Hybrid Heuristic Algorithm for Better Energy Optimization and Resource Utilization in Cloud Computing

Ali Abdullah Hamed Al-Mahruqi[1*], Gordon Morison[2], Brian G Stewart[3] and Vallavaraj Athinarayanana[4]

[1&4]Department of Electrical and Computer Engineering, National University of Science & Technology. Muscat, Oman
alialmahruqi@nu.edu.om

[2]School of Engineering and Built Environment, Glasgow Caledonian University, Glasgow, G4 0BA, UK,

[3]Department of Electronic and Electrical Engineering, University of Strathclyde, Glasgow, G1 1XW, UK

Abstract

Energy-efficient execution of the scientific workflow is a challenging task in cloud computing that demands high-performance computing to process growing datasets. Due to the interdependency of tasks in the scientific workflow applications, energy-efficient resource allocation is vital for large-scale applications running on heterogeneous physical machines. Thus, this paper proposes a Hybrid Heuristic algorithm based Energy-efficient cloud Computing service (HH-ECO) that offers a significant solution for resource allocation, task scheduling, and optimization of scientific workflows. To ensure the energy-efficient execution, the HH-ECO focuses on executing non-dominant workflow tasks through adaptive mutation and energy-aware migration strategy. HH-ECO adopts the Chaotic based Particle Swarm Optimization (C-PSO) principle to optimize the resource allocation, task scheduling, and resource migration by generating the global best plans without local convergence. C-PSO with adaptive mutation avoids the deterioration of global optima while finding the best host to place the virtual machine and ensures an appropriate resource allocation plan. By considering the workflow task precedence relationships during C-PSO based task scheduling, the novel hybrid heuristic method efficiently solves the multi-objective combinatorial optimization problem without dominance among the workflow tasks. The Cloudsim based simulation study delivers superior results compared to the existing methods such as the Hybrid Heuristic Workflow Scheduling algorithm (HHWS) and Distributed Dynamic VM Management (DDVM). The proposed approach significantly improves the optimal makespan to 38.27% and energy conservation to 38.06% compared to the existing methods.

Keywords: Green Cloud Computing, VM Migration, energy-efficient Resource Optimization, Multi-Objective Optimization Problem, Heuristic Algorithm, and Load Category.

1. Introduction

Cloud computing has enabled rapid on-demand resource provisioning of virtualized resources to fulfil the growing demand for computing power. The energy efficiency of cloud resources is a vital factor due to the rapid growth of cloud data centers that fuel the upsurge of energy requirements, which significantly escalates overall operational costs. Cloud data centers allocate virtualized resources to run applications for remote-end users. Managing the energy efficiency of cloud computing processes such as resource allocation, task scheduling, and resource migration is a vital factor. Energy-efficient resource allocation without violating the SLAs is a significant research issue in cloud computing due to the dynamicity, scalability, performance uncertainties, and heterogeneous application workloads that demand varying resource requirements [1]. Energy-efficient management of virtualized resources has to ensure Quality of Service (QoS) constraints of the end-user while improving the energy efficiency of operations such as efficient virtual machine allocation [2], task scheduling

[3, 4], and resource migration. Maintaining the trade-off between energy consumption and performance is often a challenging task while providing the best possible cloud service to end-users. In the cloud environment, idle Physical Machines (PMs) consume a considerable amount of resources, and it consumes more than 70% of its peak load power, which have addressed by several cloud implementations that suggest to switch off the idle machines to save energy [5]. However, in practice, it is demonstrated that energy consumption is higher when rebooting PMs due to the consumption of more time for stabilizing all the applications to work as a fully functional server after rebooting. To improve energy efficiency, several researchers have recommended that change a physical server into a sleep mode because it takes less time to stabilize all the applications [6] compared to the idle state. In contrast, overloaded PMs considerably deteriorate the quality of cloud services and slow down the execution speed, mainly due to the over-utilization of the available resources on a particular PM. Thus, optimally allocating the virtual resources to PM is crucial, and migrating overloaded Virtual Machines (VMs) to the best hosts is essential to optimize the resource utilization significantly [7]. Owing to the dynamic nature of the cloud, VMs accept dynamic workload while running applications that demand highly intensive computation, and thus, the utilization of the Central Processing Unit (CPU) varies with time.

A scientific workflow defines a set of interdependent tasks and their dependencies. It significantly changes the decision parameter on the resource management objective. Scientific workflows construct complex distributed solutions and enforce the effective usage of computational resources. Due to complex scientific operations, scientific workflow applications have become data-intensive and computation-intensive. Cloud computing is emerging as the widely utilized distributed computing environment that supports the execution of complex and large scientific workflow applications. Workflow decomposes the data-intensive and complex applications into a set of smaller tasks that are to be executed either in a serial or parallel manner concerning the nature of the application. A representation of workflow application is in the form of a Directed Acyclic Graph (DAG) that reflects the interdependency among the tasks. In a DAG structure, nodes indicate the computational tasks of the workflow application, and directed edges represent the dependencies between the tasks. During the execution of scientific workflow application, effective resource utilization is crucial to the success of robust problem-solving environments in high-performance computing. The traditional cloud computing approaches [8] not considering the dominant characteristics of scientific workflow tasks, which intricate the execution of interdependent jobs that demand substantially varying resource requirements. Thus, the violations of the assured quality of cloud services to the end-user has significantly increased in the scientific workflow. It is essential to focus on the orchestration of the workflow tasks execution on the VMs by considering the dominant characteristics. By leveraging these considerations, QoS must be ensured at a reasonable cost for the end-users while sustaining the optimal energy consumption of the resources.

Several research publications have employed the population-based meta-heuristic algorithms [9], such as PSO, which provides global solutions for the dynamic and scalable cloud environment. Even though the PSO-based algorithms provide better solutions without gradient information for combinatorial and multi-objective optimization problems, it leads to premature convergence. Improving the convergence rate of the PSO algorithm without premature convergence is a challenging task while considering the multiple objectives. Only limited research works have focused on exploiting the chaotic sequence with PSO to ensure the global convergence for multiple objectives while handling the workflow tasks in the cloud. This paper proposes a unified hybrid heuristic methodology, HH-ECO that accomplishes efficient resource management and energy optimization. The advantage of HH-ECO methodology is that it adopts the C-PSO, which accelerates the optimization of resource allocation, task scheduling, and resource migration by generating global best plans without local convergence. The proposed C-PSO algorithm based energy-efficient methodology attempts to tackle the dominance issue among the workflow tasks while scheduling and migrating the workflow tasks due to the ergodicity and disciplinarian characteristics of the chaotic method. In the HH-ECO approach, an enhanced C-PSO algorithm with an adaptive mutation solution controls the dominance area of one task to the other tasks and avoids the deterioration of global-optimum performance. It ensures the appropriate resource allocation plan to reduce the Makespan and ensure a green cloud environment. The HH-ECO approach employs the C-PSO algorithm with significant enhancements to optimize energy and non-dominant execution among the workflow

tasks during the execution of workflow tasks in the dynamic cloud environment, which addresses the stagnation problem. In essence, an enhanced C-PSO algorithm with an adaptive mutation mechanism facilitates in solving continuous problems and meets dynamic requests with a rapid convergence rate and less computation effort. Additionally, the HH-ECO methodology fine-tunes the scheduling results by globally exploring the cloud resources with the help of a self-adaptive global optimization technique of C-PSO with the adaptive mutation.

The organization of this paper is categorized into nine sections. Section 2 presents related works, and Section 3 discusses the theme of the problem. The proposed system models and preliminary details are described in section 4. Section 5 presents the problem formulation. The proposed methodology has been elaborated in Section 6. Section 7 presents the experimental setup and evaluation metrics. Section 8 illustrates the results, and section 9 summarizes the main conclusions of the work.

2. Related Work

Several resource management techniques have been implemented in the virtualized data center framework [4], [7], [8]. In cloud computing, the factors such as the heterogeneity of the task, the arrival of dynamic workloads over time, and workflow tasks create significant problems in resource allocation, task scheduling, and VM migration. This section provides a brief survey of several energy-efficient [10] resource management techniques with their advantages and limitations. The detailed analysis determines the essential characteristics of resource management algorithms and discusses various issues and solutions to optimize a single objective, bi-objective, and multi-objective [11] functions for green cloud computing.

