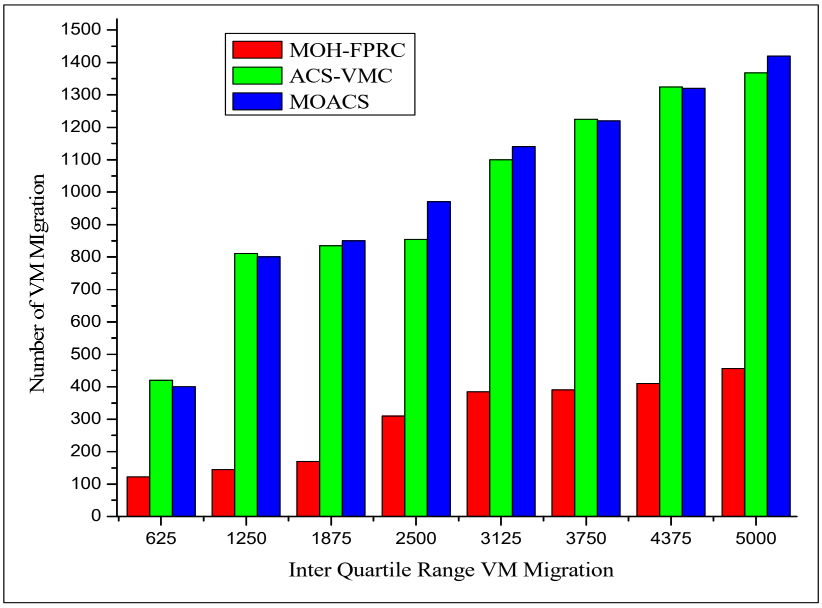


**Figure 7.** Comparison of number of *VM* migration using IQR.

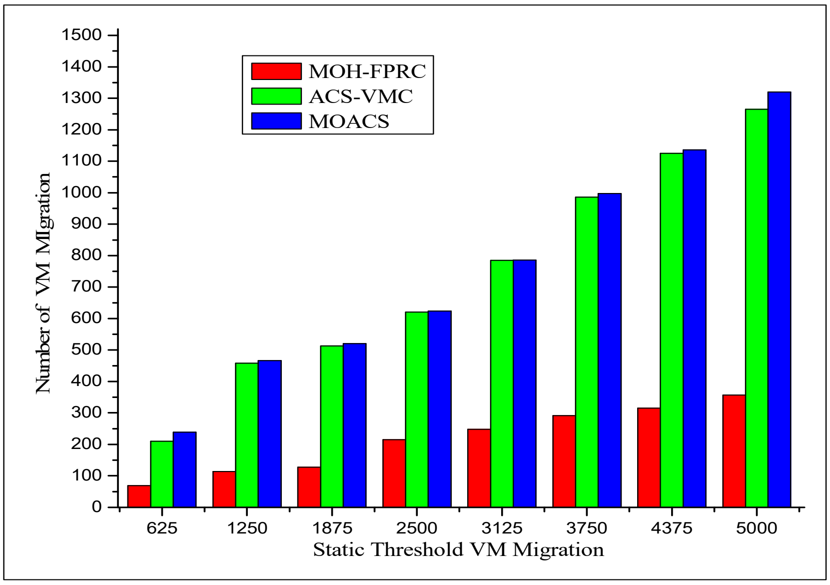


**Figure 8.** Comparison of algorithms using ST migration strategy.

This shows the reason why their energy consumption level (MOACS and ACS-VMC) increased significantly with SLA violation. Furthermore, the results indicate that, when the number of PMs increases due to the user request of VMs, the proposed MOH-FPRC algorithm performs better at minimizing the total energy consumption with SLA violation simultaneously as compared with the benchmarked algorithms. The main reason for that is the proposed algorithm employs a strategy to avoid migration that results in over-utilization of the destination PM. The total number of migrations by the proposed algorithm is less than that of the MOACS and ACS-VMC algorithm because of the use of DC and different migration strategies suitable for the Cloud data center resource management. More interestingly, there is a difference between the three migration strategies (ST, IQR, and MAD) as presented in Figures 7–9 by more than 30% higher due to the dependency of the MC strategy on upper and lower thresholds to perform the *VM* migration. This leads to the strategy migrating more VMs than the MMT strategy. Similarly, the MAD strategy remains the highest strategy in terms of VMs migration despite the availability of the PMs.

