applied sciences

Article

Workload Stability-Aware Virtual Machine Consolidation Using Adaptive Harmony Search in Cloud Datacenters

Ho Yeong Yun [1], Suk Ho Jin [2] and Kyung Sup Kim [1,*]

[1] Department of Industrial Engineering, Yonsei University, Seoul 03722, Korea; yhy900211@yonsei.ac.kr
[2] Division of Business Administration, Cheongju University, Cheongju 28503, Korea; shjin@cju.ac.kr
* Correspondence: kyungkim@yonsei.ac.kr; Tel.: +82-2-2123-4012

Abstract: Owing to the increasing complexity of managing IT infrastructure caused by rapid technological advancements, organizations are transforming their datacenter management environments from on-premises to the cloud. Datacenters operating in the cloud environment have large amounts of IT infrastructure, such as servers, storage devices, and network equipment, and are operational on all days of the year, thus causing power overconsumption problems. However, efforts to reduce power consumption are not the first priority as datacenters seek stable operation to avoid violating their service level agreements. Therefore, a research model that reduces power consumption of the datacenter while enabling stable operation by utilizing virtual machine (VM) consolidation is proposed here. To obtain the optimal solution for the proposed VM consolidation model, an adaptive harmony search methodology is developed, which expends less effort to set the parameters of the model compared to existing harmony search methods. Comparative experiments were conducted to validate the accuracy and performance of the proposed model. As a result, Original harmony search (HS) showed better performance than the existing heuristic algorithm, and novel self-adaptive (NS)-HS showed the best result among Adaptive HS. In addition, when considering workload stability, it showed better results in terms of datacenters (DC) stability than otherwise.

Keywords: cloud datacenter; VM placement; VM consolidation; adaptive harmony search

Citation: Yun, H.Y.; Jin, S.H.; Kim, K.S. Workload Stability-Aware Virtual Machine Consolidation Using Adaptive Harmony Search in Cloud Datacenters. *Appl. Sci.* **2021**, *11*, 798. https://doi.org/10.3390/app11020798

Received: 24 November 2020
Accepted: 11 January 2021
Published: 15 January 2021

1. Introduction

Datacenters (DCs) consume large amounts of energy for IT infrastructure operations, such as servers, storage devices, and network equipment, and also use energy to maintain an environment with constant temperature and humidity. DCs have significantly higher operating expenditures than capital expenditures and are considered to be the single largest power consuming buildings that account for approximately 2% of the global power consumption [1]. The recent growth of the ICT industry, along with the increasing demand for artificial intelligence, big data, and Internet of Things (IoT) has driven the transformation from on-premises to the cloud environment. Further, the number of DCs worldwide, which is about 8.6 million as of 2017, is expected to grow rapidly in the future. In fact, the Hyper-Scale DC, which is a datacenter with thousands of servers focused on processing and managing the exponentially increasing volumes of data, is expected to grow by about 53%, from 338 in 2016 to 628 by 2021 [2].

DCs must provide uninterrupted service for 365 days in a year, and the downtime cost per minute is estimated to be about $9000 [3]. In addition, malfunctions of DCs, which are associated with the social system, can cause secondary and tertiary damages, which are difficult to quantify in terms of scale. Therefore, the management of DCs has been conservative thus far instead of considering power-efficient operational measures. Examples of conservative management include allocating too many resources to react to less frequent potential peak loads when operating under the same original configuration from the beginning to end and allocating resources by estimating the demand growth of the cloud resource during the lifespan of the DC.

Power usage effectiveness (PUE) is an evaluation index of a DC's energy efficiency and is the ratio of the total power requirement of the DC to the power consumed by the IT equipment in the DC. A PUE value close to unity means higher energy efficiency [4]. Globally, companies have built and operated eco-friendly DCs utilizing geological advantages, such as using external air-conditioning in the arctic circle or building DCs at the bottom of the sea [5,6]. It also utilizes solar, wind, and renewable energy to operate DC efficiently [7]. These eco-friendly DCs show high energy efficiencies because heat management of the equipment requires less manual effort. However, as these DCs are built in remote areas, they are harder to access and respond to because of latency. Therefore, eco-friendly DCs with geological advantages are utilized as backups to analyze and store data rather than as main DCs for operating mission-critical systems. Thus, to improve the energy efficiencies of DCs, utilizing only geological advantages has limitations, and efforts must be considered to improve the operating systems internally.

Many studies related to computing systems for energy efficiency have been conducted at the DC, operating system (OS), virtualization, hardware, and firmware levels. Analyses of various methodologies to reduce the power consumption of the DC have shown that the most effective scheme involves a server-idle switch [8]. It has been shown that the server-idle switch has a significant effect on energy savings because leaving the server on and idling accounts for approximately 70% of total energy consumed in comparison to full utilization of the server [9].

Servers (physical machines: PMs) have higher specifications than personal computers, with the purpose of providing computing power for large volumes of data, and server virtualization technology is applied to efficiently utilize these resources. Server virtualization refers to logically dividing a PM into several virtual machines (VMs). As shown in Figure 1, virtualization runs the application by dividing the main OS into several smaller OSs through a hypervisor, which is a virtualization platform, unlike the traditional architecture of one OS for each server. The hypervisor allocates PM resources, such as central processing unit (CPU), memory, and network, to individual OSs. One unit of such an OS is called the VM. Most of the recently developed PMs for DCs are managed in virtualization environments.

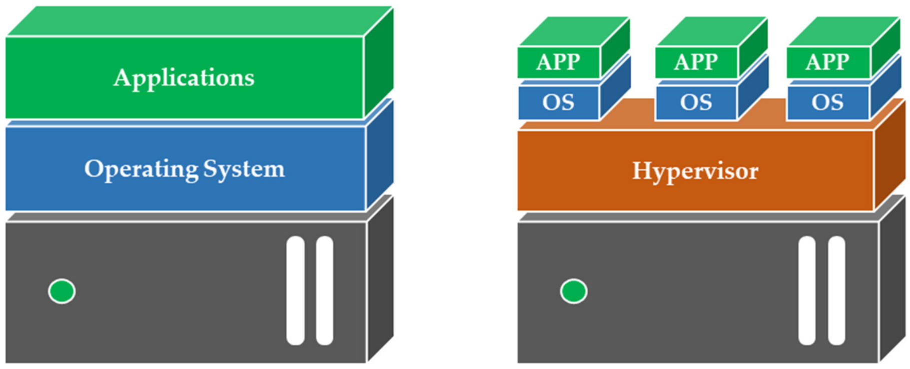


Figure 1. Traditional and virtual architecture server comparison.

The hypervisor configures the virtualization environment and offers live migration features that allow the operational VM to move between PMs without service interruptions. When a destination PM has available resources for the VM resource allocated to a source PM, then a VM can be migrated. When all the VMs in a certain PM are migrated to another PM, the original PM is empty and available for the idle switch. This methodology minimizes power consumption by migrating the VMs and switching a PM to idle, i.e., VM consolidation, which is one of the methods to effectively improve excessive power consumption by a DC [10].

VM consolidation maximizes PM idle switching by migrating the VMs to appropriate places, but the VM migration itself should be discouraged as much as possible [11,12].

Sometimes, downtimes occurring during VM migration can violate the SLA although applications installed in the VM continue operating during the migration [13,14]. In addition, migration itself is a task that induces overloading; hence, it is not recommended to migrate several VMs simultaneously. In the actual environment, this function is only utilized as cold migration after suspending the VM during a scheduled preventive maintenance. From the practical perspective of operating a DC, decision makers must have some insights into PM–VM mapping information. However, if this information changes frequently owing to VM migration, the complexity of management is intensified.

PMs generate large quantities of heat while handling tasks; this heat generated by a PM is proportional to its performance and power consumption, which means that high performance and high power consumption generate large quantities of heat. Excessive heat generated from the rack of a DC may cause equipment damage, which adversely affects the lifespan of IT equipment. Consequently, budget planning should be established for when and how much of the IT equipment must be replaced during the course of operating a DC. However, budget planning for DCs cannot be accurately implemented, and breakdown cannot be accurately predicted if the PMs are frequently replaced before expiration of their lifespans owing to performance degradation.

In this study, we propose a VM consolidation model to reduce the DC power consumption and minimize VM migrations. The VM consolidation model improves the power efficiency and ensures reliable operation of the DC. In addition, we explore VM combinations to steadily maintain PM workloads to enable reliable operation from the perspective of lifespan of the IT infrastructure. To effectively solve the proposed model, we apply the adaptive harmony search method to reduce the complexity of setting parameters among the various harmony search methods, which have shown excellent performance for optimizing various problems in the field.

The remainder of this paper is organized as follows. In Section 2, research related to VM consolidation and harmony search are reviewed. We propose the VM consolidation model in Section 3 and describe the design of the metaheuristic technique for solving the model in Section 4. Section 5 presents the experimental design and results, and Section 6 presents some conclusions and scope for future study.

2. Related Work

VM placement and VM migration are the backbone to the VM consolidation process [15]. VM placement involves deciding the PM to which a VM must be assigned and is similar to the multidimension bin packing problem (BPP), which concerns filling the maximum number of items into minimum number of bins. Here, the PM is considered as the bin, and the VM is considered as the item in VM placement. BPP can be a two-dimensional or three-dimensional problem, and the dimensions of the BPP comprise width, depth, and height. For VM placement, the number of dimensions increase according to the number of resources (CPU, memory, bandwidth, etc.). Specifically, VM placement is defined as a multidimensional vector packing problem with independence of each dimension, as shown in Figure 2. VM placement has the characteristic that a problem starts with a VM that is already assigned to the PM because the DC is operated throughout the year and differs from the BPP as it places the item in an empty bin [13,16].

The VM consolidation framework uses a separate monitoring tool, as shown in Figure 3, where the PM is monitored to minimize the number of active PMs using the idle switch in the event of an overload or underload. VM placement is then implemented if certain conditions are met. The key element of VM consolidation is VM placement, which consists of the source PM selection to determine which PM is placed in the power-saving mode, VM selection to determine which VM to migrate from the source PM, and destination PM selection to determine which PM is assigned the selected VM. It detects PM overload or underload; the detected PM then selects the VM to migrate from the VMs assigned to the source PM, and VM placement is implemented to decide the PM to migrate to.

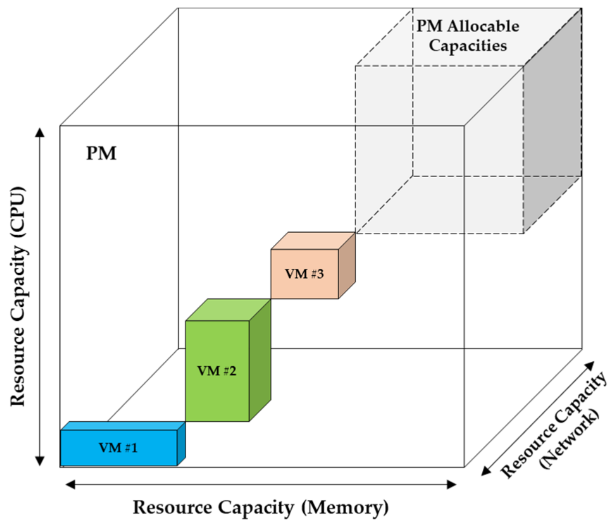


Figure 2. Multidimensional vector packing problem scheme.