2.1 Workflow Task Scheduling approaches

A multi-objective task scheduling algorithm is proposed in [12] and assigns tasks to VMs to improve the throughput and reduce the cost in terms of processing time. The assignment of task priority relies on the QoS requirements of the applications. The tasks and VMs are sorted using a non-dominated sorting based on the running capacity before task scheduling. This algorithm is suitable for QoS parameters such as execution time and cost, but not appropriate for energy efficiency. A meta-heuristic PSO algorithm is exploited to minimize the execution cost of the application in [13][14] [15]. However, the earlier standstill of the PSO particles before reaching an optimal global value, and the low-level diversification among the particles, creates premature convergence when a large number of jobs are running on the cloud. A Hybrid Heuristic Workflow Scheduling algorithm (HHWS) in [14] determines the optimal solution for the workflow tasks by combining a PSO algorithm and a Gravitation Search Algorithm (GSA). Even though the method provides cost and deadline-aware workflow scheduling, it fails to ensure the globally optimal solution when applying the GSA on affected tasks that may create a negative impact on the k-best particles within the PSO process, and thus, possible to allow local convergence. Consequently, it leads to the dominance execution among the workflow tasks while focusing on reducing the cost and deadline alone.

The research work [15] presents a discrete Comprehensive Learning PSO (CLPSO) approach for scheduling the workflow tasks based on a Set-based PSO (S-PSO) to address the user-defined QoS constraints. However, the meta-heuristic algorithms for workflow applications reduce the speed to reach the global optima. EnReal model [16] categorizes the VMs into splittable and non-splittable VMs and effectively utilizes the slack time of the workflow tasks. Consequently, it schedules the online scientific workflows in the cloud using different energy-efficient dynamic scheduling methods. A Multi-Objective list-based workflow scheduling heuristic [17] maintains a trade-off between the Makespan and energy efficiency, which is the extension of Heterogeneous Earliest Finish Time (HEFT), named MOHEFT. By applying the empirical models, it predicts the execution time and energy consumption for the workflow tasks based on the historical data of the real workflow task execution. However, historical execution based workflow scheduling leads to computational complexity. The evolutionary Multi-objective Optimization (EMO) model [18] resolves the problem of workflow scheduling on Infrastructure as a Service (IaaS) platform. It presents the encoding, population

initialization, fitness evaluation, and genetic operators concerning the problem, ensuring higher stability of task-instance assignments of workflows with the minimized Makespan and cost. Even though it mutates the tasks randomly, it lacks to perform the mutation during the execution of the workflow tasks, which leads to ineffective results in terms of higher energy consumption for the dynamic cloud environment. A Multi-Objective Genetic Algorithm (MOGA) ensures the minimized makespan based on the deadline and budget constraints and the improved energy efficiency for workflow scheduling in the cloud by considering the conflicting interest of the cloud stakeholders and employing the DVFS. Even though the MOGA model improves resource utilization during workflow scheduling in the cloud, it fails to improve the user satisfaction level [19]. A Cost and Energy-Aware Scheduling (CEAS) algorithm [20] minimizes the energy consumption and execution cost while considering the deadline by incorporating sub-algorithms, including VM selection, task merging, VM reuse, and the slack time reclamation algorithms. To schedule the workflow applications, the research work [21] applies a hybridization of GSA and Heterogeneous Earliest Finish Time (HEFT) to accomplish the bi-objective optimization to minimize both the makespan and cost. However, it provides ineffective results for the complex tasks that arrived in the cloud environment. Reliability and Energy Efficient Workflow Scheduling (REEWS) method [22] maximizes the reliability and minimizes the energy consumption while maintaining the workflow dependency and considering the QoS and deadline constraints specified by the users. It achieves the objectives by applying four main phases, such as priority computation, task clustering, target time distribution, and cluster assignment of the resources based on the voltage or frequency levels. The research work [23] presents an energy-efficient, cost-effective, and QoS-aware scheduling model for the workflow applications in the cloud. It employs the per-core DVFS method on multi-core heterogeneous processors and considers the effects of input error on the execution time of the tasks. A Hybrid algorithm-based scientific workflow scheduling approach [24] considers the execution time, load balancing, and monetary costs while scheduling the workflow tasks on the cloud computing resources. It applies the preprocessing to prepare the tasks for the scheduling algorithm based on the dependencies and employs the PSO algorithm for workflow scheduling. A hybrid meta-heuristic algorithm [25] resolves the discrete task-scheduling problem in the distributed computing environment using the Hybrid Discrete Particle Swarm Optimization (HDPSO) algorithm. By applying the Hill-Climbing algorithm with the local search trend, it balances exploitation and exploration and avoids the sub-optimal trap. The energy-aware workflow scheduling approach [26] employs the bat algorithm and presents the Energy-Aware, Time, and Throughput Optimization heuristic (EATTO) algorithm to minimize the execution time and energy consumption of the compute-intensive workflows in the cloud environment. Moreover, it maximizes the throughput without compromising the Quality of Service (QoS) while resolving the multi-objective optimization problem. A firefly based multi-objective scheduling strategy [27] considers the multiple objectives such as makespan, the workload of servers, reliability, and resource utilization in the cloud during the scheduling of the workflow tasks. It assigns the workflow tasks to the appropriate VM instances along with the balancing of the workloads and utilization, and minimization of the makespan with the reference of the deadline. An Improved Many Objective Particle Swarm Optimization (I_MaOPSO) approach [28] addresses the workflow-scheduling problem in the cloud considering four objectives, such as cost minimization, energy consumption minimization, makespan minimization, and reliability maximization. By applying the four greedy heuristic methods, it maintains the equilibrium between the exploitation and exploration during the scheduling and tends to the convergence of the non-dominated solutions along with the generation of the high-quality initial population, social leader selection, and cognitive leader selection to balance the intensification and diversification capabilities of the scheduling algorithm.

2.2 Resource Allocation and Migration approaches

Resource allocation is a process of allotting VMs to PMs. The user establishes service level agreements with the cloud service provider [29] and resource management techniques attempt to ensure the SLA as well as reduced energy consumption [30] [31]. Managing the SLA is formulated as a multi-objective combinatorial

problem in most of the recent applications [29] [32]. The research work [32] applies an ant colony optimization algorithm based VM placement strategy to reduce resource wastage and energy consumption. By applying the ACO algorithm, it obtains a Pareto set to resolve the multi-objective constraints. A Distributed Dynamic VM Management (DDVM) mechanism in [33] allocates and reactively reallocates the VMs using a First fit heuristic algorithm to accomplish the distributed and dynamic adaptation, eliminating the single point failure. However, its reactive VM management leads to an unmanageable optimal solution selection process during the arrival of numerous similar applications requiring VM management. This method maximizes power consumption due to the lack of consideration of the restart time of the inactive server. A dynamic resource allocation technique in [34] optimizes the allocated resources in VMs based on user demands. According to the dynamic workload over time, this technique adaptively multiplexes the VMs to PMs. However, heterogeneity among PMs and VMs encounters allocation issues within the above techniques, since it tends to imbalance a load of PMs. The research work [35] optimizes energy efficiency during communication and computation by enabling smart resource management for real-time vehicular cloud services. It accomplishes the QoS requirements by distributing the intensive computation to the distributed fog computing infrastructures. However, this adaptive resource management model fails to focus on reducing energy consumption during VM migration. In these situations, the energy-aware resource allocation algorithms presented in [36] and [37] take into account the resource utilization and energy consumption. Energy-efficient resource allocation has proposed traditional data centers that have distinguishing features such as heterogeneous workload, average load rate, and intensive completion time [38]. However, the assumption of allocating heavy load applications to new generation servers and light load applications to older generation servers deteriorates the efficiency of load balancing. Most resource management techniques exploit heuristic optimization algorithms to solve the energy-efficient placement of VMs in a cloud environment [39]. However, only a few of the techniques consider VM migration during running time, but these techniques result in significant time overhead. A scheduling method for multi-VM migration in the cloud is proposed in [40] to reduce VM migration time, but lack of correlation between migrated VMs considerably degrades the performance and increases the time of migration [41]. In essence, the determination of VM migration time is a significant challenge in cloud computing services [42]. There is a necessity to consider load category (i.e., idle, underloaded, balanced, and overloaded) in attaining better trade-off between resource utilization and Makespan without violating the SLAs. An Energy-Efficient Hybrid (EEH) framework [43] performs the request scheduling and the server consolidation to improve the efficiency of the electrical energy consumption in the datacenter. Initially, it sorts the user requests for the time and power needs to execute the tasks before scheduling and determines the underloaded and overloaded servers to migrate the virtual machines to the new servers to reduce the energy consumption.

