Energy-Aware Virtual Machine Allocation in DVFS-Enabled Cloud Data Centers

Javad Masoudi¹, Behnam Barzegar*, Homayun Motameni³

¹Department of Computer Engineering, Sari Branch, Islamic Azad University, Sari, Iran
²Department of Computer Engineering, Babol Branch, Islamic Azad University, Babol, Iran
³Javad.Masoudi@iausari.ac.ir, Barzegar@iauns.ac.ir, Motameni@iausari.ac.ir

Corresponding author: Behnam Barzegar (Barzegar@iauns.ac.ir)

ABSTRACT Energy management is considered the major concern in cloud computing, which supports the rapid growth of data centers and computing centers; therefore, energy and load balancing have become crucial issues in cloud data centers. To address this issue, the present paper proposed a two-phase energy-aware load balancing (EALB) scheduling algorithm using the virtual machine migration through the Particle Swarm Optimization (PSO) algorithm to be applicable to dynamic voltage frequency scaling-enabled cloud data centers, which is called EALBPSO. In the first phase, an objective function was employed to deactivate a large number of physical machines in order to reduce energy consumption. The main idea of the algorithm was to maximize load balancing in the second phase, in which the remaining virtual and physical machines were used as the PSO inputs, and an objective function was also defined to distribute the load appropriately among the physical machines. In addition, a dataset was developed to test different parameters and scenarios with the aim of assessing the effectiveness of the proposed EALBPSO algorithm in comparison with other algorithms already proposed in the literature for similar purposes. The experimental results demonstrated that the proposed algorithm was capable of saving up to 0.896%, 9.716%, and 10.8% energy compared with the MDPSO algorithm, Kumar et al.’s algorithm, and Dahsti et al.’s algorithm, respectively, and also it showed 5.91%, 16%, and 16.267% improvements for the number of virtual machines migrations, and 3.867%, 8.623%, and 6.953% improvements for the deviation of processors, all compared with their competitors stated above, respectively.

INDEX TERMS Green data center; DVFS-enabled; Virtual Machine Placement

I. INTRODUCTION

During the past few years, the computational world has evolved, and this evolution has resulted in the emergence of new computational paradigms such as cloud computing. Cloud computing systems have emerged as a popular computing paradigm for hosting large computing systems and delivering different sets of services for accessing the resources virtually through the Internet, and the users pay per demand. Virtualization technology is one of the fundamental components of cloud data centers, especially in case of
infrastructure-based services. The rapid increase in the users’ computational needs has led to enormous consumption of energy in green data centers. As the number and dimensions of cloud data centers have grown, the energy consumption problem has become one of the most important issues, which increases the energy costs of service providers, thereby playing a central role in increasing the emission of CO$_2$ and greenhouse gases polluting the environment [1], [2].

Therefore, many methods have been suggested in the literature to decrease the energy consumption rate. Researchers have recently been interested in solving this problem by decreasing the number of physical machines through virtual machine (VM) migration. In fact, the migration of virtual machines can have a significant effect on the reduction of energy consumption [4].

Distributed load balancing is of high importance in physical machines because maintaining the Service Level Agreement (SLA) is a major problem in servicing cloud clients. Load balancing is used as a technique for distributing workload fairly among computing resources in the cloud environment [5], [6]. It is an essential method used to achieve the optimal efficiency of resources, increasing throughput in the shortest response time and decreasing the task completion time in the cloud environment. Analyses have shown that cloud datacenter nodes typically become unbalanced during an operation process, which can reduce the efficiency level. Thus, load balancing is essential for maintaining the cloud efficient and effective.

The challenge is that energy consumption minimization and load balancing maximization are two contradictory and inconsistent goals. Workload should be distributed fairly among computing resources to maximize load balancing. This can increase energy consumption, whereas increasing load balancing has the potential to decrease the execution time of tasks. To effectively address this challenge, this paper introduces an innovative method that can decrease the energy consumption on the basis of the number of active physical machines. As a result, a method is proposed by considering both objectives: to distribute workload fairly in cloud datacenters while reducing the energy consumption. Since task scheduling on the processors of cloud datacenters is an NP problem with multiple contradictory and inconsistent objectives, a method is proposed in two phases using the Particle Swarm Optimization (PSO) algorithm. In the first phase, an objective function is proposed for PSO to deactivate a larger number of physical machines for energy consumption reduction. In the second phase, the remaining virtual and physical machines are used as the algorithm inputs and another objective function is proposed to distribute workload fairly among the physical machines. The output of this phase is used as the layout of the virtual machines on physical machines.

The contributions of this paper are as follow:

- A system is proposed for power-aware virtual machine allocation (EALBPSO) in cloud data centers.
- The paper proposes two novel fitness functions based on our defined term energy, which can effectively reduce the overall energy consumption and maximize the load balancing of processors in Standard Deviation.
- Based on an evolutionary algorithm, we introduce a comprehensive VM allocation approach that can efficiently converge to a VM-to-PM mapping with best possible energy efficiency.
- A novel optimization approach, called EALBPSO, is proposed for initial placement of virtual machines, which aims to lower power consumption and maximize the load balancing of the data center processors.

The rest of the paper is organized as follows. Section 2 reviews the related work. Section 3 introduces the system model. Section 4 describes the proposed algorithm and discusses the way it performs the defined tasks. Section 5 presents the experimental setup and reports the simulation results compared with other algorithms already presented in the literature. Finally, Section 6 presents the conclusion.

II. Related Work

Generally, in data centers, one of the challenging issues is
how to allocate energy resources in a most efficient way. Note that the NP-hardness of the VM allocation problem has been already proved in the literature [5], [8]. Accordingly, it can be inferred that an optimal arrangement is not achievable in a deterministic polynomial time.

Garg [1] proposed an innovative penalty-guided hybrid approach, called the PSO-GA algorithm, which was introduced for the purpose of solving the constrained optimization problems. In PSO-GA, first, PSO is implemented to find the problem solution; after that, the best solution found in the current population executes the GA operators (crossover, mutation, and elitism) for the aim of enhancing the exploration of the solution space for the next generation.

Zhou et al. [2] proposed a cloud computing energy consumption model considering the processor’s costs for the execution and transmission processes. With the use of the model’s results, a task scheduling optimization algorithm, called modified particle swarm optimization (M-PSO), was also designed applicable to handling the local optimum issues and accelerating the convergence process.

Singh Patwal et al. [3] proposed a new integrated optimization technique, called TVAC-PSO-MS, to effectively schedule the optimum generation of the hydro-thermal and pumped storage hydro-thermal systems that are incorporated with solar units. This approach makes use of TVAC-PSO to search for the initial solution; then, it implements the mutation strategies for the aim of enhancing the search capability of the model. The TVAC-PSO-MS technique applies the Cauchy, Gaussian, and Opposition-based mutation strategies consecutively to the local best solution achieved by TVAC-PSO. In this composition, Cauchy improves the exploration capacity through making available longer jumps and larger perturbation. Afterwards, the Gaussian strategy is implemented to keep a balance between the exploration and exploitation capabilities.