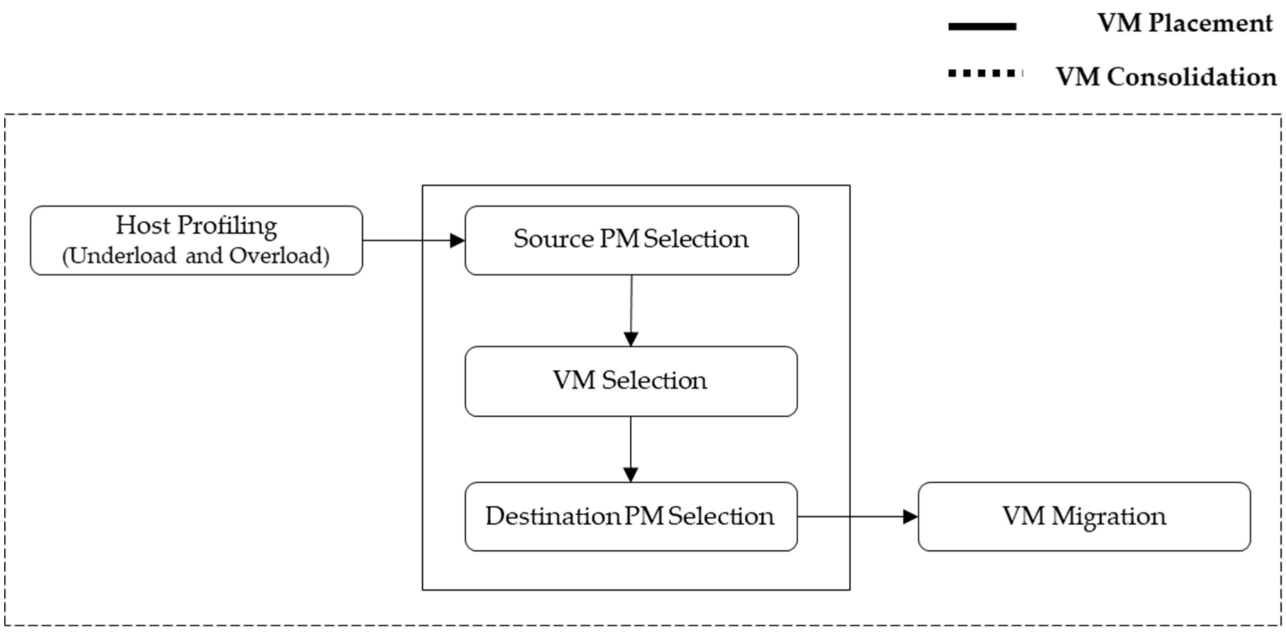


Figure 3. Virtual machine consolidation framework.

Threshold-based algorithm (THR), which is the most basic methodology to detect PM overload, calculates the average CPU usage for a specific time period *t* and determines that the PM is overloaded if the defined threshold is exceeded. The THR is a simple method but difficult to apply to dynamic and unpredictable environments [17]. Therefore, to overcome the THR limitations, an adaptive utilization threshold-based algorithm is suggested to analyze the collected workloads of the virtual machines statically and adjust the thresholds automatically [8,18]. The median absolute deviation (MAD) and interquartile range (IQR) are used as statistical measures. The MAD is a technique that uses the median instead of the average to find the absolute deviation, and IQR measures the overload using the quartile range. This is a general technique for measuring outliers, and the PM overload can also be interpreted in a manner similar to searching for outliers. The adaptive utilization threshold algorithm is more suited for dynamic environments than the static utilization threshold, but its overload forecast performance is poor. Local regression methods such as local regression robust (LRR) are suggested for better performances than the static and adaptive thresholds.

PM with overload detection is used to decide the VM to which migration must occur. The VM selection policies suggested are dynamic management algorithm (DMA) policy, which selects the VM with the lowest CPU utilization to minimize migration cost, minimum migration time (MMT) policy, which selects the VM with the lowest migration completion time, random selection (RS) policy, which selects a random VM [19], and maximum correlation (MC) policy, which selects the VM that has the highest CPU utilization correlation with the original VM [20]. Comparison of the results for each policy shows that the MMT has better results than RS and MC, suggesting that minimizing of VM migration time is an important objective. For DMA, MMT, and MC, the use of the fixed standard renders the policies unsuitable for dynamic environments. To improve these shortcomings, fuzzy Q-learning (FQL), which is an online decision-making strategy, is suggested. The FQL can choose one of the VM selection strategies dynamically to obtain better results than using the individual selection strategies alone [21].

VM placement determines the PM to which the selected VM is migrated, which is the same mechanism as the BPP as mentioned previously. The BPP is a typical NP-hard problem, and the solution exploration time exponentially increases with the size of the problem. Hence, studies proposing optimization techniques, such as integer programming (IP), linear programming (LP), integer linear programming (ILP), and binary integer programming (BIP), to solve for VM placement simplify the conditions and have long execution times. Consequently, most of the research related to VM placement apply the greedy heuristic or metaheuristic techniques, such as first fit decreasing (FFD) and best fit decreasing (BFD) strategies [22].

Adamuthe et al. designed a suitable genetic algorithm (GA) for VM placement and compared this with the FFD method [23]. Wu et al. utilized the genetic algorithm, NSGA, and NSGA II to maximize load balancing and profit and to minimize wastage of resources for VM consolidation [24]. Mark et al. predicted the VM resource demand and proposed the evolutionary algorithm to minimize cost by optimizing VM placement [25]. Ant colony optimization (ACO), which was inspired by the pheromone mechanism in ants, demonstrated high performance in various studies, but it was characterized to be an algorithm specialized for routing problems. However, the ACO method became one of the most well-known metaheuristic techniques that can be applied for VM consolidation since several works have defined the VM migration rule as the pheromone [26–28]. Farahnakian et al. suggested the green cloud computing method for the Ant colony system-based VM consolidation (ACS-VMC). They used the ant colony to find the most appropriate PM for VM placement after detecting the overload host using linear regression based on CPU usage prediction.

Kansal and Chana proposed the VM consolidation model to manage a cloud resource and increase its utilization by applying the artificial bee colony (ABC) method to solve the model [29]. This proposed VM consolidation model aimed to reduce energy consumption by referring to historical resource utilization and energy consumption data. Ibrahim et al. proposed particle swarm optimization (PSO) to reduce power consumption without violating SLA and showed excellent results in terms of SLA through a comparative experiment with Power-Aware BFD Algorithm [30]. Haghighi et al. used k-means clustering and proposed an improved micro-genetic algorithm. This effectively reduces the energy consumption of DC and simultaneously considers continuous quality of service. Comparative experiments were conducted with PSO and GA, and improved results were shown in terms of VM migration and make-span [31]. Kim et al. proposed the grouping harmony search (GHS) method to solve the VM consolidation problem by applying an efficient representation aimed at maximizing the migration efficiency, which is the ratio of the number of released PMs to the number of migrations [32]. The simulation is referred from a generator presented in an existing study, and comparison experiments between the harmony search (HS) and GHS are performed. Fathi and Khanli also applied the HS to solve the VM consolidation model by considering energy [33]. Renugadevi et al. proposed the power-aware algorithm and an adaptive HS to reduce the power consumption of the PMs [34]. They proved the efficiency of the HSA for the energy problem, but the simulations did not consider varying workloads.

This study (1) minimizes the power consumption of a DC by considering the violation of SLA and minimum VM migration simultaneously, (2) suggests the VM consolidation model that induces sustainability of the DC, and (3) apply new VM Selection rules the HS as an efficient method for solving the proposed model. Furthermore, (4) an adaptive HS is designed for simple parameter setup and comparative experiments are conducted to validate the proposed model.

3. VM Consolidation Model

VM consolidation is primarily aimed at increasing the power efficiency by dynamically switching the PM to idle, but its priority is low from the perspective of an actual DC administrator. The reason for this is that failures occur when operating enough number of PMs in the current environment. In this situation, reducing the number of PMs to increase the energy efficiency of the DC, where stability is the priority, increases the risk of failure in an uncertain environment, which is considered as a risk factor from the perspective of an administrator. In other words, energy efficiency is inversely related to stability of the DC from a conservative operational point of view. In addition, an administrator needs to be aware of PM–VM assignments, but frequent VM migration from VM consolidation can disrupt management efficiency. Therefore, this study proposes a VM consolidation model to maximize the operational stability considering the maximum energy efficiency of the DC, increase in the number of applications of the quality of services (QoS), and persisting period of the PM. This section describes the VM placement and VM migration, which are key elements of VM consolidation, and the objective function of this study.

3.1. Virtual Machine Placement

This study targets a virtualized environment that assigns PM resources to a large number of VMs and considers CPU, memory, and network as the resources. The CPU is the most influential factor for CM consolidation since it is the resource that has the largest impact on performance and energy consumption of the PM. Consequently, most of the research on VM placement consider the CPU. Memory is also considered here since it is the resource affecting the performance of the PM and is a major factor in VM migration. Swap memory in the storage drive is used when the PM does not have spare memory, which slows down, causing decrease in the QoS and failures. Network bandwidth is also an element to consider since its DC network environment is remote, but it is only considered as a resource affecting VM migration and not a resource that is assigned, like the CPU and memory.

The constraint for VM placement is defined as follows:

$$\sum_{i=1}^{P} x_{ij} = 1 \qquad \forall j, t \tag{1}$$

Equation (1) represents that the sum of every VM assigned to PMs is 1 and also indicates that only one VM can be assigned to a PM. x_{ij} is a binary variable representing whether the jth VM is assigned to ith PM (if it is assigned, 1; otherwise, 0).

$$\sum_{j=1}^{V} c_j \cdot x_{ij} \leq C_i \cdot thr \qquad \forall i \tag{2}$$

$$\sum_{j=1}^{V} m_j \cdot x_{ij} \leq M_i \cdot thr \qquad \forall i \tag{3}$$

Equations (2) and (3) are the constraints for the PM capacity, and the sum of the total resources (CPU and memory) of VMs assigned to ith PM cannot exceed the total amount of resources for the ith PM. In addition, the spare resources of the PM to prepare for a sudden increase in resource demand is reflected in the total amount of resources for the ith PM as a threshold.