3. Problem Theme

Several techniques have been suggested for cloud computing in terms of better performance and energy efficiency, but still, there is a deficiency in maintaining a trade-off between resource utilization and Makespan without violating SLAs. Most of the energy-aware resource management techniques consider a trade-off between under-utilization and over-utilization of resources as a primary way to produce an energy-optimized green cloud environment. However, the dynamic nature of workloads raises different issues and deteriorates the performance of the existing resource management techniques. It necessitates the investigation of multi-objective resource management schemes for the complex scientific workflow tasks. The possible ways to attain multi-objective resource management, including the provisioning of the optimal number of PMs for VM allocation, migrating under or over-utilized VMs to the best fit servers, switching OFF the un-utilized PMs, and providing energy-efficient task scheduling. Firstly, deciding the exact threshold for capacity utilization is critical without degrading the performance of the hosts to avoid the violations in maintaining a trade-off between the performance level and resource consumption. Secondly, current resource management techniques shut down

the VMs of the idle server to save energy. However, additional energy has been consumed to reboot the PMs, as discussed previously. The existing techniques allocate CPU and memory resources to workflow tasks based on the assumption that the individual best fit solution has no negative impact on another solution, which leads to a multi-objective combinatorial optimization problem for multi-interdependent task execution optimization. Thirdly, energy-efficient management of resources requires discovering the solution for each task that is non-dominated by other jobs in the solution space. Contrary to the single heuristic algorithm, hybrid heuristic algorithms significantly utilize the properties of different algorithms and achieve complementary advantages to solve multi-interdependent task execution optimization in the cloud environment. Thus, this proposed methodology attempts to provide energy optimization without violating SLAs using hybrid heuristic optimization algorithms.

4. System Model and Preliminaries of Algorithms

This section provides a system set up for executing the proposed HH-ECO algorithm over the cloud environment, including VM allocation, workflow task scheduling, and VM migration. Let, the virtualized cloud server consists of a set of *m* physical computing hosts defined by $H= \{PM1, PM2,...PMm\}$. For the i^{th} PM $(i = 1, 2,...,m)$, the processing Capability (*Ci*) is defined as the CPU performance in Millions of Instructions Per Second (MIPS), which varies for each PM. Let,*n* be the number of VMswithin any given *PMi* having different capacity, i.e.,$PMi = \{VM1i, VM2i, ...VMni\}$. Each *VMji* has the processing capability *Cji* that is subject to $\Sigma^{n}_{j=1} (Cji) \leqslant Ci$. There are two states for each *PMi,* such as active,i.e., *APMi*, when $Ci>15\%$, and idle, i.e.,*IPMi*, when $5\% \leqslant Ci \leqslant 15\%$. These values are decided from previous research work [44] that focuses on server performance on a practical high-end server.

PSO-based Workflow Scheduling: Three main steps involved in the PSO are computing the fitness of every particle, computing local and the global best fit, and updating the position and velocity of each particle, which are iterated until stopping criteria is met [45]. The PSO heuristic model optimizes the execution of a task-resource mapping according to its principles. Initially, the heuristic model computes the mapping for all the tasks without considering the workflow dependencies to optimize the overall computation cost of the workflow application. According to the PSO-based mapping, the scheduling heuristic validates the dependencies between the workflow tasks. If the mapped resource is unavailable for a particular task during execution, it recalculates the PSO based mappings to support the execution of the workflow tasks dynamically. In subsequence, it repeats these scheduling steps until scheduling all the workflow tasks [46]. In the first step, the PSO algorithm randomly initializes the position and velocity of the particles. In the context of workflow scheduling, the task-resource mapping process is the particles, and the number of workflow tasks is the dimension of the particles. By performing the fitness function, the evaluation of each particle has computed. The computational method PSO identifies the solution in the region of a given problem. By applying the simple mathematical formula over the position and velocity of particles, the PSO moves these particles in the region and identifies the best position.

CPSO-based Workflow Scheduling: In the PSO algorithm, random sequence helps simulate the analysis, sampling, and decision making, which reduces the computation time. The random sequence is appropriate for one set of tasks, but not enough for another set of tasks. The chaos is a random sequence with regularity, which speeds up the generation of chaotic sequence. Compared to the PSO algorithm, CPSO [47] improves the global convergence and ensures the global best solution in which chaotic system based generated sequences are substituted for the random sequence of the PSO. In the scheduling model, the CPSO exploits the chaotic sequence for initialization and updation, which is different from the PSO algorithm. With the incorporation of the chaotic sequences, the CPSO algorithm [48] determines a better scheduling plan and attains minimum time and cost than the PSO algorithm.

By using the Chaotic based Particle Swarm Optimization (C-PSO) algorithm, it is possible to carry out an optimal allocation of VMs on PMs using *Ci* during the idle state '*s*,' where no load is assigned. The value of *Ci* in terms of MIPS can be varied under different VM load conditions. The scientific workflow application is represented as a directed Acyclic Graph (DAG) model $G= \{T, E_T\}$ [36][49] in Figure 1, where *T* defines a set

of tasks, and the processes in the execution order are named as tasks, and E_T refers to a set of dependencies between tasks $\in T$ of an application. Figure 1 describes a set of processes with the following notations*(V, W, X, Y, Z)*.

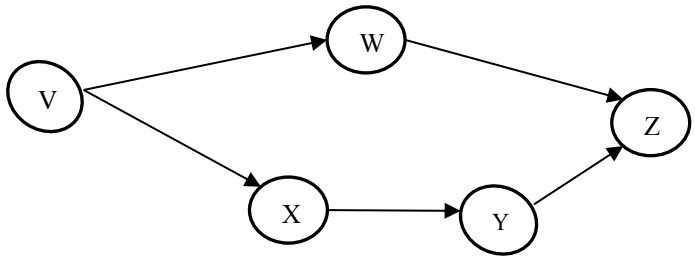


Figure1: Example Workflow of Tasks

An edge is defined as a set of dependencies between Tasks [36]. In Figure 1, the dependencies are in the form of edges $E_T = ((T_v)^1, (T_x, T_w)^2, (T_y)^3, (T_z)^4), \in T$ in which the superscripts denote the level of task processing. For instance, the succeeding tasks T_x and T_w depend on data generated by the preceding task (T_v) for its execution, and so they are executed in parallel after completing the task *V*. TaskT_v is in the first level of task processing, and after completing the T_v, the second level of tasks $(T_x, T_w)^2$ can be executed. Moreover, the exit task $T_z$cannot start before the completion of T_w and T_y. In other words, the task T_z depends on the data generated by the preceding tasks with the same priority ($T_w$andT_y) for its execution. For a given task, T_v, the parameters are denoted as $T_v = \{AT_v, TS_v, DL_v\}$, where AT_v, TS_v, and DL_v indicates the arrival time, task size, and a deadline of task T_v, respectively. Makespan is the execution time of an initial task along with its succeeding tasks, whereas the execution cost is the time expense of allocated VM for a task completed. The Makespan defines the completion time of tasks from T_v to T_z. To minimize the Makespan and execution cost, C-PSO takes the workflow task precedence relationship to provide adaptive task mutation. For this task, the workflow task precedence relationship, such as a set of Preceding Tasks and Succeeding Tasks of T_v is denoted as $\{PT\}T_v$and $\{ST\}T_v$, respectively. According to the disequilibrium state of resource utilization, C-PSO categorizes the active PM into Over Loaded (PM_{OL}), Under Loaded (PM_{UL}), and Balanced (PM_{BL}) [44] conditions. C-PSO takes the $PM_{OL}$and PM_{UL}as input and determines the best suitable destination $PM_{IL}$to execute the migrated tasks efficiently. The idle server utilization ranges between 5% to 15%, which is denoted as (PM_{IS}). The parameters below are introduced after stressing a high-end server in each category [44] in an ESXi 8.5 Hypervisor setup on PMs. The PM was consolidated with 17 VMs, and the test was conducted by sending the load to the VMs with results retrieved and recorded. The parameters may change slightly due to the nature of the load being considered and tested in a physical server. However, based on the behaviour of the server in terms of performance, these "fixed" values were considered in this paper for each category.

$$\text{Load category} = \begin{cases} PM_{OL(i)} & \text{if } C_i \geq 75\% \\ PM_{UL(i)} & \text{if } 15\% \leq C_i \leq 40\% \\ PM_{BL(i)} & \text{if } 41\% \leq C_i \leq 60\% \\ PM_{IS(i)} & \text{if } 5\% \leq C_i \leq 15\% \end{cases} \quad (1)$$

5. Problem Formulation

This section provides the fundamental issues in cloud computing processes and explains how the performance of HH-ECO is high, through a small example. Let the Cloud (CL) consists of PM, VM, Tasks.

The VM resources need to be allocated within each PM to execute the tasks. Each PM is modelled using its Makespan (MS), SLA violations, and Degree of Disparity (DD). DD defines the unevenness of workload among all the allocated hosts in which the minimum DD represents the situation when the system is balanced.