Kumar and Raza [4] proposed a VM scheduling strategy based on the PSO algorithm in a way to appropriately place VM in cloud IaaS. This strategy is essentially concentrated on effectively allocating VM to physical servers with the aim of minimizing the total resource wastage and the number of implemented servers. Some simulations were carried out in order to check how VMs are allocated to the servers and also to assess the performance quality of the proposed algorithm regarding its scalability. They made a comparison between the results obtained from their proposed algorithm and those of the First-Fit, Best-Fit, and Worst-Fit placement strategies. The experimental results confirmed the efficiency of the algorithm in performing the tasks defined.

Xu et al. [5] attempted to comprehensively review the literature in order to provide a deep insight into the ways to enhance the future potentials, to identify the existing challenges, and to analyze the algorithms currently used to allocate VMs to hosts in infrastructure clouds. Their study was particularly concentrated upon the issues related to load balancing. In addition, they examined the classification targeting load balancing algorithms applied to the VM placement in cloud data centers, and classified the investigated algorithms on the basis of their classification approaches.

A big challenge to the cloud computing platform is how to allocate VM. The main concern is the way to obtain the goal of load balancing of multi-dimensional resources. In [6], at the first step to solving this problem, a mathematical model was proposed in order to address the defined problem through leveraging the ACO technique. The authors also attempted to enhance ACO through suggesting the concept of load imbalance degree and proposing the PM selection expectation. The results of the experiments revealed that the proposed ACO-based algorithm outperformed the basic ACO algorithm as well as the other algorithms recently proposed in the literature.

Two key issues in cloud environments of both homogenous and heterogeneous servers are virtual machine placement (VMP) and energy efficiency. Liu et al. [7] used evolutionary computing to solve the VMP problem through minimizing
the number of active physical servers to schedule the used servers in a way to save energy as much as possible. They developed a novel approach on the basis of ACS in order to make use of the advantages of the ACS algorithm in solving combinatorial problems. This strategy was adopted to accomplish the VMP objectives. They combined their approach with the order exchange and migration (OEM) local search techniques to design OEMACS. The resultant model was found more successful than the traditionally-used heuristic and other evolutionary-based approaches, particularly in case of VMP with bottleneck resource features. In addition, OEMACS was shown helpful in saving energy and more effective utilization of various resources.

Keshanchi et al. [8] introduced an enhanced GA-based algorithm for the purpose of optimizing the task scheduling solutions. Considering the model checking techniques, the authors proposed a behavioral modeling approach in order to efficiently analyze the effectiveness of their algorithm. Next, they extracted the expected qualifications of the proposed algorithm in the form of Linear Temporal Logic (LTL) formulas.

An intelligent cloud workflow scheduling system was introduced by Jia et al. [9] from the users’ point of view with the aim of decreasing the expense of using the IaaS cloud services. Their study, as claimed by the authors, contributed to the body of knowledge in two ways: First, the improvement of the models of the applications and the computing resources. Their model considered the main memory impacts, which resulted in proposing an innovative method of estimating the execution time. This way, the model could be of higher practicality, which provided, in turn, a scheduling of higher accuracy level. The second contribution was designing the A-ACO model as a novel adaptive ACO variant with taking into consideration two valuable heuristic factors. Their experimental results revealed that the proposed method could outperform the rivals in terms of both success rate and total cost. In addition, the efficiency of their method was confirmed by the results of the run time analyses conducted in their study.

Yan et al. [10] proposed an innovative approach, called SOWO, on the basis of the discrete particle swarm optimization workload approach with the aim of minimizing the number of active physical machines in VMP. The obtained results confirmed the SOWO practicability and superiority. SOWO, in comparison to the Open Stack native scheduler, reduced around 50% of the physical machine consumption while increasing the memory use of physical machine by more than two times.

Khosravi et al. [11] proposed an innovative VMP algorithm with the aim of enhancing the environmental sustainability through considering the distributed data centers with various carbon footprint rates and PUEs. Based on the simulation results, their algorithm succeeded in decreasing energy consumption rate and CO$_2$ emission and, at the same time, it could maintain the same level of service quality in comparison with similar algorithms already proposed in the literature.

Dahsti et al. [12] proposed hieratical architecture to meet the needs of both providers and consumers in these technologies. In that research, a novel service was also designed in the PaaS layer in order to schedule the consumer tasks. From the providers’ perspective, if the physical machine specifications and the user’s requests are not compatible in cloud, some problems, e.g., high rate of energy consumption and energy-performance trade-off, may appear, which finally can result in a reduction in profits. The authors [12] modified the Particle Swarm Optimization (PSO) for the purpose of guaranteeing the service quality of the users’ tasks and enhancing the energy efficiency. The final objective of these strategies was the reallocation of the migrated virtual machines in the overloaded host. According to the simulation results in CloudSim, the simulation condition was near to the real environment.

Xiong et al. [13] proposed an energy-efficient VM allocation
algorithm on the basis of PSO and the energy efficient multi-resource allocation model in order to decrease the power consumption of CPU and disk in cloud data centers. They defined the fitness function of PSO as the total Euclidean distance that can be used to identify the optimum point between energy consumption and resource consumption. Ibrahim et al. [14] proposed a Power-Aware technique based on Particle Swarm Optimization, called PAPSO, for VM placement with migration to the best PMs with the aim of reducing energy consumption without violating SLA. Reddy et al. [15] proposed the modified discrete particle swarm optimization (MDPSO) considering PSO for the purpose of VM initial placement. In addition, they introduced a new VM selection algorithm in order to optimize the current allocation on the basis of memory usage, bandwidth utilization, and the VM size in order to minimize the operational costs and negative environmental impacts and also to guarantee the service-level agreements for the services delivered by data centers. Zhang et al. [16] proposed an innovative efficient evolutionary resource reservation-based service for the VM allocation. Their model was confirmed capable of maximizing the energy efficiency of a cloud data center while integrating more reserved VMs. Their study aimed to accurately estimate the energy consumption rate. For this purpose, there is a need for simulating all the VM allocation updates, which requires much time when executed by means of conventional cloud simulators. To cope with this problem, a simplified simulation engine was designed for CloudSim, which could increase the speed of the evolutionary approach process. Soltanshahi et al. [17] proposed a novel green solution to allocate VM to physical hosts in a way to decrease the operating costs and CO2 emission rate in cloud data centers with the use of the Krill Herd algorithm. Due to the quick growth of cloud computing systems, more and more data centers are required, which results in the surge of energy consumption in cloud data centers. Even more significant challenges have been posed on energy consumption due to the advent of cloud computing. Fu et al. [18] proposed the binary PSO (BPSO) algorithm with the aim of designing VMP strategies in a way to decrease the power consumption that was measured using an energy efficiency fitness function proposed in that study. It requires multiple iterations and updates for the VMP process. Ebadifard et al. [19] proposed a static task scheduling method on the basis of PSO, where the assumption was that the tasks are non-preemptive and independent. Their method was found successful in the enhancement of the basic PSO performance with the application of a load balancing technique to VMs. Task-resource assignments are similar to the position vectors of particle. At first, the position vectors are produced in a random way; afterwards, at each iteration, these vectors are enhanced with the use of a load-balancing technique. This technique can reduce makespan and, at the same time, increase the resource utilization and improve the performance by using the benefits of appropriate fitness functions. Rajasree and Beegom [20] proposed a discrete version of PSO, called Integer-PSO. It is applicable to task scheduling within cloud computing environments. It is also able to optimize both single-objective and multiple-objective functions. To find the optimum schedule in a reasonable time, it is necessary to take into account several factors, including energy, cost, execution time, profit, and other QoS parameters at the user side. Garg and Sharma [21] proposed the multi-objective reliability redundancy allocation problem of a series system. In this problem, providing a reliable system with saving no designing cost is taken into consideration as two different aims. Because of the non-stochastic, uncertain, and conflicting factors, it is not simple to decrease the system costs and, at the same time, to enhance the system’s reliability. Such conditions make the decision-making process so difficult, and due to the existence of several objectives at one time, the fuzzy multi-objective optimization
problem (FMOOP) comes to exist, which necessitates the Pareto optimal solutions instead of a single optimal solution.