3.2. Power Consumption Minimization for DC

The purpose of VM consolidation is to minimize power consumption for energy efficient management of a DC. In the recent operation environment of the DC, accurate power consumption data can be collected through the IoT sensors embedded in a server. However, it is difficult to identify the power consumption of a particular system. Similarly, new PM–VM mapping information obtained by the VM consolidation needs to be estimated since it is a plan that has not yet been implemented.

Fan et al. presented the power consumption estimation model to estimate the power consumption of a particular system by analyzing the power consumption data from thousands of PMs. The analysis results show strong correlations between power consumption and CPU utilization, and the power consumption tends to increase linearly when utilization of the CPU changes from idle to full utilization [35].

Based on the model by Fan et al., the estimated power consumption of the PM is calculated as shown in Equation (4).

$$W(i) = \begin{cases} W_{idle,\,i} + \left(W_{busy,\,i} - W_{busy,\,i}\right) \times \ c_i & \sum\limits_{j=1}^{V} x_{i,j} > 0 \\ 0 & \sum\limits_{j=1}^{V} x_{i,j} = 0 \end{cases} \tag{4}$$

$$W(i) = \int_0^t W(c_{jt})dt \tag{5}$$

Here, $W_{idle,\,i}$ and $W_{busy,\,i}$ represent the idle status and power consumption (W) of fully utilized ith PM and are constants defined according to the specifications of the PM. Power consumption of the PM $W(i)$ is calculated proportional to CPU utilization, and is $W_{idle,\,i}$ even when CPU utilization is 0%. In general, CPU utilization of 0% means that the PM is idle, and the power consumed can be conserved by setting this PM to the power-saving mode. CPU utilization in a 24-h operating DC environment is time-series data that varies according to user demand, and the cumulative amount of power consumed over a specific time period t ($\mathrm{PM_i}$) is calculated as shown in Equation (5).

3.3. VM Migration

The virtualized environment provides live migration, which supports non-disrupted service by the VM during migration to another PM. However, live migration decreases the performance owing to overload during the process of memory copying and causes a short downtime even though the application service does not stop, which is why frequent VM migrations should be avoided. VM migration is divided into precopy and postcopy at the point of VM interruption. Precopy is the commonly used technique, and its process is shown in Figure 4 [10].

Precopy migration begins with premigration and reservation to check whether the VM can be migrated. When migration is decided, the iterative precopy process transmits the memory pages repeatedly. All the memory pages are sent to the destination PM during the first iteration, and the modified memory pages (dirty pages) are sent repeatedly during subsequent iterations. The process of sending the memory pages causes overheads. During the stop and copy, the VM is suspended at the source PM and the entire network traffic is redirected to the destination PM. Then, the remaining dirty pages and kernel data, such as CPU states and registers, are sent. The commitment process confirms that the VM is migrated to a destination PM, which is then considered as the primary PM, and the VM that is in the source PM is abandoned. During activation, the VM is activated after appropriate actions for the destination PM, such as connection of the device drivers and change of IP addresses.

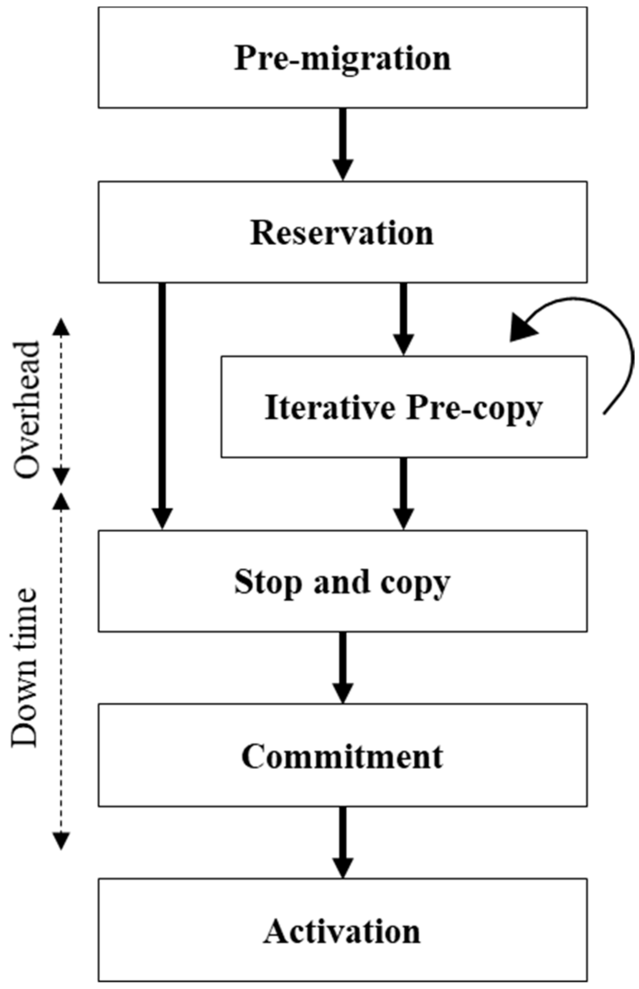


Figure 4. Procedure of precopy migration.

This study calculates the migration cost to avoid VM migration and reflects this in the VM consolidation model. Factors affecting the VM migration are the memory size of the VM, page dirty rate, CPU utilization of the source/destination PM, network bandwidth, and application sensitivity. Accurate calculation of the VM migration considering all of these factors have challenges. Therefore, this study considers the VM migration cost proposed by Wu et al., as shown in Equation (6) [36].

$$MC_j = m_j \times dRatio_j \times sen_j \tag{6}$$

VM migration sends the memory page, so it correlates with the memory size (m_j) of the VM. Page dirty rate ($dRatio_j$) is also considered to calculate the migration cost since a higher percentage of the page dirty rate increases the number of memory pages during migration. In addition, migrating a VM to the destination PM is achieved through the network associated with a network bandwidth. Application sensitivity (sen_j) is defined as a variable considering a network-intensive VM with the heavier burden of the migration process from providing streaming services compared to a VM that is not network-intensive.

The current allocation information of the PM–VM(x) is changed to x' according to the VM consolidation condition. VM migration cost from VM consolidation $MC(x')$ is calculated by Equation (7). The absolute value of the difference between x and x' indicates the number of changed VMs when the source and destination PMs are compared, and half of the absolute value represents the number of migrated VMs. The total VM migration cost $MC(x')$ is calculated by multiplying the number of migrated VMs and the migration cost MC_j. $Max(MC)$ represents the cost when all the VMs are migrated.

$$MC(x') = \frac{1}{2} \times \sum_{i=1}^{P} \sum_{j=1}^{V} |x_{ij} - x'_{ij}| \times MC_j \tag{7}$$

$$Max(MC) = \sum_{j=1}^{V} MC_j \tag{8}$$

3.4. Objective Function

The proposed VM consolidation model aims to reduce power consumption and avoid VM migration for stable DC operations. To evaluate the results of the VM consolidation model, the consolidation score, which unifies the values for the amount of energy and migration cost, is defined and calculated as shown in Equation (9).

$$\boldsymbol{minimize}\ f = \alpha \cdot \left(1 - \frac{W(x) - W(x')}{Min(W)}\right) + (1-\alpha) \cdot \frac{MC(x')}{Max(MC)} \tag{9}$$

The VM migration cost is evaluated by normalizing the number of migrations compared to the worst case, when all VMs are migrated. The variables evaluating the power consumption is normalized and the power saved is compared to the maximum power savings ($Min(W)$) based on the specification of the current PM and VM [37]. The maximum power savings concept is demonstrated in previous studies. Two of the multiobjective variables are calculated as one objective function using the weight variable α. This weight variable represents the conservatism of DC management; the low value of α chooses an operational plan to reduce the amount of energy by enduring some penalties and high value of α represents the small interest in reducing the amount of energy. The value of α is adjusted by the decision maker who determines the operation strategy of the DC.

4. Application of HS to VM Consolidation

VM placement has the same mechanism as BPP, which is an NP-hard problem, and its computational complexity increases as the number of nodes increases. Thus, efficient algorithm design should be considered to obtain a high-quality local optimal solution. The DC environment, which is the subject of the VM consolidation model application, is generally a large-scale problem as it is an environment operating with a large number of PMs and even larger number of VMs.

HS is an algorithm specialized for discrete problems, and has advantages such as easy implementation and quick convergence [38]. Compared to other metaheuristics, GA generates new chromosomes using only one or two existing ones, but HS is advantageous for exploration because it generates new solution based on all the solutions. In addition, unlike ACO, which is difficult to define a pheromone rule and requires updating the pheromone of all ants for every iteration, HS has the advantage of being able to quickly converge because it creates only one solution for each iteration. For this reason, HS was adopted.

4.1. Harmony Search

The HS is an algorithm that imitates the process of developing chords by improvised music players and is relatively simple compared to other algorithms; however, it shows high performance in optimal search in many engineering fields [38]. The algorithm is a population-based metaheuristic in which a solution is represented by the harmony memory (HM), and the number of HMs is determined by the harmony memory size (HMS). HS uses three operators to develop from dissonance to harmony. The operators comprise memory consideration (MC) that extracts the sound played in the past, pitch adjustment (PA) that allows adjustments to the extracted sound within the range of the bandwidth (BW), and randomization that allows random plays. The process of generating solutions for each operator is shown in Equations (10) and (11).

$$x_i^{New} = \begin{cases} x_i \in \left[x_i^1,\ x_i^2,\ \ldots\ x_i^{HMS}\right] & \text{IF } rnd1 \leq \text{HMCR} \\ x_i \in \left[x_i^{Lower},\ x_i^{Upper}\right] & \text{IF } rnd1 > \text{HMCR} \end{cases} \tag{10}$$

$$x_i^{New} = \begin{cases} x_i^{New} + BW & \text{IF } rnd2 \leq \text{PAR} \\ x_i^{New} & \text{IF } rnd2 > \text{PAR} \end{cases} \tag{11}$$

HS generates the new HM x_i^{New} in order starting at index i. First, it generates first random number between 0 and 1 to determine whether to generate a solution using the HMC or randomly. If it is smaller than or equal to the HMCR, one of the HMs is selected to generate x_i^{New}. The second random number is generated, and if it is smaller than the PAR, a neighborhood solution that is calculated by adding a random value to x_i^{New} within the appropriate range (BW) is generated as x_i^{New}. If the first random number is greater than the HMCR, the solution is generated randomly in a feasible range. If a newly generated HM x_i^{New} is a better solution than the worst HM from the existing HMs, the newly generated HM replaces the existing worst HM. The process is then repeated a certain number of times to obtain the best HM.

4.2. Harmony Search for VM Consolidation

VM placement allows decisions to select the source PM for power saving (or other purposes), selects the VM to migrate from the selected PM, and selects the destination PM to which the VM migrates. The proposed HS is described in this section and is designed to efficiently solve the VM placement model. The harmony considering rate (HMCR), which determines whether to create the new harmony based on the existing harmony or to generate randomly, pitch adjustment rate (PAR), which determines whether to consider the neighborhood values from the existing harmony, and bandwidth (BW), which determines the range of the neighborhood value, are the HS parameters that must be set separately.