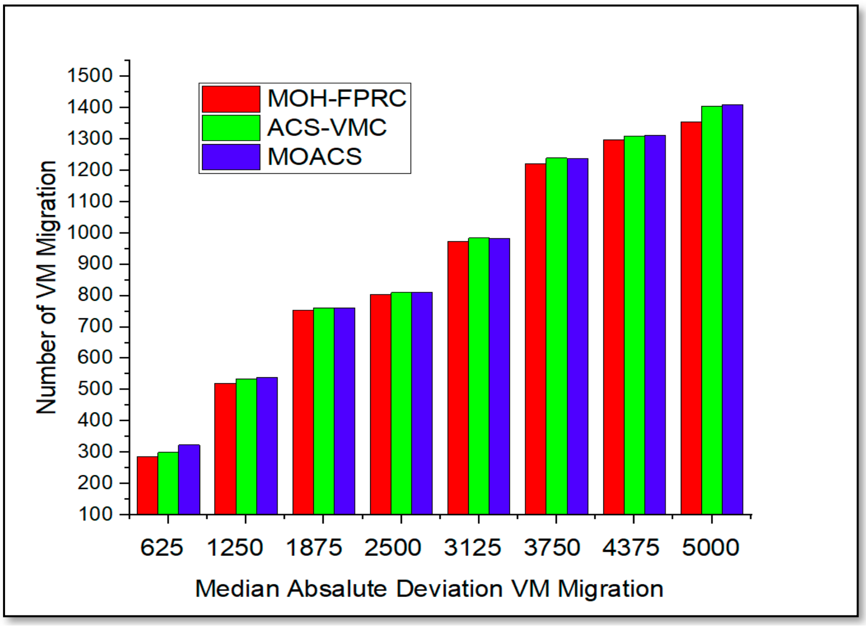


**Figure 9.** Comparison of algorithms using MAD migration strategy.

As can be observed from the above-mentioned figures, the performance improvements are a result of the incorporation of the LNS strategy in the local search of the conventional FPA, which significantly improved the convergence of underlying MOH-FPRC that avoid local entrapment. Therefore, the use of the LNS and DC methods efficiently enhances the search performance to obtain better non-dominated solutions. In this way, the data center resource management has been improved in terms of energy efficiency and SLA violation, and the impact on the data center has been equally improved. This proves that MOH-FPRC is an effective and efficient solution method for solving large-scale resource consolidation optimization problems.

## 6. Conclusions

In this article, we presented a MOH-FPRC algorithm. The proposed algorithm addressed conflicting objectives between energy consumption and SLA violation in the context of large-scale data centers to improve resource management. To achieve optimal resource consolidation, the MOH-FPRC algorithm incorporated the Pareto optimization strategy, DC strategy, resource monitoring, VM and PM selection, and a transition state. This resulted in efficient and reliable resource consolidation for CC data centers with minimum energy consumption and fewer SLA violations. The DC strategy maps VM requests based on their application requirement to assist in finding the appropriate cluster among the clusters that minimize the energy consumption and reduce the complexity that lead to local entrapment avoidance. The avoidance of entrapment is the result of the LNS strategy integrated into the FPA algorithm. The MOH-FPRC algorithm optimizes the allocation of VMs, monitors the performance of VM and PM, migrates VMs, and decides on the transition state of the resources. Furthermore, the proposed algorithm reduces the migration requirement of VMs to PMs, thereby making it robust and energy-efficient. Therefore, the proposed algorithm can effectively save energy consumption of the Cloud datacenter while reducing SLA violation rates compared to the benchmark algorithms. The simulation results significantly show the performance of MOH-FPRC as compared to similar nature-inspired consolidation-based algorithms using different performance evaluation metrics. However, the article also reveals a gap between Cloud user requirements and the IaaS provider in terms of resource management. Overall, utilizing CDC resource management techniques to build a more credible green data center with Fog computing is a promising research direction.