In the study of Khosravi et al. [22], the problem under investigation was how to determine the amount of energy cost that we could save if we are aware of the future levels of renewable forms of energy (solar/wind), which are accessible in data centers. This study was concentrated on introducing algorithms with entire and partial knowledge about the future accessibility of renewable energy levels with the aim of migrating VMs to another data center in a certain region where there is not adequate renewable energy resource at the host data center. At the first step, they introduced the optimal offline algorithm with the objective of minimizing the energy costs. As it is necessary to be aware of the future levels of renewable energy sources to have an optimum offline algorithm, they attempted to design two online algorithms, i.e., a deterministic algorithm and a future-aware online algorithm. The former has no knowledge about the future levels of renewable energy sources and the latter has only a limited knowledge about the future levels of these sources.

Alharbi et al. [23] proposed a technique based on mathematical modeling and heuristics method named EEVM-CF for VM placement services from the cloud computing to the fog computing with the aim of reducing energy consumption. Table 1 presents a brief description of the relevant literature.

### III. Proposed System Model

This section proposes the formal models of the proposed system architecture, energy consumption, migration overhead energy, as well as the constraints on the problem formulation.

For the better understanding of the systems model and the proposed algorithm, the various parameters used in this paper along with a brief description are summarized in Table 2.

### TABLE 1. Summary of the related work

| Reference                  | Scenario       | VM Allocation | VM Uniformity | Methodology                        | VM resource type          | Scheduling Objective                  |
|----------------------------|----------------|---------------|---------------|------------------------------------|---------------------------|---------------------------------------|
| Dinesh Reddy et al. [15]   | Private cloud  | Dynamic       | Heterogeneous | Modified Discrete PSO               | CPU                       | Min number of migrations              |
|                            |                |               |               |                                    |                           | Min number of migrations              |
|                            |                |               |               |                                    |                           | Min Energy Consumption                |
| Soltanshahi et al. [17]    | Hybrid cloud   | Dynamic       | Heterogeneous | Krill herd algorithm               | CPU                       | Min energy consumption                |
|                            |                |               |               |                                    |                           | Min number of SLA violations          |
| Song et al. [24]           | Public cloud   | Dynamic       | Homogeneous   | HLA platform                       | CPU                       | Min migration latency                 |
| Cho et al. [25]            | Private cloud  | Dynamic       | Heterogeneous | Ant Colony Optimization & Particle Swarm Optimization | Multiple                  | Min number of migrations              |
| Thiruvenkadam et al. [26]  | Private cloud  | Dynamic       | Heterogeneous | Genetic algorithm                  | Multiple                  | Min number of migrations              |
| Verma et al. [27]          | Private cloud  | Dynamic       | Heterogeneous | Best-Fit Decreasing Heuristic       | CPU                       | Min virtual machine down time         |
| Wang et al. [28]           | Hybrid cloud   | Dynamic       | Heterogeneous | Level genetic algorithm            | CPU & DISK                | Min Energy Consumption                |
|                           |                |               |               |                                    |                           | Min Network Load                      |
| Ramezani et al. [29]       | Hybrid cloud   | Static        | Homogeneous   | Particle Swarm Optimization        | CPU                       | Min task execution time               |
|                           |                |               |               |                                    |                           | Min task transfer time                |

### TABLE 2. Definition of notations

| Description | Notations | Description |
|-------------|-----------|-------------|
| Energy consumption of a physical machine | \( P(E) \) | \( b \) Network bandwidth |
| Power consumption at t | \( P_{\text{Proc}}(t) \) | \( v_{i}^{\text{par}} \) Transferred memory within the \((i, j)\)th interval |
| Maximum power consumption | \( P_{\text{max}} \) | \( \nu_{i} \) Velocity of particle \( i \) at iteration \( t \) |
| Processor usage rate | \( u(t) \) | \( V_{i}(t + 1) \) Velocity of particle \( i \) at iteration \( t+1 \) |
| No. of virtual machines | \( N \) | \( X_{i}(t + 1) \) Current position of particle \( i \) at iteration \( t+1 \) |
| No. of physical machines | \( N_{\text{PM}} \) | \( X_{i}^{\text{pos}} \) Position vector of particle \( i \) |
| Total energy consumed by the ith VM | \( E^{\text{tot}}(i) \) | \( g^{\text{pos}} \) Position vector of the best particle in the population |
| Maximum CPU usage required by the ith VM | \( P_{\text{cpu}}(i) \) | \( w \) Inertia weight |
| Maximum memory required by the ith VM | \( P_{\text{mem}}(i) \) | \( c_{1} \) & \( c_{2} \) Acceleration coefficient |
| Memory capacity of the jth physical machine | \( P_{\text{cap}}(j) \) | \( r_{1}(j) \) & \( r_{2}(j) \) Random number between 0 and 1 |
| Constant parameters | \( a \) & \( b \) | \( U_{\text{pm}} \) Processor usage percentage in the 4th PM |
| Amount of memory transferred during the VM migration | \( V_{\text{mem}} \) | \( U_{\text{m}} \) Memory usage percentage in the 4th PM |
| Time spent calculating the memory transferred in the 4th interval | \( V_{\text{mem}}(i) \) | \( u_{\text{ave}} \) Memory usage percentage in the active PM |
| The ith VM | \( \text{VM}(i) \) | \( U_{\text{ave}} \) Average memory usage percentage in active PM |
| The ith task | \( T(i) \) | \( \text{Task Generator} \) Generating and distributing tasks among the existing VMs |
| The task queue of the ith VM | \( Q(i) \) | \( \text{Hortli} \) The ith physical machine |
A. Proposed System Architecture
The high-level architecture of the proposed energy-aware VM allocation and migration (EALBPSO) in cloud data centers is displayed in Fig. 1. This subsection analyzes the system framework in which the proposed system was designed. In the proposed method, datacenters of \( m \) physical machines with different processing features are selected. Every physical machine has a processor with multiple kernels; the performance of each kernel is defined in Millions Instructions per Second (MIPS). A physical machine is characterized by its memory and processing power. Using the datacenter user interface, users send their requests to the system for the development of a VM with required resources. Based on the resources demanded, the VM is allocated to the first available physical machine. In a specific time, datacenters administrators state the need for decreasing the energy consumed by the datacenter through migrating VMs from idle physical machines and turning off passive machines. However, it is necessary to consider the overhead cost of transferring VMs to decrease the total energy consumption. Given the remaining physical machines, the load distribution process is then applied to these machines.