4.2.1. Solution Representation

For VM placement, HM contains the PM–VM mapping information, which is the VM assignment to PM information. One HM is in a permutation form, where each array represents the VM index, and each index value represents the PM index to which the VM is assigned. This solution representation is schematically shown in Figure 5.

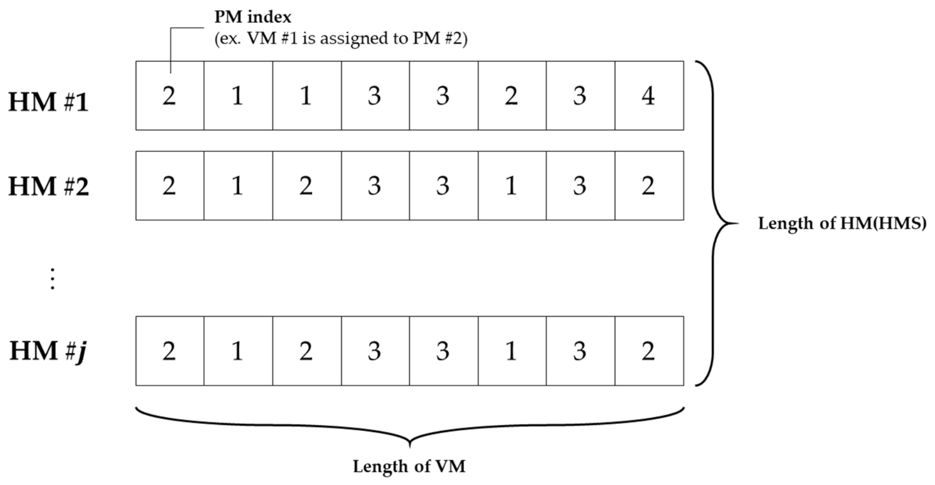


Figure 5. Solution representation for physical machine (PM)–virtual machine (VM) mapping.

4.2.2. Pitch Adjustments of HS for VM Consolidation

For HS, the most influential operator that creates a significant neighboring solution is the PA. The HMC is the standard for neighborhood search, and randomization contributes to avoiding the local solution. Their suitable combination plays an important role in creating an improved solution. The PA is designed to demonstrate high performance and search for an efficient solution for the VM consolidation model. In VM consolidation, VM migration occurs when either overload or underload of the PM workload is detected. The contrapositive of the proposition mentioned above becomes "if overload or underload of the PM workload does not occur, then VM migration is not necessary". Ultimately, if the PM workload remains constant, the VM need not migrate. As shown in Figure 6, if a VM with the inverse workload is combined, the PM workload remains constant, and the VM can achieve sustainable operation by minimizing DC power consumption and VM migration, which is the objective of VM consolidation. The measure indicating the degree of stability of the PM workload is calculated as the standard deviation of the workload time-series data. The closer the standard deviation is to 0, the more stable is the workload.

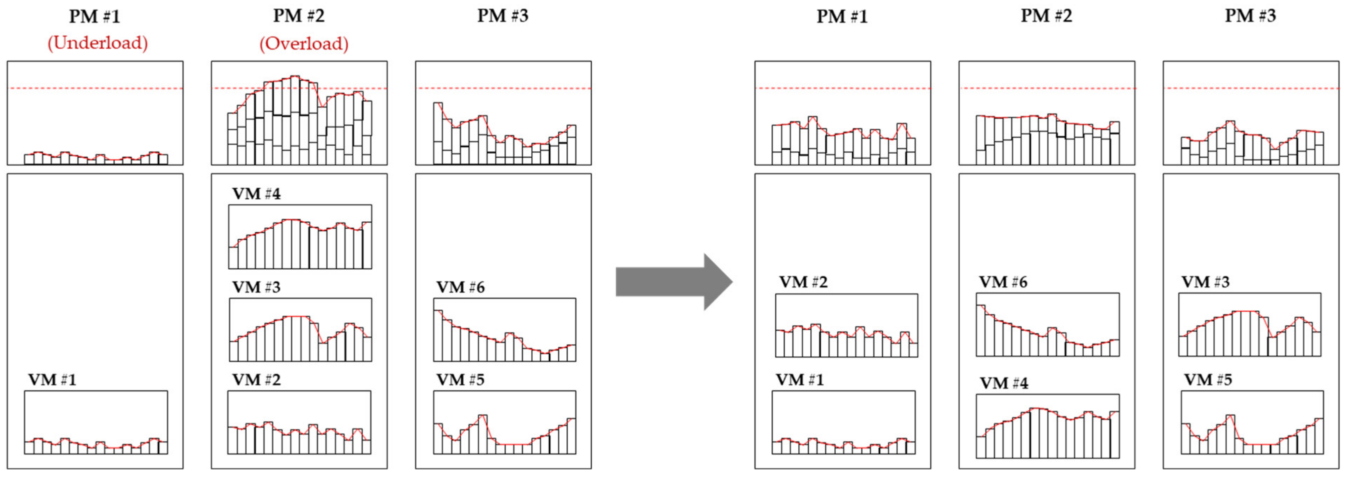


Figure 6. VM consolidation with stable PM workload.

When generating a new HM, the first random number is generated based on the length of the VM, and whether MC or randomization is performed is determined collectively. Next, the MC generates a second random number for the determined VM index and determines whether it is a PA. Applying PA to the corresponding VM means selecting the VM to be migrated, which corresponds to VM selection in VM placement. The VM selection consists of three cases.

First, a VM allocated to a PM with detected overload is selected. When PM overload occurs, it violates the SLA, so the VM of the PM must be migrated. For this reason, the VM located in the PM where overload is to be detected is always selected. Second, a VM allocated to a PM that can be idle is selected probabilistically. Since VM consolidation aims to minimize the power consumption of the DC, it is necessary to select a PM that can idle the server. A PM capable of idle conversion is a PM that has an assigned number of VMs of 1 or less or is classified as underload by a specific condition (THR, MAD, IQR, etc.), and a VM assigned to the corresponding PM is selected probabilistically. Third, when a specific VM is migrated, a VM with high workload stability is selected probabilistically. When the VM is migrated to each PM, the PM stability ($varz$) measure calculates the workload of the PM (z_{it}) by summing the workload of the VMs (c_{jt}) assigned to each PM as shown in Equation (12), the average of the workload for all PMs ($\overline{Z}$) as shown in Equation (13), and is calculated as the standard deviation between each workload and the average as shown in Equation (14).

$$z_{it} = \sum_{j=1}^{V} \left(c_{jt} \cdot {x'}_{ij} \right) \tag{12}$$

$$\overline{Z} = \sum_{i=1}^{P} \sum_{t=1}^{T} z_{it} / T \tag{13}$$

$$varz = \frac{\sum_{t=1}^{T} \left(z_{it} - \overline{Z} \right)^2}{T} \tag{14}$$

The smaller the value, the higher is the stability measure, and it is considered a VM with a higher priority to migrate to another PM. Among the VMs that can be migrated, a higher level VM with a high stability measure is randomly selected. This VM selection rule was defined as a stability correlation (SC) policy. A second random number is generated for the VM index selected under the above three conditions, compared with PAR, and whether or not the PA is determined.

The VM whose PA is determined generates a neighboring solution by adding the values in the BW, which is the same as the concept of migrating the VM to the destination PM. The feasible PM list that can migrate the VM thus becomes the bandwidth, and the roulette-wheel-based weighted selection is made based on the stability measure.

4.2.3. Procedure of HS for VM Consolidation

The proposed VM consolidation model aims to achieve its original purpose of reducing the DC power consumption while deriving stable operation by searching for combinations of stable VM and reducing unnecessary VM migrations. To explore the stable workload combinations hereafter, an environment that can predict the workload is assumed based on the VM workload collected. Workload prediction is an essential process for resource allocation, capacity planning, and energy savings in dynamic cloud computing environments [39]. VM consolidation utilizes a separate monitoring tool to detect overload or underload of the PM workload. When certain conditions are met, VM placement is implemented based on the prediction data and resource information of the VM workload, and HS is utilized to find the optimal PM–VM mapping information. The pseudo code representing the proposed VM consolidation model is shown in Table 1.

To implement VM placement, the initial value for the HS parameters (HMS, HMCR, PAR, and BW) are set at the beginning of the algorithm (line 3), PM–VM mapping information for consolidation is entered and duplicated for the same number of times as the HMS (line 4). The HS method uses the MC, PA, and RS to generate a new HM with improved objective value through repeated iterations (line 7–37). The new HM is generated sequentially through HS operators for the length of HM (number of VMs) (line 9–28). The first number is randomly generated between 0 and 1 to determine the existence of the MC and randomization collectively (line 10–15). Then, the second random number is generated to determine the existence of PA (line 22–28). If the second random number is smaller than PAR, then the neighborhood solution is generated using PA (line 29–30). The method of generating the neighborhood solution and the conditions of determining the PM that is migrated to the VM are described in Section 4.2.2. If the new HM generated has a better objective value than the worst HM from the existing HMs, the new HM is replaced with the worst HM (line 34–36). Although the objective value is worse, even when the stability of the PM is improved, the worst HM is replaced with the new HM. When the stopping condition is met, the HS terminates and VM migration occurs with the newly generated PM–VM mapping result (x') (line 38).

4.3. *Adaptive Harmony Search for Parameter Setting*

Metaheuristic algorithms can be applied to a variety of problems, but the process of appropriate parameter setup for different problems should be performed. In addition, since these algorithms are based on probability, it is common to conduct statistical analysis on repeated experiments; the number of iterations here can be a disadvantage considering the combinations of parameters. Despite these efforts for the solution quality, the parameters cannot guarantee solution quality with slight changes in the nature of the problem. To overcome this disadvantage, a self-adaptive HS, which explores and obtains the parameters itself, is widely being studied. This study addresses the parameter-setting-free harmony search (PSF-HS), novel self-adaptive harmony search (NS-HS), Improved Harmony Search (I-HS) and parameter adaptive harmony search (PA-HS) to reduce the complexity of setting the parameters [40–43].

Table 1. Pseudo code for harmony search (HS) in VM consolidation framework.