Definition: Given CL, HH-ECO aims to provide energy-efficient cloud computing processes such as resource allocation, task scheduling, and migration. These processes must return a considerable execution time, without violating the SLAs. However, CL includes heterogeneous PMs and VMs, which create both server overloading and underloading issues. The energy-efficient cloud computing problem may be considered as a problem space P^S in each process, where 's' represents the space. The particles 'p' is flown through the solution space, S^S. Every individual particle 'p' flies through P^S. Instead of providing the best solution to each particle, the best possible solution to all the particles without dominance is defined as the global optima. The impact of the best solution of a particle over other particles is named as the dominance relationship. The level of dominance relationship between two particles leads the time complexity, i.e., the time taken to run the HH-ECO algorithm, to either best, average, or worst case. Table 1 illustrates the time complexity of particles over the solution space in PSO. In the solution space, there are two different solutions, So1 and So2. Providing the best solution $s_1 \epsilon\ S^S$ for particle 'i', p_i tends the time complexity of another particle k to the average solution. Selecting the best or average solutions for each particle without dominating the solution of other particles reduces the time complexity (O). In Table 1, the solutions 1 and 2 are represented as So1 and So2, respectively. The task size represents the length of the task, and Waiting Time represents the amount of time a job is in the idle state before it is processed.

Table 1: Time Complexity of Particles over Solution Space in PSO

Problem Space	Solution So1	Solution So2
p_i	Best: O((Task Size/$C_{VM(i)}$))	Average: O((Task Size/C_{VM})+Waiting Time due to VM(i) overloading))
P_k	Average: O((Task Size/$C_{VM(i)}$) + Waiting Time due to VM(i) overloading))	Worst: O((Task Size/C_{VM}) + Migration Time to VM(j) + Waiting Time due to VM(j) overloading))

The SLA violation needs to be ensured with the Degree of Disparity (DD) defined by the equation (2), i.e.

$$DD = (ET_{max} - ET_{min})/ET_{avg} \quad (2)$$

Where, ET_{max}, ET_{min}, and ET_{avg} denotes the maximum, minimum, and average task execution time, respectively, among all the hosts. DD is mainly due to improper allocation of VMs and tasks in the computing phases. DD is primarily due to improper allocation of VMs and tasks in the computing phases. While using PSO, local convergence to an inappropriate solution is a primary reason behind the DD problem, since local convergence violates the SLA. Local convergence is referred to as the convergence of a particle for satisfying SLA before reaching its appropriate solution. To avert local convergence in PSO, the C-PSO method applies the process of mutating the particle to another position in S^S, which controls the dominance area (i.e., time complexity) of one solution to the other and avoids the deterioration of a global optimum. The mutation is the process of particle movement towards the global best position in which the movement relies on the particle

velocity, previous best position, and inertia weight. Applying C-PSO with a mutation in each phase in CL, it permits HH-ECO to provide a time computation efficient solution.

Resource Allocation and Task Scheduling: The mutation algorithm reduces the total execution time of all PMs and reduces the DD value by identifying the global best solution using C-PSO. The optimal C-PSO algorithm in HH-ECO derives a solution for each job and allocates the set of VMs in PM_i. The examination of energy-efficient resource allocation relies on the assumption of evenly distributed tasks among resources. Consider an equal load distributed among various PM_i. The execution time of tasks decides the total cost of PM_i and the DD value. The total cost is the summation of the execution time of 'l' tasks executed by the PM_i, i.e. $\sum_{tasks=1}^{l} Execution\ time_{\text{task}}$. To utilize the advantage of energy-efficient resource allocation, HH-ECO deals with the task scheduling pragmatically based on the even distribution of tasks. The task scheduling measures the fitness using Makespan and total execution time, where the fitness is defined as the condition of being a suitable solution without dominance. Based on the workflow, the tasks are mutated among PMs to reduce the waiting time, resulting in energy-efficient task scheduling.

VM Migration: Due to the lack of knowledge in overall PM capacity utilization, PM_i can be overloaded or under-loaded at any time. When a PM is in an overloaded state, the VM is migrated using the load category and C-PSO. Waking up the PM for migration increases the total cost. The waking-up process is likely to increase the responding time of the application in which the waking-up time varies according to the state of the resource, such as idle, sleep, or shut down. The decision of VM migration along with the task without considering the domination of the succeeding task may affect the MS. There are three cases of VM migration considered.

Case (i): if the resource is in an idle state, there is no need for a wake-up time because the idle resource is in an active state,

Case (ii): if the resource is in sleep mode, it takes a wake-up time of 300ms to become an active resource,

Case (iii): if the resource is in shutdown state, it requires a wake-up time of 480ms for the transition [50].

Thus, the consumption of wake-up time is *idle* < *sleep* < *shut down*. Also, to decrease the response time, applying the pre-wake up policy without sacrificing energy efficiency is necessary. Hence, HH-ECO focuses on implementing the pre-wake up policy considering the makespan, computational cost, and energy consumption. The following section explains the processes of the HH-ECO in detail.

6. The HH-ECO Methodology

The details of the HH-ECO in different phases are illustrated in Figure 2. The HH-ECO approach focuses on enriching three cloud-computing processes, such as VM allocation, task scheduling, and VM migration. In the VM allocation phase, each VM has the predefined CPU, memory, storage, IP address, and so on. The goal of VM placement is to find the best PM to run the appropriate VMs. The second phase is task scheduling, in which hierarchical task scheduling plays a significant role in workflow applications. Scheduling the tasks on available VMs, based on the Workflow precedence is essential. Thirdly, in the VM migration phase, the overloaded VM must be migrated to idle PMs.

The HH-ECO approach combines the Chaotic and PSO algorithms and adaptively applies the C-PSO to cloud computing processes. As PSO is a meta-heuristic algorithm that searches only the best fit solution in local optima, it incurs an earlier standstill of the particles before reaching the global optima due to the low-level diversification among particles. This problem is called premature convergence. Thus, the proposed work applies a chaotic mapping function to assist the particles in breaking away from the local optima when it reaches the premature convergence in each iterative searching process and improves the accuracy of the HH-ECO. To prevent the PSO algorithm from being trapped into a premature convergence, it is essential to apply control over local exploitation, i.e., taking advantage of previous best solutions, and global exploration, i.e., the heuristic search over new regions. PSO performance significantly relies on its parameters, especially the velocity of the previous particle. The previous velocity provides the necessary momentum for particles to find the best fit solution for the search space. Thus, proper control of the particle velocity is essential to determine the optimum

solution accurately and efficiently. To ensure a global convergence, the HH-ECO approach attempts to employ the chaotic sequence with high randomness and regularity, which improves the diversity among the solutions. Moreover, the adaptive inertia weight factor based proposed model effectively balances the local and global exploration to avoid premature convergence.

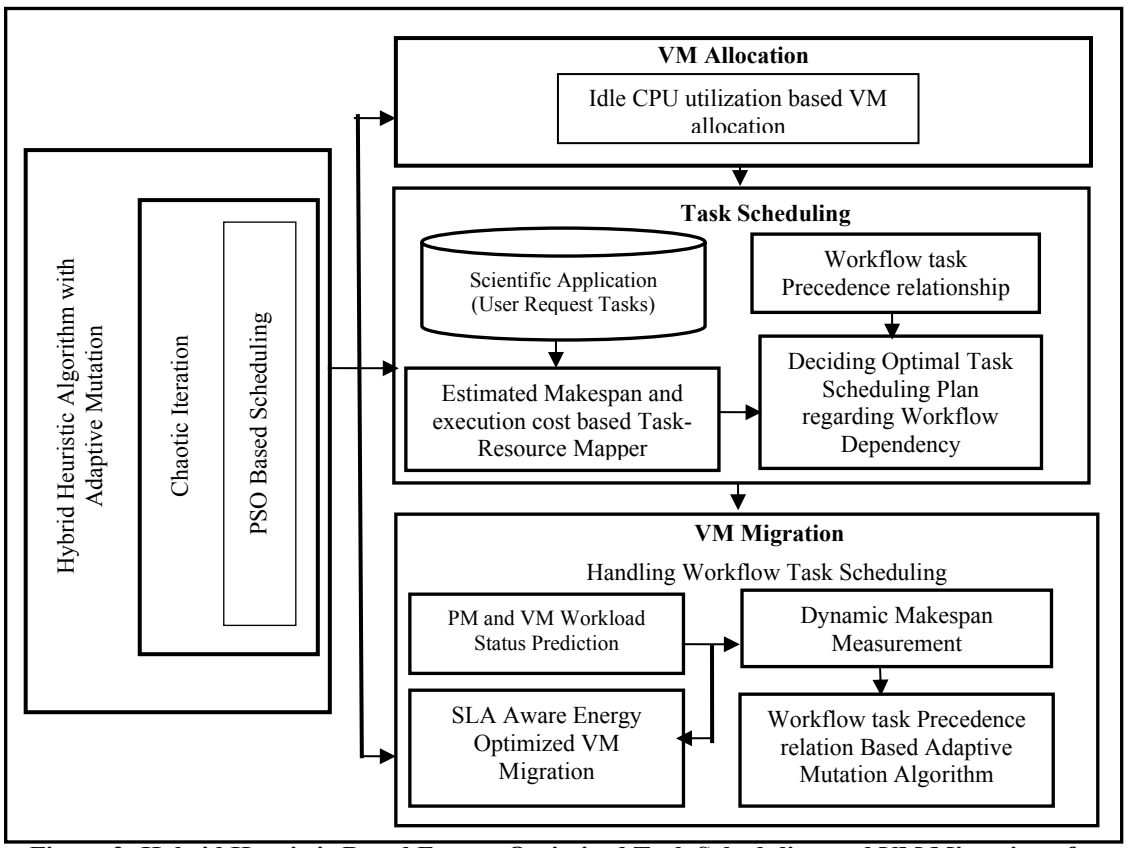


Figure 2: Hybrid Heuristic Based Energy Optimized Task Scheduling and VM Migration of Scientific Workflow