B. Energy Consumption Model
In physical machines, energy consumption depends on the processor and memory usage rates. In fact, a processor is the major energy consumption factor in these machines. Accordingly, using the resources of a physical machine is based on the processor usage rate. Given the fact that the use of a processor depends on the existing workload on a physical machine, the VM energy consumption model can be determined using Equation (1) based on the processor usage rate:

\[
E = \int_{t_1}^{t_2} P(u(t)) \times dt
\]

where \( E \) represents the energy consumption of a physical machine within \([t_1, t_2]\), and \( P(u(t)) \) shows the power consumption at \( t \) and is calculated using Equation (2):

\[
P(u(t)) = c \times P_{max} + (1 - c) \times P_{max} \times u(t)
\]

where \( P_{max} \) indicates the amount of power consumed when 100% of the physical machine is exploited, \( c \) denotes the percentage of energy consumed by an idle physical machine, and \( u(t) \) indicates the processor usage rate \((u(t) \in [0,1]))\) that is determined through the workload resulting from the execution of a cloud service on a physical machine.

C. Problem definition
In the cloud computing architecture, during an operation, VMs can migrate from one machine to another. Generally, there can be several host machines for a VM to migrate to. At the same time, VM migration can change the machine energy consumption. Thus, it is necessary to correctly place and arrange VMs on host machines in energy consumption optimization and to switch off the unused machines. It is assumed that \( n \) VMs should be executed on \( m \) host machines. In fact, the problem is to find a model for the migration of VMs on host machines. In addition to reducing energy consumption, this model should be able to transfer a
larger number of VMs. The total energy consumption of a
datacenter can be obtained from Equation (3), as follows:

\[
E_t = \min \sum_{i=1}^{n} \sum_{j=1}^{m} E_i \times y_{ij} \\
\text{S.t.} \\
\sum_{i=1}^{n} r_i^{cpu} \times y_{ij} < c_j^{cpu} \\
\text{and} \\
\sum_{i=1}^{n} r_i^{mem} \times y_{ij} < c_j^{mem}
\]

(4)

\[
\sum_{j=1}^{m} r_{ij} = 1 \quad i = 1.2.3 \ldots \ldots n
\]

(5)

where \( m \) denotes the number of physical machines in the
datacenter, \( E_i \) indicates the total energy consumed by the \( i \)th VM in an interval, \( r_i^{cpu} \) refers to the maximum CPU usage required by the \( i \)th VM, \( r_i^{mem} \) shows the maximum memory required by the \( i \)th VM, \( c_j^{cpu} \) indicates the CPU capacity of the \( j \)th physical machine, and \( c_j^{mem} \) refers to the memory capacity of the \( j \)th physical machine.

Equation (4) shows that the set of resources required by VMs on a physical machine should be exceeded by the capacity of that physical machine.

According to Equation (5), a VM can only be executed on one physical machine. Since physical machines are not homogenous, \( c_j^{cpu} \) and \( c_k^{cpu} \) are not equal (\( j \) and \( k \) refer to the physical machines of the datacenter).

D. Migration Overhead Energy

The migration of VMs can result in a considerable energy overhead. A linear model was proposed in this paper to show the migration energy consumption, which depends on the copied memory capacity of the VM from a physical host in the source to a physical host in the destination. Equation (6) calculates the energy consumed during the migration process:

\[
E_{mig} = \alpha \times V_{mig}\beta
\]

(6)

Where \( \alpha \) and \( \beta \) are constant parameters, and \( V_{mig} \) signifies the amount of memory that should be transferred during the VM migration.

In case of migration of virtual machines, we set the value of parameter \( \beta \) as one and otherwise zero. Also, considering the environment that may be homogeneous or heterogeneous, The \( \alpha \) parameter is set to one for a heterogeneous environment and 0.8 for homogeneous environments.

The VM migration is performed through several steps. First, the entire RAM content is transferred from a physical host in the source to a physical host in the destination. After that, the RAM pages, which are rewritten and changed by the host during migration, are sent. The number of these pages depends on the VM memory usage model and is measured by the page dirty rate (\( d \)). If every page size is assumed \( l \) bytes with the previous interval assumed to be \( t_{i-1} \), the time spent calculating the memory transferred in the \( i \)th interval (\( v_{mig,i} \)) can be calculated using Equation (7):

\[
v_{mig,i} = \begin{cases} v_{mem} & i = 0 \\ d \times l \times t_{i-1} & i \neq 0 \end{cases}
\]

(7)

where \( t_{i-1} \) can be obtained by Equation (8):

\[
t_{i-1} = \frac{v_{mig,i-1}}{b}
\]

(8)

where \( b \) shows the network bandwidth, and \( v_{mig,i-1} \) indicates the transferred memory within the \((i-1)\)th interval. The entire memory consists of the summation of pieces of memory transferred in each interval. The final step is called the termination-copy phase in which it is impossible to make the page dirty and to write in the memory so that the final pages could be copied. As a result, the total memory transferred for one VM can be obtained by Equation (9):

\[
v_{mig} = \sum_{i=1}^{n+1} v_{mig,i}
\]

(9)

Where \( n+1 \) denotes the number of steps required to transfer the memory of a VM from a physical machine in the source to another in the destination. In fact, the extra 1 indicates the final step in the termination-copy phase.
a cloud datacenter. This section presents the details of the EALBPSO-DVFS algorithm proposed in this paper to schedule the dependent tasks with the aim of decreasing energy consumption and increasing load balancing. Considering the efficiency and energy models presented in Section 3, the impacts of the proposed method with the expected objectives could be discussed. The proposed algorithm involves two phases each of which is analyzed in the following subsections.

A. First Phase: The Use of the Particle Swarm Optimization (PSO) Algorithm for Energy Consumption Reduction

In the first phase, a method is proposed to minimize the total energy consumption and the migration energy overhead and to estimate the memory and processor usage rates required by every VM. Therefore, a heuristic algorithm is employed to provide a solution to the problem of aggregating energy-aware VMs. The overall flowchart of the proposed method is shown in Fig 2.

As a result of implementing the above-noted steps, the output of the algorithm’s first phase (which includes the method of distributing VMs on physical machines) will be applied as the input to the second phase. However, physical machines are active in this phase only if their names are mentioned in the first phase output.