Pseudo Code for Harmony Search

```
1:  Input: List of PMs, VMs, $x$ (VMs are allocated PMs before consolidation)
2:  Output: $x'$ (VMs are allocated PMs after consolidation)
3:  Initialize the DC status and Harmony Search parameters
4:  Replicate the vectors of the Harmony Memory on $x$, HM = $\{x^1, x^2, \ldots, x^{HMS}\}$
5:  Calculate the objective function $f(x^1, x^2, \ldots, x^{HMS})$
6:  Recognize the worst vector in HM, $x^{worst} \in \{x^1, x^2, \ldots, x^{HMS}\}$
7:  REPEAT
8:      $x' = \phi$
9:      FOR $i$ = 1 to VMs.Length
10:         IF (Rand() $\leq$ HMCR) THEN        // Memory Considering
11:             $x'_i = x_i^{U[1,HMS]}$
12:           $x'_i$.is HMCR = true
13:         ELSE                                // Random Playing
14:           $x'_i = X_i$                      // Select from feasible PMs
15:         END IF
16:     END FOR
17:     FOR $i$ = 1 to VMs.Length
18:         Calculate workload stability by PM when VM is migrated
19:         $BW_i \in \{SC^1, x_i^2, \ldots, x_i^{PM}\}$     // workload stability of the $i$th VM
20:     FOR $i$ = 1 to VMs.Length                          // Pitch Adjusting
21:       IF ($x'_i$. is HMCR == true) THEN // VM Selection
22:             IF (PM[$x'_i$].is Overload) THEN
23:                 $x'_i$.is PAR = true
24:             ELSE IF (PM[$x'_i$]. is Underload && Rand() $\leq$ PAR) THEN
25:                 $x'_i$. is PAR = true
26:             ELSE IF(Average($BW_i$) < Top 10% && Rand() $\leq$ PAR) THEN
27:                 $x'_i$. is PAR = true
28              END IF
29:             IF ($x'_i$. is PAR == true)           // Destination PM Selection
30:                 $x'_i = BW_i^{Roullette[1,PM \times BandwidthRate]}$
31:             END IF
32:       END IF
33:     END FOR
34:     IF ($f(x') < f(x^{worst})$) then
35:         Replace $f(x^{worst})$ with $f(x')$
36:     END IF
37: END REPEAT
38: Find the current best harmony ($x'$)
```

The PSF-HS, introduced by Geem et al., implements the operation type memory (OTM), which stores the operator that improves the solution during iterations. It updates the HMCR and PAR by remembering the operator contribution to the generation of the improved new HM among the HMC, PA, and RS. The characteristic of increasing HMCR and decreasing PAR with increasing number of iterations is maintained to improve search efficiency by increasing the global search at the beginning of the iteration and increasing the local search towards the end of the iteration.

$$HMCR_i = \frac{n\left(y_i^j = HMC\right)}{HMS}$$

$$PAR_i = \frac{n\left(y_i^j = PA\right)}{HMS}$$

$$OTM = \begin{vmatrix} y_1^1 = RS & y_2^1 = PA & \dots & y_n^1 = HMC \\ y_1^2 = HMC & y_2^2 = RS & \dots & y_n^2 = RS \\ \dots & \dots & \dots & \dots \\ y_1^{HMS} = HMC & y_2^{HMS} = HMS & \dots & y_n^{HMS} = PA \end{vmatrix}$$

Based on the NS-HS proposed by Luo, which finetunes the improvisational music by a musician from the beginning to end, a self-adaptive variant that does not consider the PAR is proposed. The HMCR is calculated by substituting n, the dimension of the problem. The BW is calculated by utilizing the number of iterations k, number of improvisations NI, and standard deviation of the HM objective function values f_{std} to reflect the finetuned characteristic of exploring the solution wider at the beginning and narrower towards the end. The generation of the new HM uses two random numbers, $U[0, 1]$ and $U[-1, 1]$, for the HMCR and dynamic BW.

$$HMCR = 1 - \frac{1}{n+1}$$

$$x_i^{New} = \begin{cases} x_i^j + U[-1,\ 1] \times\ BW_i & if\ r < HMCR \\ BW_i^{Lower} + U[0,1] \times \left(BW_i^{Upper} - BW_i^{Lower}\right) + U[-1,1] \times BW_i & if\ r \geq HMCR\ and\ f_{std} > 0.0001 \\ min\ x_i^j + U[0,1] \times \left(max\ x_i^j - min\ x_i^j\right) + U[-1,\ 1] \times BW_i & if\ r \geq HMCR\ and\ f_{std} \leq 0.0001 \end{cases}$$

$$BW_i = \begin{cases} \frac{BW_i^{Upper} - BW_i^{Lower}}{100} \times \left(1 - \frac{k}{NI}\right) & if\ f_{std} > 0.0001 \\ 0.0001 & otherwise \end{cases}$$

Mahdavi et al. proposed the I-HS, which increased the diversity of solution vectors at the beginning of the search and converges to the optimal solution through fine tuning as it goes to the end. As the number of iterations continues, the value of PAR decreases, and the fatigue of repeated experiments can be reduced compared to Original HS with fixed parameters.

$$PAR = PAR^{min} \times \frac{PAR^{max} - PAR^{min}}{NI} \times k$$

Kumar et al. and Mahdavi et al. proposed the PA-HS, which improves upon the I-HS that is characterized by the dynamic changes to the PAR and BW for each iteration. The PAR increases linearly and BW decreases exponentially within this range. Kumar et al. proposed that the HMCR and PAR be dynamically set and changed to linear and exponential forms through iterative calculations.

$$HMCR = HMCR^{min} \times \frac{HMCR^{max}/HMCR^{min}}{NI} \times k$$

$$PAR = PAR^{min} \times exp\left(\frac{\ln\left(PAR^{max}/PAR^{min}\right)}{NI}\right) \times k$$

5. Simulation

5.1. Data Set

This study verifies the proposed VM consolidation model by simulations with the actual PM and VM specifications. The PM consists of three heterogeneous PMs with different specifications, as shown in Table 2.

Table 2. Properties of PMs.

	Dell R515	**HP DL380 G8**	**HP DL585 G7**
# of Processors	2	2	4
# of Cores	6	8	12
Memory (GB)	16	32	64
Idle Power (W)	213	109	258
Peak Power (W)	420	276	396

The simulations use the actual workload of the DC, Bitbrains trace. Bitbrains is a service provider that specializes in managed hosting and business computation for enterprises, such as major banks, credit card operators, and insurers. The Bitbrains trace is a dataset that collects business-critical workloads generated while operating web, mail, and application servers, among others. The dataset used in this research consists of fastStorage and Rnd trace. The fastStorage trace is the workload of 1250 VMs, which are connected to fast storage area network (SAN) devices that is collected over a month. The Rnd trace is the workload of 500 VMs, which are connected to the fast SAN device or to the much slower network attached storage (NAS) devices that are collected over three months [44]. This study applies the 500 VMs of Rnd trace workload to the simulation, which has the higher management system ratio to fastStorage, and the characteristics of the dataset are shown in Table 3. The simulation of VM consolidation is based on 500 VMs, and each VM is randomly placed on a heterogeneous PM with the specifications shown in Table 2. The simulations were programmed in the C# language and carried out using a personal computer with an Intel i5-10500 CPU 3.10-GHz processor with 32 GB of RAM and a Windows 10 operating system.

Table 3. Properties of Bitbrains trace (Rnd).

Date	**# of VMs**	**CPU**		**Memory**	
		Mean (%)	**Std Dev. (%)**	**Mean (%)**	**Std Dev. (%)**
2013-7	500	4.893	9.076	10.807	13.505
2013-8	500	6.959	13.23	9.263	10.065
2013-9	500	5.732	11.234	9.937	11.553

5.2. Performance Measures

In addition to the usual performance measurements for VM consolidation, such as the reduction in energy consumption, number of migrations, and cost, SLA violation representing the quality of the DC operation environment is considered. Users want stable service from the cloud provider for a mission-critical system influenced heavily by the short delay rate of the service that the user is trying to provide. One of the most obligatory conditions for a cloud provider to provide a stable service environment to users is the stable operation of the DC. Beloglazov et al. proposed an equation for determining SLAV representing the number of SLA violations that occurred in Infra structure as a service (IaaS) environment. The SLAV uses the SLA violation time per active host (SLATAH) and performance degradation due to migrations (PDM), where SLATAH represents the percentage of overload occurring time during the PM operation, as shown in Equation (15). Overload is calculated on the basis of whether CPU utilization has exceeded the threshold. The PDM is calculated by dividing the estimated performance degradation C_{degr_j} by the total CPU capacity request for the VM C_{req_j}, and C_{degr_j} represents the performance degradation (10%) caused by VM migration as shown in Equation (16).

$$SLATAH = \frac{1}{P}\sum_{i=1}^{P}\frac{T_{over_i}}{T_{active_i}} \tag{15}$$

$$PDM = \frac{1}{V}\sum_{j=1}^{V}\frac{C_{degr_j}}{C_{req_j}} \tag{16}$$

$$SLAV = SLATAH \cdot PDM \tag{17}$$

5.3. HS Parameter Tuning Using Adaptive HS

An experiment was conducted to determine the optimal parameters for the HS for comparison with existing studies. We compared the optimal parameter results with the traditional HS and adaptive HS, which changes the parameters dynamically. The original HS used the parameters of the best and worst results as the basis. The Bitbrain dataset used in the simulation is the actual DC operation environment data without considering VM consolidation. The nature of this data showed that VM migrations occur more at the beginning than towards the end of the simulation. Comparative experiments to find the optimal parameters for the HS and adaptive HS compared VM placement results based on the PM–VM assignment at the beginning of the simulation, and the iterations were performed a total of 30 times for 10,000 times each.

For the original HS, the HMCR was set between 0.5 and 0.9 and PAR was set between 0.3 and 0.7. The parameter values with the best experimental result were 0.9 for the HMCR and 0.5 for the PAR. The parameter values with the worst experimental results were 0.5 for the HMCR and 0.6 for the PAR. Experimental results for each parameter combination; the high values of the HMCR showed better results than the low values. This can be interpreted as generating meaningful solutions (MC) referring to the previous solution obtaining better results than generating solutions randomly. However, the PAR showed a tendency for better results with values in the middle of the range, but not proportional to the parameter values. This can be interpreted as a non-excessive and appropriate neighborhood solution search that is effective for finding better solutions.

The results of the original and adaptive HS were then compared. For the adaptive HS, the worst and best values of the original HS were set as the upper/lower limits of the dynamically changing parameter from start to end of the experiment. The experimental results in Figure 7 and Table 4 show that each average value of the adaptive HS result was better than the worst result of the original HS, and the best result was the average value of the NS-HS. Based on the average values, the adaptive HS algorithms showing better results than the original were NS-HS and PSF-HS.

Based on the best results, all the adaptive HS algorithms showed better results than the original HS. Figure 8 shows the graph of the solution search process for each algorithm based on the worst result. From the point of view of hill climbing, which aims to create significant neighborhood solutions to obtain the global optimum, an algorithm can be evaluated with good quality using the numbers of the solution improvement frequencies. Adaptive HS seeks diversity through a wide solution search area at the beginning and reduces the search area to obtain the optimal solution. Experimental result show that the solution improved at the beginning of the algorithm, but the solution did not improve towards the end for the original HS. However, the adaptive HS showed that the solution improved even towards the end. Accordingly, the adaptive HS is an efficient algorithm to determine the optimal solution in terms of exploration and results.