**Author Contributions:** Conceptualization, M.J.U. methodology, M.J.U., A.A., L.A.G. and D.G.; formal analysis, H.C.; investigation, L.A.G., D.G. and H.C.; resources, M.J.U., D.G. and H.C.; data curation, M.J.U.; writing—original draft preparation, M.J.U., A.A., H.C.; writing—review and editing, D.G. and H.C.; visualization, M.J.U., A.A. and L.A.G. All authors have read and agreed to the published version of the manuscript.

**Funding:** Funded by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2022R178), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

**Institutional Review Board Statement:** Not applicable to our study.

**Informed Consent Statement:** The study does not involve human subjects.

**Data Availability Statement:** The data is freely available online: https://github.com/beloglazov/planetlab-workload-traces (access on 4 March 2018). Simulator: IntelliJ IDEA release version 3.4.

**Acknowledgments:** This research was funded by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2022R178), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

**Conflicts of Interest:** The authors declare that there is no any conflict of interest.

## Abbreviations

| **Abbreviation** | **Meaning** |
|---|---|
| CC | Cloud Computing |
| IaaS | Infrastructure as a Service |
| PaaS | Platform as a Service |
| SaaS | Software as a service |
| PM | Physical Machine |
| VMs | Virtual Machines |
| PMs | Physical Machines |
| CPU | central processing unit |
| FPA | Flower Pollination Algorithm |
| LNS | Local Neighborhood Search |
| SLA | Service Level Agreement |
| FPRC | Flower Pollination Resource Consolidation |
| MOH-FPRC | Multi-Objective Hybrid Flower Pollination Resource Consolidation |
| ACO | Ant Colony Optimization |
| QoS | Quality of service |
| PSO | Particle Swarm Optimization |
| CSO | Cuckoo Search Optimization |
| SFLA | Shuffled Frog Leaping Algorithm |
| ACS-VMC | Ant Colony System-based VM Consolidation |
| MO-CSOA | Multi-Objective CSO Algorithm |
| VMC-ACO | VM Consolidation in Cloud data centers using ACO metaheuristics |
| MPSO | Modified PSO |
| UP-POD | utilization of resources through the host over-load detection |
| UP-PUD | host under-load detection |
| RL | Reinforcement Learning |
| DC | Dynamic clustering |
| $SLAV$ | SLA violation |
| $EU(t)_j$ | energy consumption |
| $SVM(A_i, B_j)$ | SLA violation |
| $R^c$ | resource consolidation |
| MOH-FPRC | Multi-Objective Hybrid Flower Pollination Resource Consolidation |
| DC | Dynamic Clustering |
| IQR | Inter Quartile Range |
| ST | Static Threshold |
| MOACS | Multi-Objective Ant Colony System |
| ICT | Information and Communication Technology |