1) CALCULATING THE POSITION AND VELOCITY OF EACH PARTICLE

The position of a particle in iteration $t$ is calculated as a vector calculated by Equation (10):

$$X_i^t = (x_{i1}^t, x_{i2}^t, \ldots, x_{iD}^t) \in \{1.2.3.\ldots.N\} \quad (10)$$

Moreover, each particle has a velocity shown by a vector obtained from Equation (11) in iteration $t$:

$$V_i^t = (v_{i1}^t, v_{i2}^t, \ldots, v_{iD}^t) \quad (11)$$

In each iteration, every particle has a memory of its best previous position, which is shown by vector $L$ obtained by Equation (12):

$$L_i^t = (l_{i1}^t, l_{i2}^t, \ldots, l_{iD}^t) \quad (12)$$

The best current position experienced by all particles of a swarm is shown by vector $G$ in Equation (13):

$$G^t = (g_1^t, g_2^t, \ldots, g_D^t) \quad (13)$$

The position and velocity of each member are updated using Equations (14) and (15), respectively:

$$V_i(t + 1) = wV_i(t) + c_1r_1(t)(L_i(t) - X_i(t)) + c_2r_2(t)(G(t) - X_i(t)) \quad (14)$$

$$X_i(t + 1) = X_i(t) + V_i(t + 1) \quad (15)$$

In Equations (10)-(15), $t$ denotes the iteration number, $C_1$ and $C_2$ refer to the learning factors that control the particle movement, $r_1$ and $r_2$ are two random numbers within $[0, 1]$, and $w$ is the algebraic weight initialized within $[0, 1]$.

2) MAPPING THE VM MIGRATION PROBLEM ON PSO

This subsection analyzes the details of mapping the VM aggregation problem onto the PSO algorithm. In this problem, the datacenter consists of $m$ physical machines on which $n$ VMs are executed. VMs should be aggregated to minimize the datacenter energy consumption. For instance,
if there are 4 physical machines and 3 VMs, the arrangement is (1, 3, 2) in which the first VM is placed on the first physical machine, whereas the second VM is placed on the third physical machine, and the third VM is placed on the second physical machine. The goal is to install VMs on physical machines in a way to minimize the energy consumption based on Equation (1). A solution or a particle is regarded as a model for the distribution of n VMs on m physical machines. A solution is defined as an N-dimensional vector, and each dimension shows to which physical machine a VM was assigned by Equation 16.

Solution(i) = {k1, k2, k3,...kN} 1 ≤ k_i ≤ M and i ∈ [1,N]  (16)

3) POSITION OF EVERY PARTICLE

The position of every particle is defined through the distribution vector of N VMs on M physical machines. In fact, if the position of a particle is shown as (1, 2, 3,…, k), it means that the first, second, third, and kth physical machines, respectively.

4) OBJECTIVE FUNCTION OF EACH PARTICLE

Since every particle is defined as an array, each dimension of a particle shows a physical machine; thus, the objective function equals the total energy consumption of machines existing in each dimension of the particle. The energy consumption of every particle can be obtained using Equation (17):

\[ \text{fitness}(X_i) = \sum_{k=1}^{N} E_k \quad k = x^r_{ij} \cdot x^r_{ij} \in X^r_i \]  (17)

5) CALCULATING VELOCITY VECTOR

The velocity vector was defined to result in a new position vector when the position vector of a particle is applied at each step. In fact, the velocity V of each particle is defined as a sequence of “movement” of the position components of that particle, which is calculated by Equation 18:

\[ V = ((i_k, j_k)) \quad i_k, j_k \in \{1,...,N\} \quad k = 1,...,||V|| \]  (18)

\[ V = ((i_1, j_1), (i_2, j_2), ..., (i_{||V||}, j_{||V||})) \]  (18)

where \( ||V|| \) signifies the sequence length.

6) CALCULATING THE NEW POSITION OF A PARTICLE

When the velocity v is applied to a position vector, the i1 and j1 components are first exchanged with each other in the position vector. After that, the i_2 and j_2 components are substituted. The same procedure is followed for i_3 and j_3 until \( [i]_1, [i]_{||V||} \) and \( [j]_{||V||} \). For instance, if \( p = (2, 1, 4, 3, 5, 6) \) and \( v = ((1, 2), (2, 3)) \) refer to a position vector and a velocity vector, respectively, then the following positions are obtained by applying v to p:

(1, 2, 4, 3, 5, 6): substitution of 1 with 2
(1, 4, 2, 3, 5, 6): substitution of 2 with 3

The algorithm for the allocation of VMs using our EALBPSO is illustrated in Algorithm 1.
\[ V_i(t + 1) = wV_i(t) + c_1r_1(t)(L_i(t) - X_i(t)) + c_2r_2(t)(G(t) - X_i(t)) \]
\[ X_i(t + 1) = X_i(t) + V_i(t + 1) \]

7. If the stop condition is met, terminate the iteration process of the algorithm, and output the approximate optimal solution and related information. If not, repeat Steps 4 to 7 until the stop condition is met.

### B. Second Phase of the Proposed Method

In this subsection, the second phase of the proposed method is analyzed. The first phase output was used as the input of the second phase. In the first phase, the number of VMs was reduced to decrease the energy consumption. The second phase aimed to place VMs on the remaining physical machines in a way to reduce the energy consumption. In this phase, the PSO algorithm was used; however, the objective function was designed for load distribution.

1) **OBJECTIVE FUNCTION**

In the second phase, the objective function is based on load distribution as expressed in Equation (19):

\[
F = \sum_{k=1}^{P_{\text{new}}} (U_{ck} - U_c)^2 + (U_{bk} - U_b)^2
\]

where:
- \(U_{ck}\): The processor usage percentage in the \(k\)th physical machine
- \(U_{bk}\): The memory usage percentage in the \(k\)th physical machine
- \(U_c\): The average processor usage percentage in the active physical machines
- \(U_b\): The average memory usage percentage in the active physical machines

This function calculates the variance of processors used for each distribution model. The small value of this function indicates that the distribution of VMs was performed better with a higher value of distribution model.

### V. Performance evaluation with simulation

This section analyzes the results of simulating the proposed method and introduces the simulation environment. A few parameters are also presented to evaluate the proposed method. Finally, the methods are compared in the defined scenarios.