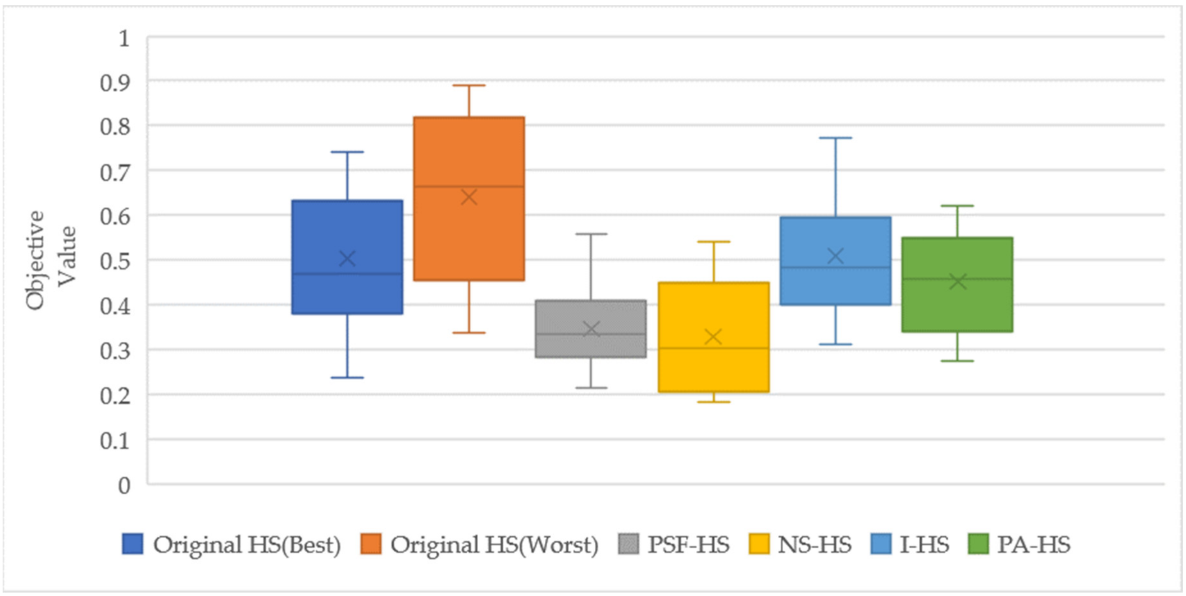


Figure 7. Comparative experimental results for original HS and adaptive HS.

Table 4. Comparative experimental results for HS and adaptive HS.

Algorithms	Mean	Worst	Best	Std Dev.
Original HS (Best)	0.503383	0.740271	0.236577	0.144973
Original HS (Worst)	0.640093	0.889	0.337712	0.180478
PSF-HS	0.345021	0.557234	0.21522	0.087343
NS-HS	0.328725	0.540743	0.181	0.125616
I-HS	0.509136	0.771547	0.312424	0.132324
PA-HS	0.451241	0.620795	0.274073	0.103505

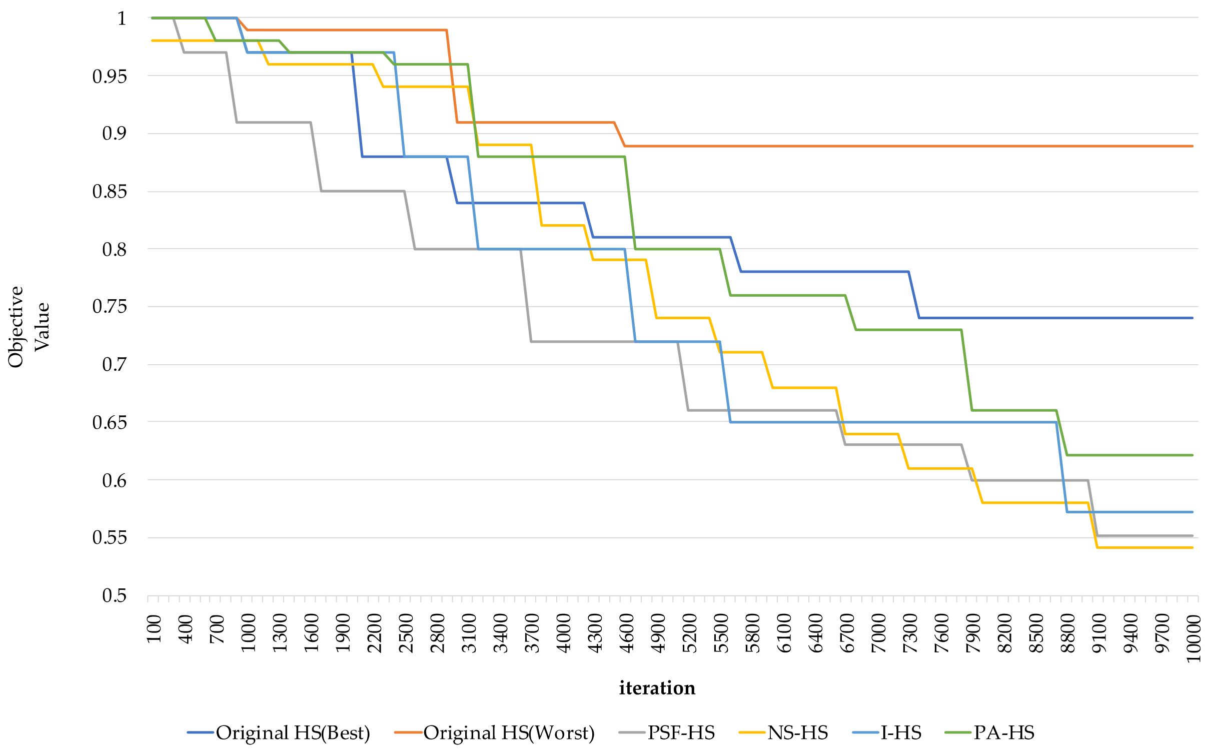


Figure 8. Comparison of improvements in HS and adaptive HS by iteration.

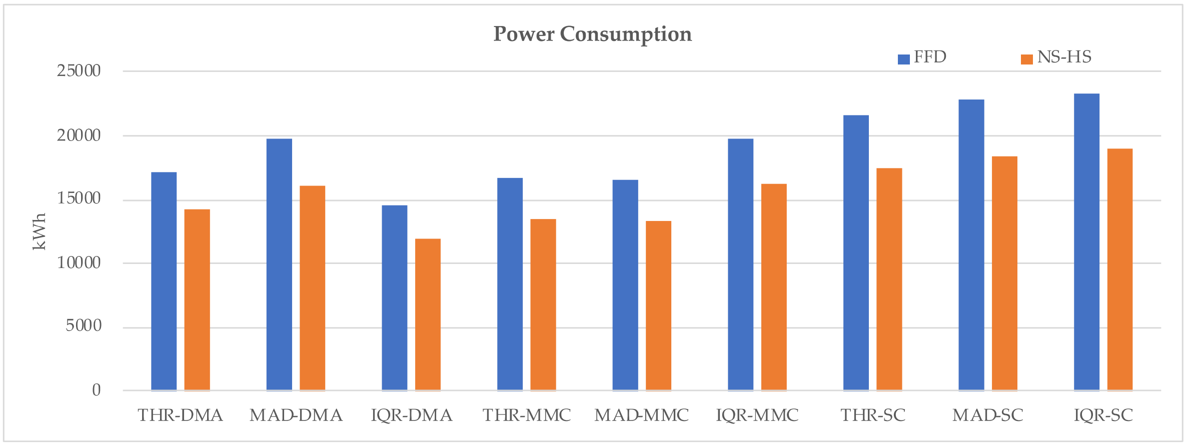
5.4. Experiment for SC Policy Performance Verification

This section presents the performance evaluation of the proposed SC policy for an effective algorithm for VM consolidation. The HS sets NS-HS as the parameters that show the best results in the previous experiment. A comparative analysis is conducted with the FFD, which was the most commonly applied heuristic algorithm in previous studies. The THR, MAD, and IQR policies are applied for VM consolidation as PM overload is detected during VM migration. The DMA, MMC, and SC policies proposed in this study are applied for to select the VMs to migrate during PM overload detection. The FFD and NS-HS are applied to determine the destination PM for the selected VMs for migration. Based on the above three policies of the source PM selection and three policies of VM selection, a total of nine combinations were compared and tested. The experimental results are shown in Figures 9a–c and 10a–c shows the graph representing the performance measure results for each algorithm combination.

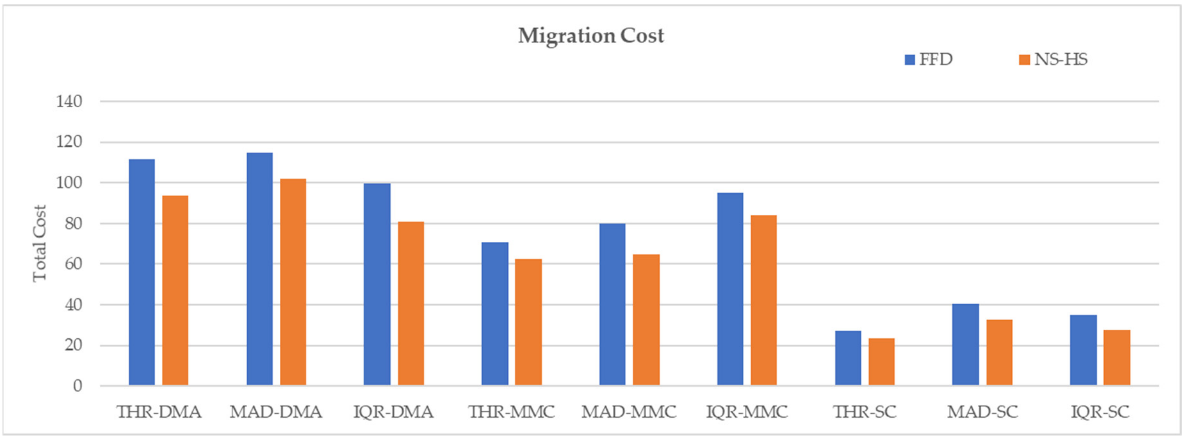
Figure 9a–c shows the results of comparing power consumption, VM migration cost, and VM migration count, which are the performance measures related to the objective function, for each combination. By applying VM consolidation, the power consumption was reduced in all combinations, and the power savings of the NS-HS were better than those of the FFD. The SC policy combination proposed in this study showed less improvement in terms of power savings compared to other VM selection combinations. However, it showed improvement in terms of VM migration cost and number of VM migrations. The THR-SC combination with the best migration cost showed about 77.15% improvement compared to the worst MAD-DMA, and number of migrations, which is the best THR-SC combination, showed about 53.15% improvement over the worst THR-DMA combination. The results of the NS-HS versus FFD were better in terms of performance measures associated with the overall objective function, and the combination with the SC policy was less effective in terms of power usage than the other combinations, but improved in terms of VM migration, which is applied as the penalty in VM consolidation.