The HH-ECO approach attempts to identify the best solution to a particle problem over a region iteratively. The inertia weight plays a vital role in controlling the previous velocity of a particle on the movement of particle toward the best solution. The C-PSO adaptively controls the inertia weight using the fitness measurement and moves it toward the best solution without local optima. In essence, the randomness of the chaotic theory improves the PSO algorithm with the chaotic mutation operator. According to the PSO procedure, the CPSO algorithm prevents the convergence of the particle position and leads other particles to move away from the local optimal solutions within a short period.

6.1 VM Allocation with Adaptive Mutation

With the target of ensuring both the performance and energy efficiency, the HH-ECO focuses on allocating the VMs to minimize the energy consumption instead of meeting SLAs.The HH-ECO approach implements the VM allocation on heterogeneous PMs and VMs. CPU and memory utilization are the most critical components since they contribute significantly to the performance of servers [7]. Consider that ‘n’ number of VMs has to

be allocated in 'm' number of PMs. According to their capacity, the HH-ECO categorizes both the VMs and PMs into Small, Medium, and Large types [51] and then allocate the VMs to its appropriate PMs using the C-PSO algorithm.

Equation (3-5): The PSO models the simple mathematical formulae using particle velocity, position, and inertia weight. Note that, in every iteration, the C-PSO generates 'k' solutions, and the best solution is considered as G_{Best} among them. The previous positions of all particles in a j^{th} solution of the i^{th} iteration act as P_{Best} for the j^{th} solution of the $i+1^{th}$ iteration. The chaotic algorithm applies an inertia weight W to control the impact of the previous velocity on the current one and balance between exploitation and exploration in the PSO algorithm. Larger inertia weight guides the particles to the global search, while a smaller factor leads particles to a local search through the previously estimated best solutions. The HH-ECO estimates the adaptive inertia weight factor (W) and new velocity (V_j^{ITER+1}) using equations (3)-(5), where W_{max} and W_{min} refer the maximum and minimum values of inertia weight (W), and $Fitness_{max}$, $Fitness_{min,}$ and $Fitness_{avg}$ represent the maximum, minimum, and average fitness of the particle in all solutions of the i^{th} iteration respectively. Equation (4) computes the weighted threshold (W_α) based on the inertia weight and the fitness values of particles in the corresponding iteration to adaptively compute the inertia weight. In equation (5), c1 and c2 are positive constant parameters, and r1 and r2 represent the random value generated from the random function. The HH-ECO model applies the C-PSO algorithm to optimize each cloud computing phase using the equations (3-5).

$$\text{Inertia Weight, } W = \begin{cases} W_{min} + W_\alpha, & \text{if } Fitness_{max} \le Fitness_{avg} \\ W_{max}, & \text{if } Fitness_{max} > Fitness_{avg} \end{cases} \quad (3)$$

$$W_{\alpha=} \frac{(W_{max} - W_{min})(Fitness_{max} - Fitness_{min})}{Fitness_{avg} - Fitness_{min}} \quad (4)$$

$$V_j^{ITER+1} = W * V_j^{ITER} + c_1 r_1 \left(P_{Bestj} - X_j^{ITER}\right) + c_2 r_2 \left(G_{Best} - X_j^{ITER}\right) \quad (5)$$

According to the participation of a single type of PM over the total C_i of all the PMs, the HH-ECO approach allocates the VMs. An example of the PM allocation to A, B, and C types of VMs is depicted in Table 2. The total capacity of A, B, and C are presumed to be 30MB, 45MB, and 45MB, respectively. The allocation plan (A+B+C) for Small PMs is *20.6%* of total VM capacity (*i.e. 26/(26+47+53) = 20.6%*) to reach its Max speed. However, deciding the allocation per PM and how many VMs are allocated in each type A, B, and C to reach *20.6%* of VM capacity is quite complicated. The proposed work applies the best-fit meta-heuristic PSO algorithm for chaotic iterations and determines the VM allocation plan that fits with the capability of the PM.

Table 2: Example scenario for VM Allocation Strategy

PM type and Total Capacity (MB)		VM type and Total Capacity (MB)			VM Allocation (%)
		A	B	C	
Small (S)	26	30	45	45	20.6
Medium (M)	47				37.3

Large (L)	53				42

To demonstrate the steps of the C - PSO procedure in the cloud environment, two case examples are used, explaining the C-PSO processes in VM allocation.

Case 1: Initially, HH-ECO decides the participation level of each type of VM and executes the fitness function. Notably, the total participation level of all types of VMs is equal to *20.6%,* as per table 2. Different combinations of VMs are created to reach a *20.6%* participation level. Each combination is denoted as a particle and the position vector of particles. According to PSO, initially, the values of W and V_j^{ITER} are randomly chosen from 0 to 1. By using the W and V_j^{ITER} , the value of V_j^{ITER+1}Is estimated using equation (5). In the initial case, the value of $P_{Bestj}$and $G_{Best}$are equal to the position vector of the first particle. X_j^{ITER}also represents the first particle's position vector. Thus, both the terms, $c_1 r_1 \left(P_{Bestj} - X_j^{ITER}\right)$ *and* $c_2 r_2 \left(G_{Best} - X_j^{ITER}\right)$ tend to zero, and the new velocity value is equal to $W * V_j^{ITER}$. In the next solution of the same iteration, the $W_\alpha$value is identified using equation (3).

Case 2: After the first iteration, $W_\alpha$is estimated. Moreover, P_{Bestj} and G_{Best}values of the particle are also identified. In the previous iteration, if the value of fitness is maximum than $fitness_{avg}$, W_α is high and so the value of V_j^{ITER+1}Are also high, according to equation (5). This scenario represents that the location of the best solution is at a long distance from the previous solution, due to the high difference between them $(Fitness_{max} - Fitness_{min})$. A high value of $\left(Fitness_{avg} - Fitness_{min}\right)$ represents the closeness of the best solution. Thus, C-PSO reduces its velocity significantly. Moreover, if fitness is better than the average fitness, the high values of W and $V_j^{ITER+1}$are used to identify a suitable solution for cloud computing processes. A large inertia weight guides the particles to the global search, while the smaller factor leads particles to the local search through the previously estimated best solutions.

Equation (6): To reach the total capacity of a single type of PM, C-PSO selects different types of VMs (A, B, and C) in each iteration. For the selected VMs, the HH-ECO applies the equation (6) to measure the fitness for the VM allocation plan. The best Allocation Plan ($A_P = \{A_g, B_g, C_g\}$) is decided using the fitness measurement according to the Capacity (C_i) and Memory utilization (M_i) in many iterations, A_g represents the fraction of the number of VMs allocated to the 'A' type. The allocation plan iteration continues until fitness reaches the condition of $0.8 \leq$ Fitness $A_p < 1$, assuming that 80% of the suitability of solutions avoids imbalanced PMs. For the capacity of a single type of PM, the VMs are allocated. For instance, the participation level of a large capacity PM is *20.6%*. Equation (6) decides the allocation of the number of VMs required in each type to reach *20.6%* of the total VM capacity, where the total capacity and memory is *20.6%* of the total available VMs Capacity and memory respectively. The C-PSO algorithm executes equation (6) for Small, Medium, and Large capacity PMs individually to reach an optimal VM allocation.

$$Fitness\ A_p = \left\{\frac{\sum_{i=1}^{n} C_i}{total\ capacity}\right\} * \left\{\frac{\sum_{i=1}^{n} M_i}{total\ memory}\right\} \quad (6)$$

Equation (7): According to equation (6), the PM which has a number of VMs more than its capacity, i.e., fitness is greater than$Fitness_{avg}$, is classified under {q}. Otherwise, the PM is grouped under {p}. The VM resource allocation defines 'p' and 'q' as the number of VMs in the {p} and {q}, respectively, as mentioned in equation (7).

$$Particle\ Type = \begin{cases} \{p\} & if\ Fitness < Fitness_{avg} \\ \{q\} & if\ Fitness > Fitness_{avg} \end{cases} \quad (7)$$

HH-ECO plans to mutate a few of the VMs among PMs to equalize the capacity utilization of all PMs closer to 100%. When the fitness equation returns a value more than that of average fitness for the particles in {q}, the VM that has a high capacity in a particle has to be mutated with a VM that has low capacity in {q} to equalize the load.