#### A. Simulation Steps

The simulations executed in the present study are performed on CloudSim toolkit [28] (based on Java) that provides a proper environment for modelling and simulating the energy-aware computational resources in large-scale cloud computing datacenters. For the purpose of this study, CloudSim was installed on an Asus Notebook with Intel core i7-A540UP CPU 2.4 GHz with 8 cores and the memory of 4 GB. For the simulation process, five datacenters were created and 200 VMs were set. In CloudSim, each VM component has some features regarding VM: processor (MIPS), accessible memory (RAM), bandwidth, storage size, and the VM's internal provisioning policy. Each component involves three processor types, i.e., AMD Opteron 2218, AMD Turion 64 MT-34, and Intel core i3-540, respectively [1], all well provided with the DVFS technology. Table 3 describes the three processor types in detail.

| Processors          | AMD Opteron 2218 | AMD Turion MT-34 | Intel Core i3-540 |
|---------------------|------------------|------------------|-------------------|
| Voltage (V)         | 1.1, 1.15, 1.15, 1.20, 1.25, 1.30 | 0.9, 1.0, 1.05, 1.1, 1.15, 1.2 | 1.125, 1.125, 1.2, 1.2, 1.3, 1.3, 1.375 |
| Frequency (GHz)     | 1.0, 1.8, 2.0, 2.2, 2.4, 2.6 | 0.8, 1.0, 1.2, 1.4, 1.6, 1.8 | 3.07, 3.2, 3.4, 3.6, 3.8, 4.0, 4.2 |
| Highest power (W)   | 95               | 25               | 108               |
| Lowest power (W)    | 26.16            | 6.25             | 53                |

In the simulation scenario of the proposed method, a few VMs were first assigned randomly to each of the physical machines. Each VM, based on the service type, made a request for an amount of memory and a percentage of
processor. The task generator unit was employed to analyze the arrangement of VMs and the placement algorithm performance. VMs were analyzed in performance through load generation. The task generator unit generated the tasks at a random rate and put them in the queue of each VM. After that, VMs received the tasks from their input queues in order and processed them. Figure 3 shows a schematic view of the simulation method, and Table 4 presents the concepts used in this phase.

Table 4: Definition of Notations

| Notation | Definition |
|----------|------------|
| Host(i)  | The i<sup>th</sup> physical machine |
| VM(i)    | The i<sup>th</sup> VM |
| T(i)     | The i<sup>th</sup> task |
| Q(i)     | The task queue of the i<sup>th</sup> VM |
| Task Generator | Generating and distributing tasks among the existing VMs |

B. Simulation Parameters

- Migration Rate: This parameter shows the number of VMs transferred at the time of placement.
- Energy Consumption: This parameter indicates the energy consumption of physical machines after the transfer of VMs and deactivation of the idle machines.
- Load Distribution: This parameter shows the standard deviation of the processors used in all active physical machines. It is calculated as a new criterion to measure the load distribution on physical machines.

C. Simulation Scenarios

For the efficiency analysis, the results obtained by the proposed method (EALBPSO: Energy-Aware Load Balancing Particle Swarm Optimization on processors equipped with DVFS) were compared with those of three other methods, i.e., the MDPSO algorithm [5], Kumar et al.’s [10] algorithm, and Dahsti et al.’s [12] algorithm. Different scenarios were developed to compare the proposed method with the approaches proposed by the MDPSO [5], Kumar et al. [10], and Dahsti et al.’s [12] algorithms. Accordingly, the parameters presented in Section 4.2 are measured.
• First Scenario: Changing the Number of VMs

The first scenario compares the effects of the number of VMs on the proposed method and those of the other three algorithms noted above. Table 5 shows the parameters used in this scenario.

| Parameter                              | Value          |
|----------------------------------------|----------------|
| Number of physical machines            | 20             |
| Minimum number of VMs on a physical machine | 2             |
| PSO iterations                         | 100            |
| Processing power and capacity of physical machines | Heterogeneous |
| Total number of VMs                    | Variable (50-300) |

Table 6 compares the total amounts of energy consumed by physical machines. In fact, the physical machines with no VMs will be deactivated; therefore, the larger the number of deactivated physical machines, the lower the energy consumption of a datacenter. The proposed method outperformed the other approaches in terms of placing VMs. Therefore, a larger number of physical machines were deactivated, hence decreasing the energy consumption.
in the proposed method as opposed to the other approaches. However, the larger the number of VMs, the larger the number of physical machines on which there are VMs. This is the reason for the ascending trends in the methods. Nevertheless, the proposed method saved more energy due to performing the placement process more appropriately. It can be concluded that the proposed method, using the least amount of available resources, causes significant energy storage, and is suitable for reducing energy consumption. The experimental results demonstrated that the proposed algorithm was capable of saving up to 0.896%, 9.716%, and 10.8% energy consumption compared with the MDPSO, Kumar et al.’s, and Dahsti et al.’s algorithms, respectively.

| TABLE 6. First Scenario: Energy Consumption Comparison |
|------------------------------------------------------|
| VMs | EALBPSO | MDPSO [5] | Kumar et al.’s [10] | Dahsti et al.’s [12] |
| 50  | 0.90    | 0.91      | 0.92                | 0.92                |
| 100 | 1.01    | 1.01      | 1.03                | 1.20                |
| 150 | 1.29    | 1.30      | 1.40                | 1.60                |
| 200 | 1.43    | 1.45      | 1.65                | 1.75                |
| 250 | 1.68    | 1.70      | 1.92                | 1.97                |
| 300 | 2.23    | 2.25      | 2.47                | 2.50                |

Table 7 shows a comparison between the four methods in terms of the processor usage standard deviation. Considering a separate phase for load distribution among physical machines, the proposed method performed more successful than the other approaches. Increasing the number of VMs increased the standard deviation in four methods, because they had to use VMs. Nonetheless, the proposed method performed better than the other approaches.

| TABLE 7. First Scenario: Comparison of Scheduling Processors in terms of Standard Deviation |
|--------------------------------------------------------------------------------------------|
| VMs        | EALBPSO | MDPSO [5] | Kumar et al.’s [10] | Dahsti et al.’s [12] |
| 50         | 0.29625 | 0.29831  | 0.31523             | 0.31803             |
| 100        | 0.31526 | 0.32214  | 0.33426             | 0.33526             |
| 150        | 0.32351 | 0.33526  | 0.34957             | 0.34426             |
| 200        | 0.34426 | 0.35834  | 0.36732             | 0.35480             |
| 250        | 0.35627 | 0.37834  | 0.38341             | 0.36304             |
| 300        | 0.36785 | 0.38265  | 0.40256             | 0.37562             |

Increasing the number of VMs made most of the physical machines involved and somehow decreased the standard deviation in both methods. The experimental results demonstrated that the proposed algorithm was capable of 3.867%, 8.623%, and 6.953% improvements for the deviation of processors compared with the MDPSO, Kumar et al.’s, and Dahsti et al.’s algorithms.

- **Second Scenario: Changing Physical Machine Status**

This scenario makes a comparison between the performance of the proposed method and that of the other approaches based on the type of homogeneous or heterogeneous physical machines by analyzing the consequent effects on the simulation parameters. Table 8 shows the parameters used in this scenario.
TABLE 8. Parameters of the Second Simulation Scenario

| Parameter                     | Value |
|-------------------------------|-------|
| Number of physical machines   | 20    |
| Minimum number of VMs on a physical machine | 2     |
| PSO iteration                 | 100   |
| Number of VMs                 | 100   |

Figure 5 shows the average number of migrations in both homogenous and heterogeneous processing power and memory of physical machines. In the proposed method, all VMs were assigned to the physical machines. When the physical machines had different conditions, the proposed method performed better because it could use the status of physical machines in the rearrangement of VMs and deactivate a larger number of physical machines. However, the other approaches used only VMs, which benefited from physical machines above the threshold.