Figure 10a–c shows the results of the SLATAH, PDM, and SLAV values, which are the performance measures related to the quality of the DC operating environment. Similarly, in the performance measure, the NS-HS produced better results than the FFD. When comparing the SC policy combinations and other combinations based on the NS-HS, SLATAH showed about 26.22% improvement over the DMA combination and about 30.03% improvement over the MMC combination. PDM also showed about 31.25% improvement over the DMA combination and about 33.33% improvement over the MMC combination; SLAV, which is the product of SLATAH and PDM, showed an improvement of about 33.33–62.25%. Through these results, it is confirmed from the experimental results that the VM consolidation model considering the SC combination contributes toward stable DC operation.

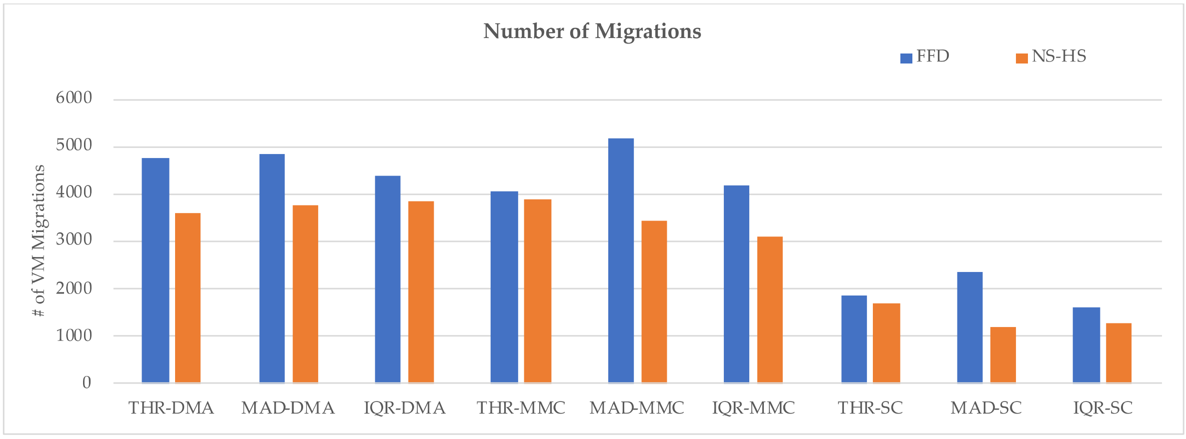
The proposed HS showed excellent results in every aspect of the objective value, such as power savings and migration cost, and also displayed good results for SLATAH and PDM, which are the aspects of SLAV, an indicator of DC operation quality. In particular, when the SC policy, which searches for a combination of VMs in which the workload of the PM is stably maintained, was applied, the power consumption was relatively high compared to other policies, but the effect of VM migration was relatively largely reduced. This showed excellent results in terms of the quality of the DC operation environment; it can thus be concluded that the proposed VM consolidation model meets the requirements of DCs that prioritize stability.



(**a**)

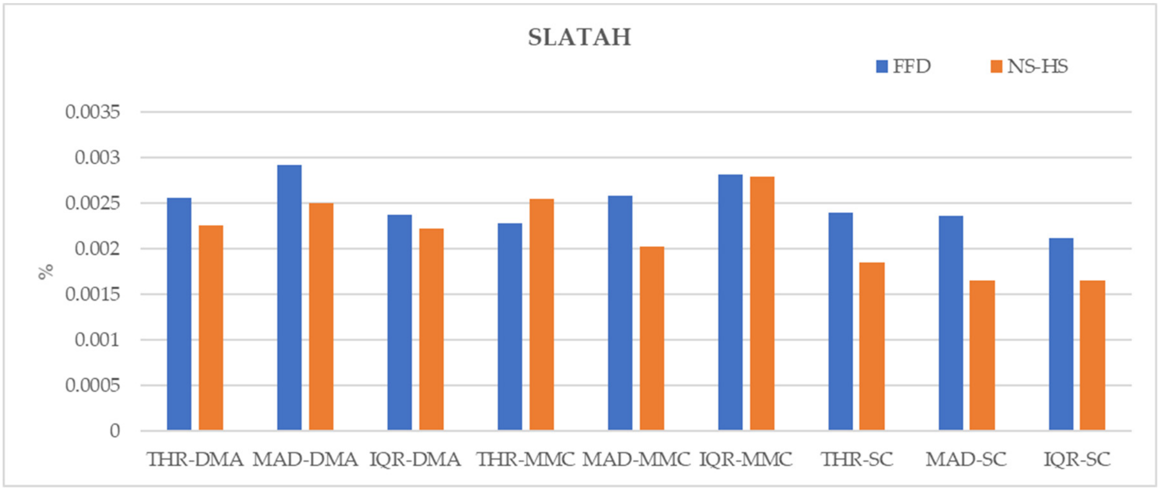


(**b**)

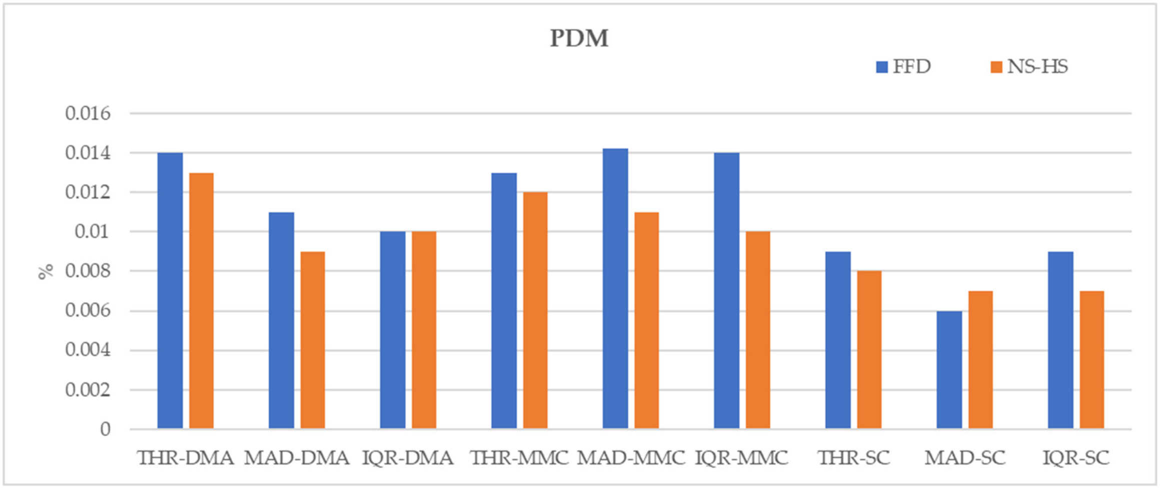


(**c**)

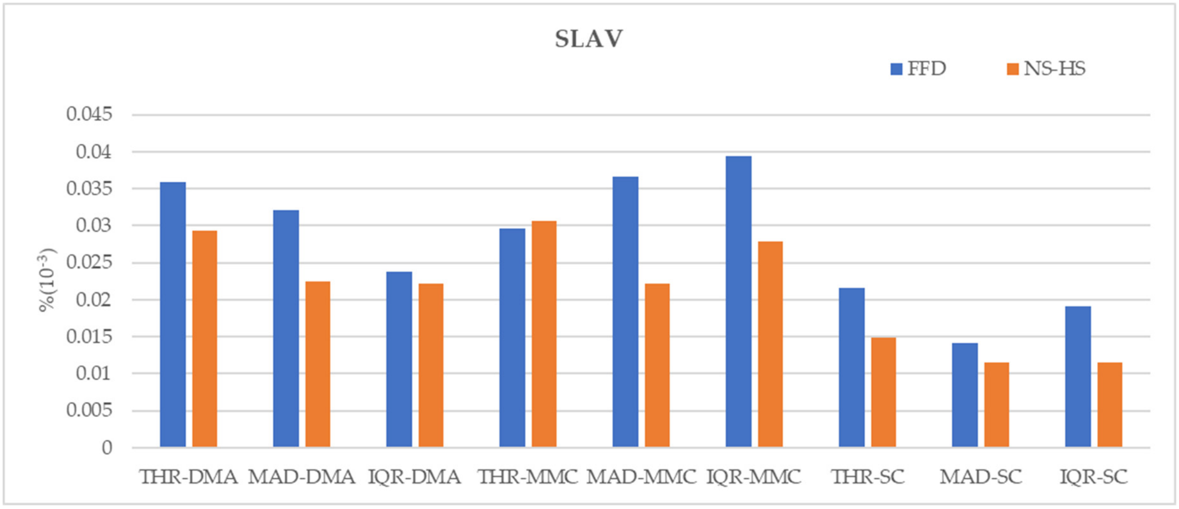
Figure 9. (**a**). Comparative experimental results of performance measures to the power consumption. (**b**). Comparative experimental results of performance measures to the migration cost. (**c**). Comparative experimental results of performance measures to the number of migrations.



(**a**)



(**b**)



(**c**)

Figure 10. (**a**). Comparative experimental results of the performance measures to the SLATAH. (**b**). Comparative experimental results of the performance measures to the PDM. (**c**). Comparative experimental results of the performance measures to the SLAV.

6. Conclusions and Future Study

In this study, we proposed a VM consolidation model to induce stable operation of a DC by exploring VM combinations, where the sum of the workloads remain stable, to reduce the power consumption of the DC. The HS scheme is designed to efficiently solve the proposed VM consolidation model, and the adaptive HS, where the parameters change dynamically, is applied. The adaptive HS shows better results than the original HS, and comparative analysis with the heuristic algorithm FFD shows that the proposed algorithm also has superior performance compared to the conventional approaches. In particular, when the SC policy, which searches for a combination of VMs in which the workload of the PM is stably maintained, showed excellent results in terms of the quality of the DC operation environment than other policies.

Benchmark data were used for the VM workload, and the VM workload was forecast to apply the proposed VM consolidation model to real world applications. In dynamic cloud computing environments, workload forecasting is an essential process for resource assignment, capacity planning, and energy savings. In a future study, an integrated VM consolidation model that combines the proposed VM consolidation model and a workload prediction subsystem will be explored. Through this, we intend to improve the limitations of this study, assuming the VM workload as a definite value.

The workload prediction subsystem in the integrated VM consolidation model would be expected to cluster similar VMs through behavioral analyses, and each cluster would be used as a unit of the learning model. The clustering forecast technique is aimed at maintaining unique patterns of VMs while increasing the management efficiency of the learning model with appropriate numbers of clusters. In the future study, the integrated VM consolidation model combining the proposed model in this study and a clustering forecasting technique that is appropriate for large-scale DC environments will be considered.