6.2 Task Scheduling with Adaptive Mutation

In Workflow applications, hierarchical task scheduling plays a significant role. To effectively plan the task scheduling in a cloud environment, the Workflow task relationship must satisfy the quality constraints. At the task level, scheduling on-demand VMs to the tasks with knowledge of workflow can minimize the total cost without violating the SLA requirements. By applying C-PSO, the HH-ECO considers the static resource utilization of the PMs under idle state and dynamically estimated Makespan and execution cost of scheduled tasks by C-PSO as input to the fitness measurement. Total execution cost denotes the summation of the execution cost of all the subtasks.

Equation (8): The HH-ECO selects the best scheduling plan using the equations (3)-(5) based on the chaotic sequence and 'W' factor. When the scheduling plan returns a higher fitness value, it converges to the optimum. The high fitness value is returned only when the tasks allocated to a particular VM with minimum Makespan than the deadline and minimum total execution cost using equation (8). In equation (8), t1 and t2 denote the weighted parameters for two objectives of makespan and total execution cost, where, $0 \leq t1, t2 \leq 1$ and $t1+t2=1$.

$$Fitness\ TS_c = t1 * \left(1 - \frac{Makespan}{Deadline\ TS_c}\right) + t2 * \left(\frac{1}{Total\ execution\ cost}\right) \quad (8)$$

In consequence, the C-PSO based scheduling algorithm decreases the scheduling space and generates the best scheduling plan. However, the generated plan only satisfies the individual particle but does not consider the dominance of other particles. For example, according to the C-PSO, the Workflow tasks in Figure 1 are scheduled for small, medium, and large capacity VMs, as shown in Figure 3.

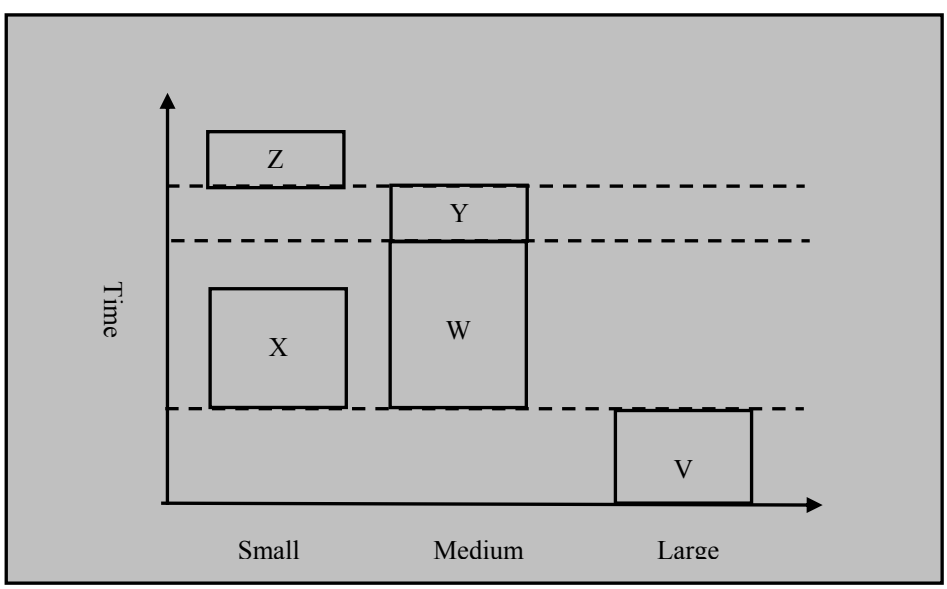


Figure 3: C-PSO Task Scheduling Plan

According to the C-PSO algorithm, task X is scheduled in a small capacity VM, after the execution of task V. However, when the succeeding task of X (Y) runs after the completion of task W, the Makespan will be increased. In essence, the decision of scheduling for task W without considering the dominance on Y affects the Makespan. To avoid this, the HH-ECO applies mutation to the task scheduling plan. Considering each task has its preceding task (PT)Ti and succeeding task(ST)Ti. If a task Ti dominates any one of its (ST) Ti, the HH-ECO applies the adaptive mutation. Accordingly, task X dominates task Y. Consequently, such a scheduling plan destroys the resource utility due to the delayed start time of the task Y by the task W even after the completion of its preceding task X. Hence, the HH-ECO approach mutates the task W on another best fit free VM based on the high value of the mutation-selection, which reduces the Makespan without wasting the resources and violating SLAs.

6.3 VM Migration with Adaptive Mutation

The HH-ECO has successfully scheduled the tasks according to the proposed task scheduling with the adaptive mutation. However, lack of knowledge in overall PM capacity utilization, each PM_i can be overloaded or under-loaded at any time. So HH-ECO monitors each PM after the tasks are scheduled and decides the state of PM ($PM_{OL(i)}$, $PM_{UL(i)}$, and $PM_{IL(i)}$). In the context of VM migration, consider $PM_{OL(i)}$ and $PM_{UL(i)}$, are source PMs. A set of predicted underloaded and overloaded PMs is denoted as $Source_{UPM} = \{S_{UPM1}, S_{UPM2}, ..., S_{UPMk}\}$ and $Source_{OPM} = \{S_{OPM1}, S_{OPM2}, ..., S_{OPMk}\}$ respectively. A set of imbalanced (active), idle servers are represented as destination PMs, Active $Dest_{(APM)} = \{D_{APM1}, D_{APM2}, ..., D_{APMk}\}$ and {Idle $Dest_{(IPM)} = \{D_{IPM1}, D_{IPM2}, ..., D_{IPMk}\}$ respectively. Moreover, $Dest_{PM}$ represents a combination of $Dest_{(APM)}$ and$Dest_{(IPM)}$.From source to destination, the VMs are moved until all the PMs are balanced. Consider that (n-k) VMs are allocated in the i^{th} PM. If the ratio of all the (n-k) VMs over the total capacity of PM exceeds the ¾ thC, then the PM is categorized into Overloaded. Otherwise, it is categorized as Under Loaded, which are finalized from the way of experiments discussed in [52, 53].

Equation (9): By experimenting, from the testing results, the proposed model decides that the idle server is 5%-15%, the underloaded server is 15%-40%, the imbalanced load server is 41%-60%, and the overloaded server is above 75% to avoid the performance degradation. These threshold limits have been determined by several factors, such as better energy consumption, the minimum number of VM migrations, and reduced SLA violations in the cloud environment. To attain the energy-efficient cloud, the proposed model performs this experiment as the prerequisite process that evaluates the performance of these factors for the different threshold range value.

$$\text{Load Category} = \begin{cases} \textit{Under Load} & \textit{if } 0.15C_i < \left(\frac{\sum_{j=1}^{n-k} C_{ji}}{C_i}\right) \leq \frac{3}{4} C_i \\ \textit{Over Load} & \textit{if } \left(\frac{\sum_{j=1}^{n-k} C_{ji}}{C_i}\right) > \frac{3}{4} C_i \\ & \textit{if } 0.41\% C_i < \left(\frac{\sum_{j=1}^{n-k} C_{ji}}{C_i}\right) \leq \frac{3}{5} C_i \\ \textit{Balanced} & \\ \textit{Idle} & \textit{if } 0.05\% C_i < \left(\frac{\sum_{j=1}^{n-k} C_{ji}}{C_i}\right) \leq \frac{0.6}{4} C_i \end{cases} \quad (9)$$

Equations (10-12): C_{VM} is the predicted capacity utilization of the VM that creates the host either over or underloaded. The Fitness value is equal to the inverse of the difference between $C_{i \in Dest(PM)}$ and capacity utilization of the $PM_{i \in Dest(PM)} + C_{VM}$, where $PM_{i \in Dest(PM)}$ represents the i^{th} PM in a set of $Dest_{PM}$. VM migration to the high fitness destination PM favors reduced under-utilization and over-utilization of PMs. When migrating the VM from source PM into new destination PM, the capacity of the destination PM must less than or equal to its total capacity after migration happens. Otherwise, the fitness value is zero. The VMs are migrated according to the high fitness destination measured by using C-PSO. The high fitness destination is considered as a suitable solution.

$$Fitness\ Source_{PM(i)} = \begin{cases} \left(C_{i \epsilon Dest(PM)} - \left(PM_{i \epsilon Dest(PM)} + C_{VM}\right)\right)^{-1} & if \left\{PM_{i \epsilon Dest(PM)} + C_{VM}\right\} \leq C_{i \epsilon Dest(PM)} \\ 0 & Otherwise \end{cases} \quad (10)$$