The experimental results demonstrated that the proposed algorithm was capable of 5.91%, 16%, and 16.267% improvements for the number of virtual machines migrations compared with the MDPSO, Kumar et al.’s, and Dahsti et al.’s algorithms, respectively.

Table 9 shows a comparison between the four methods in terms of energy consumption. In the proposed method, physical machines experienced no significant changes in both homogenous and heterogeneous states. In fact, more movements of the heterogeneous state had no significant effects on the number of passive physical machines.

FIGURE 5. Second Scenario: Comparing the Average Number of Migrations

Table 9. Second Scenario: Comparison of Energy Consumption

|                   | EALBPSO | MDPSO [5] | Kumar et al.’s [10] | Dahsti et al.’s [12] |
|-------------------|---------|-----------|---------------------|----------------------|
| Homogeneous       | 1.06    | 1.07      | 1.12                | 1.26                 |
| Heterogeneous     | 1.12    | 1.14      | 1.21                | 1.38                 |
Table 10 shows a comparison of the standard deviations of the processors. Four methods are affected by the homogeneous and heterogeneous states of servers. The proposed method outperformed the other approaches.

| Deviation of processors | EALBPSO | MDPSO [5] | Kumar et al.’s [10] | Dahsti et al.’s [12] |
|-------------------------|---------|-----------|---------------------|----------------------|
| Homo                    | 0.31526 | 0.32214   | 0.33426             | 0.32751              |
| Hetero                  | 0.33632 | 0.35826   | 0.37325             | 0.36204              |

**• Third Scenario: Changing the Algorithm Iterations**

In this scenario, the four methods are compared in terms of algorithm iterations and consequent effects on the simulation parameters. Table 11 shows the parameters used in this scenario.

| Parameter                          | Value |
|------------------------------------|-------|
| Number of physical machines        | 20    |
| Minimum number of VMs on a physical machine | 2     |
| PSO iterations                     | 20-100|
| Number of VMs                      | 100   |

Figure 6 indicates the average number of migrations in four algorithms. When the algorithm iterations decreased, the probability of finding an optimal solution and the number of optimal migrations decreased, too. As a result, the probability of obtaining an optimal solution reduced. On the other hand, increasing the algorithm iterations escalated the probability of finding an optimal solution; therefore, a larger number of VMs were transferred. However, when the algorithm iteration reached the maximum number, it had no effects on movements.

![FIGURE 6. Third Scenario: Comparing the Average Number of Migrations](image)
Table 12 shows a comparison of the four methods in terms of energy consumption. The larger the algorithm iterations, the higher the probability of finding the correct layout for VMs. As a result, a larger number of physical machines would be switched off, and energy consumption would decrease.

| TABLE 12. Third Scenario: Comparison of Energy Consumption |
|----------------------------------------------------------|
| Number of Iterations | 20 | 40 | 60 | 80 | 100 |
| EALBPSO             | 2.25 | 2.10 | 1.48 | 1.21 | 1.01 |
| MDPSO [5]           | 2.21 | 2.11 | 1.42 | 1.18 | 1.01 |
| Kumar et al.'s [10] | 2.27 | 2.13 | 1.49 | 1.19 | 1.03 |
| Dahsti et al.'s [12]| 2.28 | 2.15 | 1.48 | 1.17 | 1.02 |

Table 13 makes a comparison between the two methods in terms of the processor standard deviation. Increasing the algorithm iteration improved the performance of the proposed algorithm in load distribution. However, since the other algorithms focus on the energy consumption reduction, increasing the iteration failed to significantly improve the load distribution. The improvement in the proposed method was due to the decrease in the number of active physical machines. As a result, the load increased on the remaining physical machines.

| TABLE 13. Third Scenario: Comparison of the Scheduling Processors in Standard Deviation |
|---------------------------------------------------------------------------------------|
| Number of Iterations | Deviation of processors |
| 20 | 40 | 60 | 80 | 100 |
| EALBPSO | 0.42348 | 0.40875 | 0.38345 | 0.35512 | 0.32891 |
| MDPSO [5] | 0.43726 | 0.41256 | 0.39321 | 0.36581 | 0.33651 |

VI. Conclusion and Future Work

The energy-aware scheduling algorithm on DVFS-enabled datacenters was found capable of reducing energy consumption and, consequently, reducing the operational cost, carbon footprint, and CO₂ emissions in high-performance distributed computing systems (HPDCSs) while increasing the system reliability and availability. In this paper, the energy-performance trade off scheduling algorithm, called EALBPSO, was proposed. A method of improving the efficiency of resources and decreasing the energy consumption was shown to be the use of the dynamic placement of VMs. Using this approach, the cloud service providers would be capable of improving the resources efficiency through placing many VMs on a single server. The service providers can also reduce energy consumption by changing the status of idle nodes to the low-power status. Living migration can also result in the transfer of VMs amongst physical machines, which can be effective on workload reduction and keep at least a few of the physical machines active at all times. The dynamic placement of VMs involves two processes: first, the migration of VMs from a host that is applied less often to active hosts; second, the migration of VMs from the overload hosts. In the present study, the results of simulating the proposed method were analyzed. After that, it was compared with three other approaches in terms of specific parameters such as the average number of migrations, energy consumption, and load distribution. According to the results, the proposed method managed to deactivate or avoid turning on a larger number of physical machines, despite the migration overhead, through a better
placement of VMs. Consequently, the proposed method succeeded to decrease the total energy consumption. At the same time, it was shown successful in the distribution of the load among physical machines by considering an additional phase.

In future work, we plan to continue our study into the optimum allocation and selection of VMs with the aim of improving the energy efficiency of data centers with taking into account both I/O- and CPU-intensive workloads.

REFERENCES

[1] Harish Garg, A hybrid PSO-GA algorithm for constrained optimization problems, Applied Mathematics and Computation, Volume 274, 2016, Pages 292-305.

[2] Zhou, Z., Chang, J., Hu, Z., Yu, J., Li, F. A modified PSO algorithm for task scheduling optimization in cloud computing. Concurrency Computat Pract Exper. 2018; 30:e4970, vol. 20, issue. 24, https://doi.org/10.1002/cpe.4970.

[3] Rituraj Singh Patwal, Nitin Narang, Harish Garg, A novel TVAC-PSO based mutation strategies algorithm for generation scheduling of pumped storage hydrothermal system incorporating solar units, Energy (2017), doi: 10.1016/j.energy.2017.10.052.

[4] D. Kumar and Z. Raza, "A PSO Based VM Resource Scheduling Model for Cloud Computing," 2015 IEEE International Conference on Computational Intelligence & Communication Technology, Ghaziabad, India, 2015, pp. 213-219, doi: 10.1109/CICT.2015.35.

[5] Xu, M., Tian, W., Buyya, R. A survey on load balancing algorithms for virtual machines placement in cloud computing, Concurrency Computat: Pract Exper. 2017; 29:e4123, Vol. 29, Issue. 12, https://doi.org/10.1002/cpe.4123.

[6] Peng Xu, Guimin He, Zhenhao Li, Zhongbao Zhang, An efficient load balancing algorithm for virtual machine allocation based on ant colony optimization, International Journal of Distributed Sensor Networks, Volume: 14 issue: 12, 2018, https://doi.org/10.1177/155047718793799.