Till date, DCs have generally adopted conservative operation measures that prioritize environmental stability owing to uncertainties. As a large number of organizations are migrating to cloud-based computing environments, the scale of the DC, which has been characterized as "Bigger is Better" is expected to grow exponentially in the future. Therefore, intelligent cloud integrated management platforms utilizing advanced technologies, such as artificial intelligence and big data analysis, along with stable and efficient automatic operation strategies must be considered so that future researches will be aimed at these goals.

Author Contributions: H.Y.Y. developed the model and wrote the majority of the manuscript. S.H.J. performed the experiments and analyzed the data. K.S.K. developed the overall idea and the basic outline of the paper. All authors have read and agreed to the published version of the manuscript.

Funding: This work has supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. NRF-2019R1F1A1059276).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: http://gwa.ewi.tudelft.nl/datasets/gwa-t-12-bitbrains.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. ICT Gartner Estimates. *Industry Accounts for 2 Percent of Global CO_2 Emissions*; Press Release: Stamford, CT, USA, 2007.
2. CISCO. *Cisco Global Cloud Index: Forecast and Methodology, 2016–2021*; CISCO: San Jose, CA, USA, 2018.
3. Emersion Network Power, Understanding the Cost of Data Center Downtime: An Analysis of the Financial Impact on Infrastructure Vulnerability, Columbus, Ohio, USA. 2011. Available online: https://www.anixter.com/content/dam/Suppliers/Liebert/White%20Paper/Downtime%20-%20data-center-uptime_24661-R05-11.pdf (accessed on 14 January 2021).
4. ISO/IEC 30134-2. *Information Technology-Data Centres-Key Performance Indicators—Part 2: Power Usage Effectiveness (PUE)*; ISO/IEC: Geneva, Switzerland, 2015; p. 11. Available online: https://www.iso.org/standard/63451.html (accessed on 14 January 2021).
5. Google: Data Centers. Available online: https://www.google.com/about/datacenters/location (accessed on 1 September 2019).

6. Microsoft: Azure Global Infrastructure. Available online: https://azure.microsoft.com/en-us/global-infrastructure/regions/ (accessed on 1 September 2019).
7. Renugadevi, T.; Geetha, K.; Muthukumar, K.; Geem, Z.W. Optimized Energy Cost and Carbon Emission-Aware Virtual Machine Allocation in Sustainable Data Centers. *Sustainability* **2020**, *12*, 6383. [CrossRef]
8. Beloglazov, A. Energy-Efficient Management of Virtual Machines in Data Centers for Cloud Computing. Ph.D. Thesis, The University of Melbourne, Melbourne, Australia, 2013.
9. Feller, E.; Morin, C.; Esnault, A. A case for fully decentralized dynamic VM consolidation in clouds. In Proceedings of the 4th IEEE International Conference on Cloud Computing Technology and Science Proceedings, Taipei, Taiwan, 3–6 December 2012; pp. 26–33.
10. Clark, C.; Fraser, K.; Hand, S.; Hansen, J.G.; Jul, E.; Limpach, C.; Pratt, I.; Warfield, A. Live migration of virtual machines. In Proceedings of the 2nd Conference on Symposium on Networked Systems Design & Implementation, Boston, MA, USA, 2–4 May 2005; Volume 2, pp. 273–286.
11. Zakarya, M.; Gillam, L. Managing energy, performance and cost in large scale heterogeneous datacenters using migrations. *Future Gener. Comput. Syst.* **2019**, *93*, 529–547. [CrossRef]
12. Kuno, Y.; Nii, K.; Yamaguchi, S. A Study on Performance of Processes in Migrating Virtual Machines. In Proceedings of the 2011 10th International Symposium on Autonomous Decentralized Systems, Institute of Electrical and Electronics Engineers (IEEE), Tokyo, Japan, 23–27 March 2011; pp. 567–572.
13. Murtazaev, A.; Oh, S. Sercon: Server consolidation algorithm using live migration of virtual machines for green computing. *IETE Tech. Rev.* **2011**, *28*, 212–231. [CrossRef]
14. Ghribi, C.; Hadji, M.; Zeghlache, D. Energy Efficient VM Scheduling for Cloud Data Centers: Exact Allocation and Migration Algorithms. In Proceedings of the 2013 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing, Institute of Elec-trical and Electronics Engineers (IEEE), Delft, The Netherlands, 13–16 May 2013; pp. 671–678.
15. Abdelsamea, A.; Hemayed, E.E.; Eldeeb, H.; Elazhary, H. Virtual machine consolidation challenges: A review. *Int. J. Innov. Appl. Stud.* **2014**, *8*, 1504.
16. Jangiti, S.; VS, S.S. Scalable and direct vector bin-packing heuristic based on residual resource ratios for virtual machine placement in cloud data centers. *Comput. Electr. Eng.* **2018**, *68*, 44–61. [CrossRef]
17. Kashyap, R.; Chaudhary, S.; Jat, P. Virtual machine migration for back-end mashup application deployed on openstack environment. In Proceedings of the 2014 International Conference on Parallel, Distributed and Grid Computing (PDGC), Solan, India, 11–13 December 2014; pp. 214–218.
18. 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]
19. Beloglazov, A.; Buyya, R. Openstack neat: A framework for dynamic and energy-efficient consolidation of virtual machines in openstack clouds. *Concurr. Comput. Pract. Exp.* **2015**, *27*, 1310–1333. [CrossRef]
20. Monil, M.; Rahman, R. VM consolidation approach based on heuristics, fuzzy logic, and migration control. *J. Cloud Comput.* **2016**, *5*, 8. [CrossRef]
21. Masoumzadeh, S.; Hlavacs, H. Integrating vm selection criteria in distributed dynamic VM consolidation using fuzzy q-learning. In Proceedings of the 9th International Conference on Network and Service Management (CNSM 2013), Zürich, Switzerland, 14–18 October 2013; pp. 332–338.
22. Zhan, Z.H.; Liu, X.F.; Gong, Y.J.; Zhang, J.; Chung, H.S.H.; Li, Y. Cloud computing resource scheduling and a survey of its evolutionary approaches. *ACM Comput. Surv. (CSUR)* **2015**, *47*, 63. [CrossRef]
23. Adamuthe, A.C.; Pandharpatte, R.M.; Thampi, G.T. Multiobjective virtual machine placement in cloud environment. In Proceedings of the 2013 International Conference on Cloud & Ubiquitous Computing & Emerging Technologies, Pune, India, 15–16 November 2013; pp. 8–13.
24. Wu, G.; Tang, M.; Tian, Y.C.; Li, W. Energy-efficient virtual machine placement in data centers by genetic algorithm. In Proceedings of the International Conference on Neural Information Processing, Doha, Qatar, 12–15 November 2012; pp. 315–323.
25. Mark, C.C.T.; Niyato, D.; Chen-Khong, T. Evolutionary optimal virtual machine placement and demand forecaster for cloud computing. In Proceedings of the 2011 IEEE International Conference on Advanced Information Networking and Applications, Biopolis, Singapore, 22–25 March 2011; pp. 348–355.
26. Gao, Y.; Guan, H.; Qi, Z.; Hou, Y.; Liu, L. A multi-objective ant colony system algorithm for virtual machine placement in cloud computing. *J. Comput. Syst. Sci.* **2013**, *79*, 1230–1242. [CrossRef]
27. 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.* **2014**, *8*, 187–198. [CrossRef]
28. Xiao, H.; Hu, Z.; Li, K. Multi-objective VM consolidation based on thresholds and ant colony system in cloud computing. *IEEE Access* **2019**, *7*, 53441–53453. [CrossRef]
29. Kansal, N.J.; Chana, I. Artificial bee colony based energy-aware resource utilization technique for cloud computing. *Concurr. Comput. Pract. Exp.* **2015**, *27*, 1207–1225. [CrossRef]
30. Ibrahim, A.; Noshy, M.; Ali, H.A.; Badawy, M. PAPSO: A Power-Aware VM Placement Technique Based on Particle Swarm Optimization. *IEEE Access* **2020**, *8*, 81747–81764. [CrossRef]

31. Haghighi, M.A.; 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]
32. Kim, M.; Hong, J.; Kim, W. An Efficient Representation Using Harmony Search for Solving the Virtual Machine Consolidation. *Sustainability* **2019**, *11*, 6030. [CrossRef]
33. Fathi, M.H.; Khanli, L.M. Consolidating VMs in green cloud computing using harmony search algorithm. In Proceedings of the 2018 International Conference on Internet and e-Business, Singapore, 25–27 April 2018; pp. 146–151.
34. Renugadevi, T.; Geetha, K.; Muthukumar, K.; Geem, Z.W. Energy-Efficient Resource Provisioning using Adaptive Harmony Search Algorithm for Compute-Intensive Workloads with Load Balancing in Datacenters. *Appl. Sci.* **2020**, *10*, 2323. [CrossRef]
35. Fan, X.; Weber, W.D.; Barroso, L.A. Power provisioning for a warehouse-sized computer. *ACM SIGARCH Comput. Archit. News* **2007**, *35*, 13–23. [CrossRef]
36. Wu, Q.; Ishikawa, F.; Zhu, Q.; Xia, Y. Energy and migration cost-aware dynamic virtual machine consolidation in heterogeneous cloud datacenters. *IEEE Trans. Serv. Comput.* **2016**, *12*, 550–563. [CrossRef]
37. Varasteh, A.; Goudarzi, M. Server consolidation techniques in virtualized data centers: A survey. *IEEE Syst. J.* **2015**, *11*, 772–783. [CrossRef]
38. Geem, Z.W.; Kim, J.H.; Loganathan, G.V. A new heuristic optimization algorithm: Harmony search. *Simulation* **2001**, *76*, 60–68. [CrossRef]
39. Ismaeel, S.; Miri, A. Multivariate time series ELM for cloud data centre workload prediction. In *International Conference on Human-Computer Interaction*; Springer: Cham, Switzerland, 2016; pp. 565–576.
40. Geem, Z.W. Parameter estimation of the nonlinear Muskingum model using parameter-setting-free harmony search. *J. Hydrol. Eng.* **2011**, *16*, 684–688. [CrossRef]
41. Luo, K. A novel self-adaptive harmony search algorithm. *J. Appl. Math.* **2013**, *2013*, 653749. [CrossRef]
42. Mahdavi, M.; Fesanghary, M.; Damangir, E. An improved harmony search algorithm for solving optimization problems. *Appl. Math. Comput.* **2007**, *188*, 1567–1579. [CrossRef]
43. Kumar, V.; Chhabra, J.K.; Kumar, D. Parameter adaptive harmony search algorithm for unimodal and multimodal optimization problems. *J. Comput. Sci.* **2014**, *5*, 144–155. [CrossRef]
44. Shen, S.; van Beek, V.; Iosup, A. Statistical characterization of business-critical workloads hosted in cloud datacenters. In Proceedings of the 2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, Shenzhen, China, 4–7 May 2015; pp. 465–474.