During the VM migration, HH-ECO allows the PM to enable sleep mode when all of its loaded VMs migrate. Moreover, it allows the PM in sleep mode to switch on only when there is no possibility to migrate a VM to an already active PM. However, the sleep mode PM might take time to become an active server. In such a time, the VM possibly executes the task in the source PM as per the scheduling plan and move it to the overloaded state. To reduce the energy consumption, HH-ECO tunes the idle server to sleep mode if a task is not allocated in the idle server. If a VM with the task 'i' is migrated due to either overloaded or underloaded PM, it ensures that the preceding task of 'i' is completed. After completing the preceding task, the task 'i' can be executed in the migratable VM. According to this procedure, HH-ECO estimates the triggering time point (τ_P) for waking up an active server from the sleep mode and triggers the server at the time of 'τ_P' and avoids unnecessary resource wastage. In equation (12), 'C_o' represents a constant value for the type of resource, and it is assigned as 2 sec for large capacity resources. MT_P and CT_P denote the migration time point and current time point, respectively, in which time point refers to the time in geometric space.

$$MT_P = CT_P + Completion\ Time\ \{PT\}T_v \quad (11)$$

$$\tau_P = MT_P - \{Wakingup\ Time + C_o\}_{VM} - MT_{VM} \quad (12)$$

MT_{VM} is the time consumed to migrate the VM from source to target resource. τ is the subtraction of $\{Waking\text{-}up\ Time + Co\}_{VM}$ and MT_{VM} from MT_P. After the completion of the preceding task $\{PT\}T_v$, the succeeding task can be executed in a migratable VM.

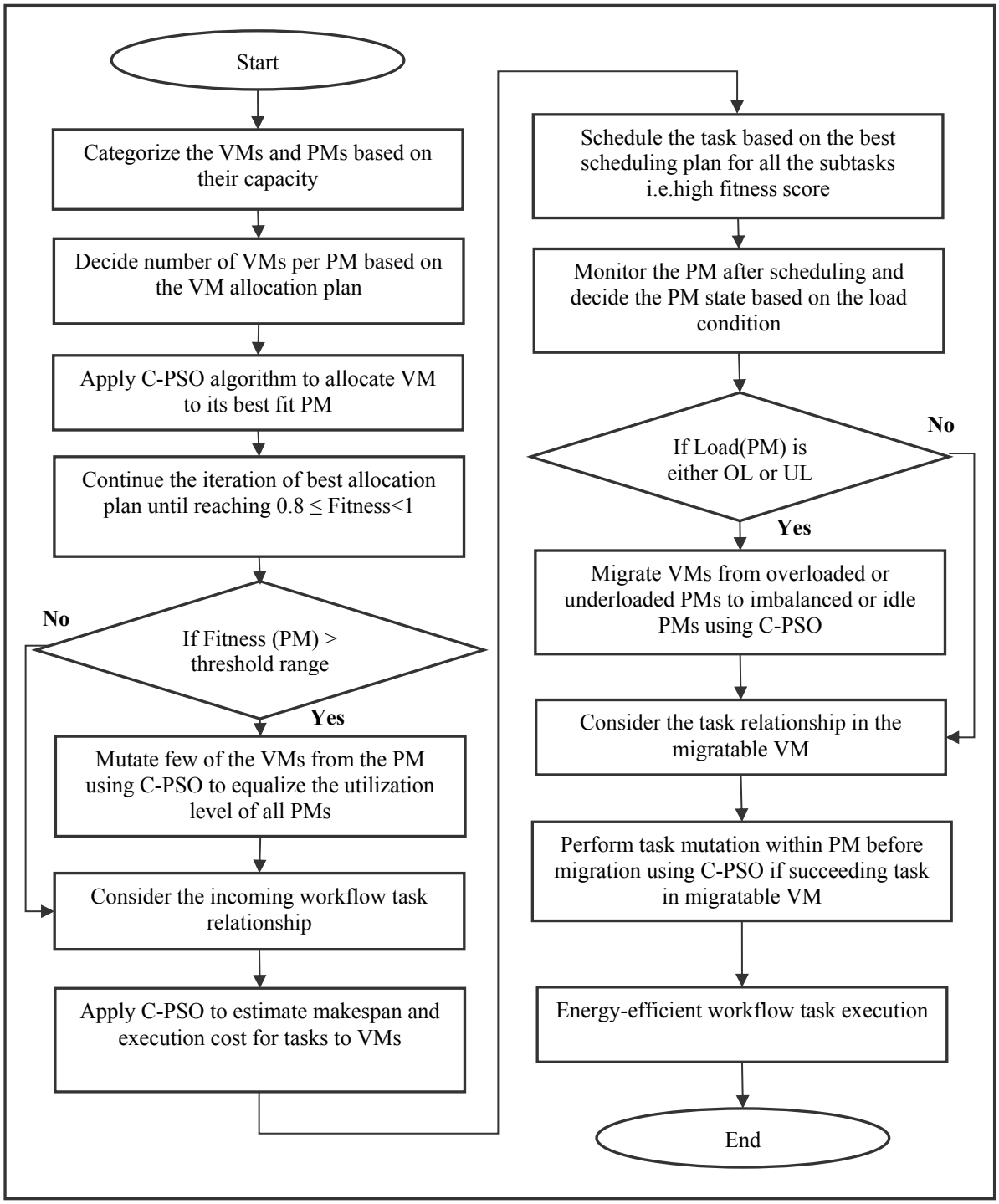

Start
Categorize the VMs and PMs based on their capacity
Decide number of VMs per PM based on the VM allocation plan
Apply C-PSO algorithm to allocate VM to its best fit PM
Continue the iteration of best allocation plan until reaching 0.8 ≤ Fitness<1
If Fitness (PM) > threshold range
No
Yes
Mutate few of the VMs from the PM using C-PSO to equalize the utilization level of all PMs
Consider the incoming workflow task relationship
Apply C-PSO to estimate makespan and execution cost for tasks to VMs
Schedule the task based on the best scheduling plan for all the subtasks i.e.high fitness score
Monitor the PM after scheduling and decide the PM state based on the load condition
If Load(PM) is either OL or UL
No
Yes
Migrate VMs from overloaded or underloaded PMs to imbalanced or idle PMs using C-PSO
Consider the task relationship in the migratable VM
Perform task mutation within PM before migration using C-PSO if succeeding task in migratable VM
Energy-efficient workflow task execution
End

Figure 4: Flowchart of HH-ECO Methodology

Moreover, the decision of VM migration along with the task without considering the domination of its succeeding task may affect the Makespan. To avoid this, HH-ECO attempts to apply task mutation among the VMs in the same PM, when there is an availability of better task scheduling before migration. Thus, HH-ECO attains better trade-off between resource utilization and makespan without violating SLAs. Figure 4 illustrates the overall sequential steps involved in the HH-ECO methodology.

7. Experimental evaluation

This section presents the experimental results using the WorkflowSim toolkit to evaluate the efficiency of the proposed approach and compares its performance with two existing cloud computing techniques.

Experimental setup: The experiments were conducted in a virtual environment using WorkflowSim. WorkflowSim is an extended model of CloudSim that supports workflow level simulation and enables seamless cloud modelling, simulation, and experimentation for the workflow tasks. The workflow model is based on the DAG model, enabling both the static and dynamic workflow schedulers. WorkflowSim comprises a set of components to simulate the workflow application; the components are workflow mapper, workflow engine, clustering engine, workflow scheduler, failure generator, and failure monitor. The experiment employs Cyber Shake workflow applications with sizes of 30, 50, and 100 workflow tasks.

Pricing for cloud services over different subscription periods is known as pricing policies. VM servers and pricing policies are based on the configuration in Amazon EC2 Infrastructure as a Service (IaaS), and on-demand VM instances are in the range of small, medium, and large. The proposed implementation model represents the processing capacity of the processor using MIPS. The simulated cloud data center consists of 10 host machines and 40 VMs involving three different types, such as small, medium, and large. Each type of host machine includes 3, 3, and 4 numbers of PMs, respectively, and the specifications of the host and VMs used in the simulated data center are given in Table 3.

Table 3: Specifications of Host and Virtual Machines

Cloud Resources	No.of machines	CPU/MIPS	RAM (MB)	Bandwidth (MB/s)	Storage (GB)
Host Machines	10	1000, 2000, 3000	1024, 2048, 3072	100	1
Virtual Machines	40	500, 1000, 1500	256, 512, 768	1	-

7.1 Evaluation metrics

The evaluation system employs a set of evaluation metrics to validate the efficiency of the proposed HH-ECO system as follows:

Makespan - the total time taken from the start time of the initial task to the end time of the final task of a sequence of tasks, which refers to the overall application completion time.