## References


1. Foster, I.; Zhao, Y.; Raicu, I.; Lu, S. Cloud computing and grid computing 360-degree compared. In Proceedings of the 2008 Grid Computing Environments Workshop, Austin, TX, USA, 12–16 November 2008.
2. Xavier, T.C.; Santos, I.L.; Delicato, F.C.; Pires, P.F.; Alves, M.P.; Calmon, T.S.; Oliveira, A.C.; Amorim, C.L. Collaborative resource allocation for Cloud of Things systems. *J. Netw. Comput. Appl.* **2020**, *159*, 102592. [CrossRef]
3. Fister, I., Jr.; Yang, X.-S.; Fister, I.; Brest, J. A Brief Review of Nature-Inspired Algorithms for Optimization. *arXiv* **2013**, arXiv:1307.4186.
4. Srinivasan, J.; Dhas, C.S.G. Cloud management architecture to improve the resource allocation in cloud IAAS platform. *J. Ambient. Intell. Humaniz. Comput.* **2021**, *12*, 5397–5404. [CrossRef]
5. Lee, Y.C.; Zomaya, A.Y. Energy efficient utilization of resources in cloud computing systems. *J. Supercomput.* **2012**, *60*, 268–280. [CrossRef]
6. Gao, X.; Liu, R.; Kaushik, A. Hierarchical Multi-Agent Optimization for Resource Allocation in Cloud Computing. *arXiv* **2020**, arXiv:2001.03929. [CrossRef]
7. Chen, X.; Zhu, F.; Chen, Z.; Min, G.; Zheng, X.; Rong, C. Resource Allocation for Cloud-Based Software Services Using Prediction-Enabled Feedback Control With Reinforcement Learning. *IEEE Trans. Cloud Comput.* **2020**, *10*, 1117–1129. [CrossRef]
8. Zhao, Q.; Xiong, C.; Yu, C.; Zhang, C.; Zhao, X. A new energy-aware task scheduling method for data-intensive applications in the cloud. *J. Netw. Comput. Appl.* **2016**, *59*, 14–27. [CrossRef]
9. Usman, M.J.; Ismail, A.S.; Abdul-Salaam, G.; Chizari, H.; Kaiwartya, O.; Gital, A.Y.; Abdullahi, M.; Aliyu, A.; Dishing, S.I. Energy-efficient Nature-Inspired techniques in Cloud computing datacenters. *Telecommun. Syst.* **2019**, *71*, 275–302. [CrossRef]
10. 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. *Concurr. Comput. Pract. Exp.* **2012**, *24*, 1397–1420. [CrossRef]
11. Lovász, G.; Niedermeier, F.; de Meer, H. Performance tradeoffs of energy-aware virtual machine consolidation. *Clust. Comput.* **2013**, *16*, 481–496. [CrossRef]
12. Gong, S.; Yin, B.; Zheng, Z.; Cai, K.-Y. Adaptive Multivariable Control for Multiple Resource Allocation of Service-Based Systems in Cloud Computing. *IEEE Access* **2019**, *7*, 13817–13831. [CrossRef]
13. Mandal, A.K.; Kahar, M.N.B.M. An Energy-aware resource management scheme of Data Centres for eco-friendly cloud computing. *J. Adv. Appl. Sci.* **2015**, *3*, 107–112.
14. Devarasetty, P.; Reddy, S. Genetic algorithm for quality of service based resource allocation in cloud computing. *Evol. Intell.* **2019**, *14*, 381–387. [CrossRef]
15. Wei, J.; Zeng, X.-F. Optimal computing resource allocation algorithm in cloud computing based on hybrid differential parallel scheduling. *Clust. Comput.* **2019**, *22*, 7577–7583. [CrossRef]