[7] X. Liu, Z. Zhan, J. D. Deng, Y. Li, T. Gu and J. Zhang, "An Energy Efficient Ant Colony System for Virtual Machine Placement in Cloud Computing," in IEEE Transactions on Evolutionary Computation, vol. 22, no. 1, pp. 113-128, Feb. 2018, doi: 10.1109/TEVC.2016.2623803.

[8] Bahman Keshanchi, Alireza Souri, Nima Jafari Navimipour, An improved genetic algorithm for task scheduling in the cloud environments using the priority queues: Formal verification, simulation, and statistical testing, Journal of Systems and Software, Volume 124, 2017, Pages 1-21, https://doi.org/10.1016/j.jss.2016.07.006.

[9] Y. -H. Jia et al., "An Intelligent Cloud Workflow Scheduling System With Time Estimation and Adaptive Ant Colony Optimization," in IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 51, no. 1, pp. 634-649, Jan. 2021, doi: 10.1109/TSMC.2018.2881018.

[10] Yan, J., Zhang, H., Xu, H. et al. Discrete PSO-based workload optimization in virtual machine placement. Pers Ubiquit Comput 22, 589–596 (2018). https://doi.org/10.1007/s00779-018-1111-z.

[11] Khorasavi A., Garg S.K., Buyya R. (2013) Energy and Carbon-Efficient Placement of Virtual Machines in Distributed Cloud Data Centers. In: Wolf F., Mohr B., an Mey D. (eds) Euro-Par 2013 Parallel Processing. Euro-Par 2013. Lecture Notes in Computer Science, vol 8097. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-40047-6_33.

[12] Seyed Ebrahim Dashti & Amir Masoud Rahmani (2015): Dynamic VMs placement for energy efficiency by PSO in cloud computing, Journal of Experimental & Theoretical Artificial Intelligence, DOI: 10.1080/0952813X.2015.1020519.

[13] An-ping Xiong, Chun-xiang Xu, Energy Efficient Multiresource Allocation of Virtual Machine Based on PSO in Cloud Data Center, Mathematical Problems in Engineering, Volume 2014, Article ID 816518, 8 pages, http://dx.doi.org/10.1155/2014/816518.

[14] A. Ibrahim, M. Noshy, H. A. Ali and M. Badawy, "PAPSO: A Power-Aware VM Placement Technique Based on Particle Swarm Optimization," in IEEE Access, vol. 8, pp. 81747-81764, 2020, doi: 10.1109/ACCESS.2020.2990828.

[15] Dinesh Reddy, V., Gangadharan, G.R. & Rao, G.S.V.R.K. Energy-aware virtual machine allocation and selection in cloud data centers. Soft Comput 23, 1917–1932, 2019. https://doi.org/10.1007/s00500-017-2905-z.

[16] Xinqian Zhang, Tingming Wu, Mingsong Chen, Tongqun Wei, Junlong Zhou, Shiyian Hu, Rajkumar Buyya, Energy-aware virtual machine allocation for cloud with resource reservation, Journal of Systems and Software, Volume 147, 2019, Pages 147-161, ISSN 0164-1212, https://doi.org/10.1016/j.jss.2018.09.084.

[17] Minoo Soltanshahi, Reza Asemi, Nazi Shafiei, Energy-aware virtual machines allocation by krill herd algorithm in cloud data centers, Heliyon, Volume 5, Issue 7, 2019, e02066, ISSN 2405-8440, https://doi.org/10.1016/j.heliyon.2019.e02066.

[18] Xiong Fu, Qing Zhao, Junchang Wang, Lin Zhang and Lei Qiao, Energy-Aware VM Initial Placement Strategy Based on BPSO in Cloud Computing, Scientific Programming, Volume 2018, Article ID 9471356, 10 pages, https://doi.org/10.1155/2018/9471356.

[19] Ebadifard, F, Babamir, S. A PSO-based task scheduling algorithm improved using a load-balancing technique for the cloud computing environment. Concurrency Computat Pract Exper. 2018; 30:e4368. https://doi.org/10.1002/cpe.4368.
[20] Beegom, A.S.A., Rajasree, M.S. Integer-PSO: a discrete PSO algorithm for task scheduling in cloud computing systems. Evol. Intel. 12, 227–239 (2019). https://doi.org/10.1007/s12065-019-00216-7.

[21] Harish Garg, S.P. Sharma, Multi-objective reliability-redundancy allocation problem using particle swarm optimization, Computers & Industrial Engineering, Volume 64, Issue 1, 2013, Pages 247-255, ISSN 0360-8352, https://doi.org/10.1016/j.cie.2012.09.015.

[22] Khosravi, A, Nadjaran Toosi, A, Buyya, R. Online virtual machine migration for renewable energy usage maximization in geographically distributed cloud data centers. Concurrency Computat: Pract Exper. 2017; 29:e4125. https://doi.org/10.1002/cpe.4125.

[23] H. A. Alharbi, T. E. H. Elgorashi and J. M. H. Elmirghani, “Energy Efficient Virtual Machines Placement Over Cloud-Fog Network Architecture,” in IEEE Access, vol. 8, pp. 94697-94718, 2020, doi: 10.1109/ACCESS.2020.2995393. Song X, Ma Y, Teng D. A load balancing scheme using federate migration based on virtual machines for cloud simulations. Math Prob Eng. 2015; 2015: pp. 1–11.

[24] Cho KM, Tsai PW, Tsai CW, Yang CS. A hybrid meta-heuristic algorithm for vm scheduling with load balancing in cloud computing. Neural Comput Appl. 2015; 26(6):1297–1309.

[25] Thirunakadam T, Kamalakannan P. Energy efficient multidimensional host load aware algorithm for virtual machine placement and optimization in cloud environment. Indian J Sci Technol. 2015; 8(17): pp. 1–11.

[26] Verma M, Gangadharan GR, Narendra NC, Vadlamani R, Inamdar V, Ramachandran L, Calheiros RN, Buyya R (2016) Dynamic resource demand prediction and allocation in multitenant service clouds. Concurr Comput Pract Exp 28(17):4429–4442.

[27] Wang X, Wang Y, Cui Y (2016) An energy-aware bi-level optimization model for multi-job scheduling problems under cloud computing. Soft Comput 20(1): 303–317.

[28] Ramezani F, Lu J, Hussain FK. Task-based system load balancing in cloud computing using particle swarm optimization. Int J Parallel Program. 2014; 42(5):pp. 739-754.

[29] Ramezani F, Lu J, Hussain FK. Task-based system load balancing in cloud computing using particle swarm optimization. Int J Parallel Program. 2014; 42(5):pp. 739-75.

[30] Barzegar, B, Motameni, H, and Movaghar, A. ‘EATSDCD: A Green Energy-aware Scheduling Algorithm for Parallel Task-based Application Using Clustering, Duplication and DVFS Technique in Cloud Datacenters’. 1 Jan. 2019, pp. 5135-5152.