16. Make IT Green. *Cloud Computing and Its Contribution to Climate Change*; Greenpeace International Annual Report; Greenpeace International: Amsterdam, The Netherlands, 2010.
17. Muthulakshmi, B.; Somasundaram, K. A hybrid ABC-SA based optimized scheduling and resource allocation for cloud environment. *Clust. Comput.* **2019**, *22*, 10769–10777. [CrossRef]
18. Kwon, S. Ensuring renewable energy utilization with quality of service guarantee for energy-efficient data center operations. *Appl. Energy* **2020**, *276*, 115424. [CrossRef]
19. Usman, M.J.; Ismail, A.S.; Chizari, H.; Abdul-Salaam, G.; Usman, A.M.; Gital, A.Y.; Kaiwartya, O.; Aliyu, A. Energy-efficient Virtual Machine Allocation Technique Using Flower Pollination Algorithm in Cloud Datacenter: A Panacea to Green Computing. *J. Bionic Eng.* **2019**, *16*, 354–366. [CrossRef]
20. Usman, M.J.; Ismail, A.S.; Gital, A.Y.; Aliyu, A.; Abubakar, T. Energy-Efficient Resource Allocation Technique Using Flower Pollination Algorithm for Cloud Datacenters. In Proceedings of the International Conference of Reliable Information and Communication Technology, Kuala Lumpur, Malaysia, 23–24 July 2018; Springer: Cham, Switzerland, 2018.
21. Joda, U.M.; Ismail, A.S.; Gital, A.Y.; Aliyu, A. Energy-Aware Distributed Multi-Cloud Flower Pollination Optimization Scheme. In Proceedings of the 2018 Seventh ICT International Student Project Conference (ICT-ISPC), Nakhon, Thailand, 11–13 July 2018.
22. Saber, T.; Ventresque, A.; Gandibleux, X.; Murphy, L. Genepi: A Multi-Objective Machine Reassignment Algorithm for Data Centres. In Proceedings of the International Workshop on Hybrid Metaheuristics, Hamburg, Germany, 11–13 June 2014; Springer: Cham, Switzerland, 2014.
23. Farahnakian, F.; Ashraf, A.; Pahikkala, T.; Liljeberg, P.; PLoSila, J.; Porres, I.; Tenhunen, H. Using Ant Colony System to Consolidate VMs for Green Cloud Computing. *IEEE Trans. Serv. Comput.* **2015**, *8*, 187–198. [CrossRef]
24. Ashraf, A.; Porres, I. Multi-objective dynamic virtual machine consolidation in the cloud using ant colony system. *Int. J. Parallel Emergent Distrib. Syst.* **2018**, *33*, 103–120. [CrossRef]
25. Liu, X.-F.; Zhan, Z.-H.; Zhang, J. An Energy Aware Unified Ant Colony System for Dynamic Virtual Machine Placement in Cloud Computing. *Energies* **2017**, *10*, 609. [CrossRef]
26. Gupta, P.; Goyal, U.; Verma, V. Cost-Aware ant colony optimization for resource allocation in cloud infrastructure. *Recent Adv. Comput. Sci. Commun.* **2020**, *13*, 326–335. [CrossRef]
27. Shooli, R.G.; Javidi, M.M. Using gravitational search algorithm enhanced by fuzzy for resource allocation in cloud computing environments. *SN Appl. Sci.* **2020**, *2*, 195. [CrossRef]

28. Durgadevi, P.; Srinivasan, S. Resource Allocation in Cloud Computing Using SFLA and Cuckoo Search Hybridization. *Int. J. Parallel Program.* **2020**, *48*, 549–565. [CrossRef]
29. Chen, X.; Wang, H.; Ma, Y.; Zheng, X.; Guo, L. Self-adaptive resource allocation for cloud-based software services based on iterative QoS prediction model. *Futur. Gener. Comput. Syst.* **2020**, *105*, 287–296. [CrossRef]
30. Praveenchandar, J.; Tamilarasi, A. Dynamic resource allocation with optimized task scheduling and improved power management in cloud computing. *J. Ambient. Intell. Humaniz. Comput.* **2021**, *12*, 4147–4159. [CrossRef]
31. Sait, S.M.; Bala, A.; El-Maleh, A.H. Cuckoo search based resource optimization of datacenters. *Appl. Intell.* **2016**, *44*, 489–506. [CrossRef]
32. Ferdaus, M.H.; Murshed, M.; Calheiros, R.N.; Buyya, R. Virtual machine consolidation in cloud data centers using ACO metaheuristic. In Proceedings of the European Conference on Parallel Processing, Porto, Portugal, 25–29 August 2014; Springer: Cham, Switzerland, 2014.
33. Li, H.; Zhu, G.; Cui, C.; Tang, H.; Dou, Y.; He, C. Energy-efficient migration and consolidation algorithm of virtual machines in data centers for cloud computing. *Computing* **2016**, *98*, 303–317. [CrossRef]
34. Hsieh, S.-Y.; Liu, C.-S.; Buyya, R.; Zomaya, A.Y. Utilization-prediction-aware virtual machine consolidation approach for energy-efficient cloud data centers. *J. Parallel Distrib. Comput.* **2020**, *139*, 99–109. [CrossRef]
35. Haghshenas, K.; Mohammadi, S. Prediction-based underutilized and destination host selection approaches for energy-efficient dynamic VM consolidation in data centers. *J. Supercomput.* **2020**, *76*, 10240–10257. [CrossRef]
36. Shaw, R.; Howley, E.; Barrett, E. Applying reinforcement learning towards automating energy efficient virtual machine consolidation in cloud data centers. *Inf. Syst.* **2022**, *107*, 101722. [CrossRef]
37. Yang, X.S. Flower pollination algorithm for global optimization. In Proceedings of the International Conference on Unconventional Computing and Natural Computation, Orléans, France, 3–7 September 2012; Springer: Berlin/Heidelberg, Germany, 2012; pp. 240–249.
38. Huang, Y.; Cheng, L.; Xue, L.; Liu, C.; Li, Y.; Li, J.; Ward, T. Deep Adversarial Imitation Reinforcement Learning for QoS-Aware Cloud Job Scheduling. *IEEE Syst. J.* **2021**, *10*, 1–11. [CrossRef]
39. Tong, Z.; Chen, H.; Deng, X.; Li, K.; Li, K. A scheduling scheme in the cloud computing environment using deep Q-learning. *Inf. Sci.* **2020**, *512*, 1170–1191. [CrossRef]
40. Chen, X.; Cheng, L.; Liu, C.; Liu, Q.; Liu, J.; Mao, Y.; Murphy, J. A WOA-Based Optimization Approach for Task Scheduling in Cloud Computing Systems. *IEEE Syst. J.* **2020**, *14*, 3117–3128. [CrossRef]
41. Tran, M.N.; Kim, Y. A Cloud QoS-driven Scheduler based on Deep Reinforcement Learning. In Proceedings of the 2021 International Conference on Information and Communication Technology Convergence (ICTC), Jeju Island, Korea, 15–19 October 2021; IEEE: New York, NY, USA, 2021; pp. 1823–1825.
42. Askarizade Haghighi, M.; Maeen, M.; Haghparast, M. An energy-efficient dynamic resource management approach based on clustering and meta-heuristic algorithms in cloud computing IaaS platforms. *Wirel. Pers. Commun.* **2019**, *104*, 1367–1391. [CrossRef]
43. Vaneet, G.; Jindal, B. Energy efficient virtual machine migration approach with SLA conservation in cloud computing. *J. Cent. South Univ.* **2021**, *28*, 760–770.
44. Bloch, T.A.; Rajagopal, S.; Ranga, P.C. IAGA: Interference Aware Genetic Algorithm based VM Allocation Policy for Cloud Systems. *Int. J. Adv. Comput. Sci. Appl.* **2022**, *13*, 754–765. [CrossRef]
45. Forestiero, A.; Papuzzo, G. Agents-Based Algorithm for a Distributed Information System in Internet of Things. *IEEE Internet Things J.* **2021**, *8*, 16548–16558. [CrossRef]
46. Gul, F.; Mir, I.; Abualigah, L.; Sumari, P.; Forestiero, A. A Consolidated Review of Path Planning and Optimization Techniques: Technical Perspectives and Future Directions. *Electronics* **2021**, *10*, 2250. [CrossRef]
47. Lin, W.; Xu, S.; He, L.; Li, J. Multi-resource scheduling and power simulation for cloud computing. *Inf. Sci.* **2017**, *397*, 168–186. [CrossRef]
48. Skendžić, A.; Kovačić, B.; Tijan, E. Effectiveness analysis of using Solid State Disk technology. In *Information and Communication Technology, Electronics and Microelectronics (MIPRO), Proceedings of the 2016 39th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), Opatija, Croatia, 30 May–3 June 2016*; IEEE: New York, NY, USA, 2016.
49. Ferdaus, M.H. Multi-objective, Decentralized Dynamic Virtual Machine Consolidation using ACO Metaheuristic in Computing Clouds. *arXiv* **2017**, arXiv:1706.06646.