Degree of disparity - the measurement of unevenness among all the allocated hosts. The minimum value of the degree of disparity represents that the system is more balanced.

Average energy consumption - the average consumed energy to run the complete application on the allocated cloud resources during the simulation. The energy consumption of the computing server is based on the

utilization of the resources. Although it is an idle server, it also consumes energy to retain the running of memory, disks, and I/O resources. The remaining consumption linearly escalates when increasing the workload of resources.

SLA Violations – depends on the number of violating tasks from the task deadline and the number of violating hosts from the optimal energy consumption level while executing the workflow application.

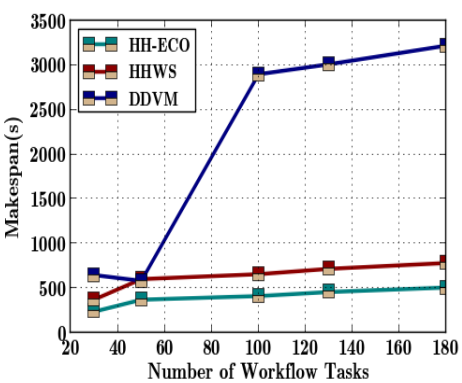
Total cost - the total cost spent to complete the entire application, including the amount spent on processing, memory, storage, and transfer cost along with the Makespan.

Migration efficiency - the ratio between the number of useful migrations and the number of migrations throughout the execution of the application. Useful migrations refer to (i) when the underloaded host becomes idle, and (ii) when the overloaded host becomes a balanced host after the migration completes.

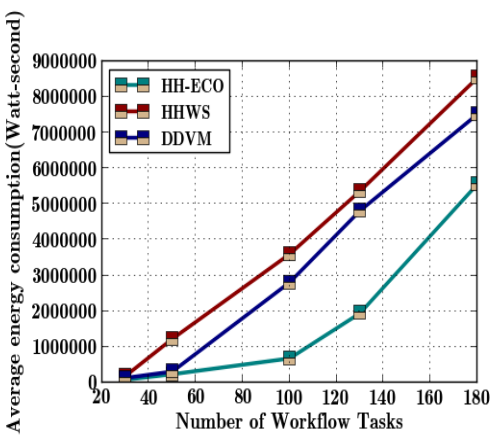
Waiting Time Ratio (WTR) – the ratio between the waiting time of the tasks during the execution of the application and the overall completion time of the application. The minimum value indicates that the system is efficient. Compared to the overall execution time, the waiting time should be a minimum. High waiting time leads to unnecessary power consumption at both user and cloud sides. The efficient allocation of VMs and tasks, as well as the workload, decides the waiting time.

8. Experimental results

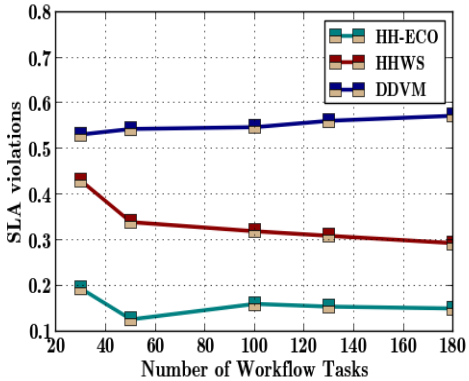
The simulation scenario conducts the following four tests: Test 1: Impact of the number of workflow tasks, Test 2: Impact of an average number of active hosts, Test 3: Impact of the percentage of resource utilization, and Test 4: Impact of server behaviour. The experimental framework compares the results obtained by the HH-ECO approach with the existing approaches such as the HHWS algorithm [14], which partially applies the mutation on the workflow tasks regardless of the optimal global solution, and DDVM [33], which merely takes into account reactive VM management with the concern of minimizing the SLA violations and power consumption.



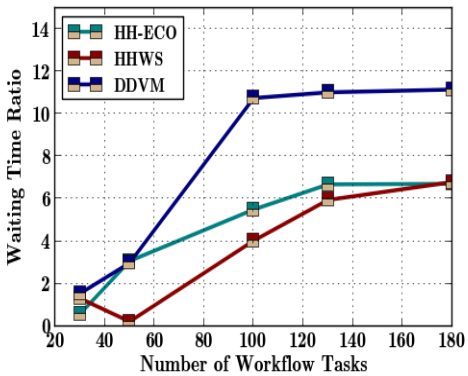
a) Makespan



b) Average energy consumption



c) SLA violations



d) Waiting Time Ratio (WTR)

Figure 5: Performance measures with varying Number of workflow tasks

Concerning Makespan, as shown in Figure 5a, it is seen that for the increasing number of workflow tasks, the HH-ECO algorithm gives minimum Makespan when compared to the existing HHWS and DDVM methods. Further, the HH-ECO approach maintains the workflow of the application and reduces the waiting time of the succeeding task, and ensures optimal execution of the workflow tasks. The performance in this approach is better than the HHWS approach when the Makespan is in the range of 35.07% to 38.27% and much better than the DDVM approach in the range of 36.2% to 85.74%. It is because the HHWS approach partitions a set of tasks to apply PSO and GSA algorithms separately. The proposed HH-ECO approach applies the chaotic based PSO algorithm during the allocation and scheduling of the workflow tasks, which tends to minimize the execution time even when there is an increased number of workflow tasks in the cloud environment. In principle, it proactively allocates the VM resources on the respective PMs and also considers the relationship among the workflow tasks during the scheduling as well as the migration.

As seen from Figure 5b, the average energy consumption in the HH-ECO approach is increased by 37.5% while the number of workflow tasks is varied from 30 to 100. However, the energy consumption of HH-ECO is less compared to the existing methods. It is because HH-ECO initially allocates VMs to the appropriate hosts in an energy-efficient manner and then optimally schedules the workflow tasks on the allocated VM resources using the C-PSO algorithm. Moreover, it applies mutation on the scheduled tasks to diminish the dominance in processing time and thus mitigates resource wastage and conserves considerable energy. On the other hand, in the same scenario, the average energy consumption of DDVM is higher, by around 11.76% compared to HH-ECO. The DDVM approach lacks in assigning the optimal maximum and minimum utilization level of the resource, which increases the number of stress resources. When the number of workflow tasks is higher, 180, the average energy consumption of the HH-ECO yields 31.76% and 25.64% lower than the HHWS and DDVM approaches. It is accomplished by the C-PSO based balancing of the resource utilization for the overloaded and underloaded resources during the workflow execution.

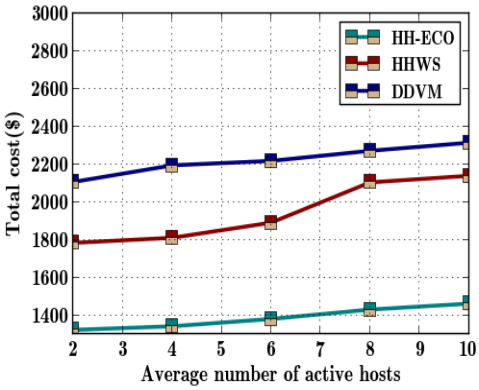
Figure 5c reveals the SLA violation ratio in the HH-ECO, HHWS, and DDVM approaches while varying the number of workflow tasks from 30 to 180. The SLA violations inversely expose the effectiveness of the system. Even when the number of workflow tasks with unique characteristics increases, the HH-ECO approach effectively restricts the violations within 15.76% and maintains the SLA. In contrast, the SLA violations are 33.88% and 55.18% in the HHWS and DDVM approaches. DDVM implements only the traditional scheduling of the first-fit heuristic method, which ensures minimum waiting time, whereas HH-ECO and HHWS can filter out the local optimal solutions. The unique consideration of the mutation in HH-ECO fetches an accurate global

solution throughout the application execution. Moreover, the HH-ECO approach analyzes the task relationship and performs the task mutation based on the task dependency before migrating to another resource in the cloud, which tends to avoid SLA violations. However, HHWS fails to consider the dominance relationship. Hence, the performance of HHWS is weaker than HH-ECO.

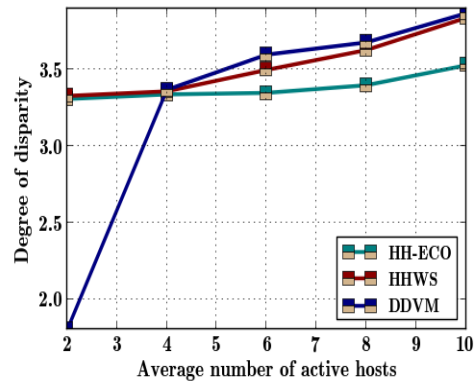
Figure 5d illustrates WTR of HH-ECO, HHWS, and DDVM approaches while varying the number of workflow tasks. WTR inherently represents the responsiveness of the cloud server to execute a workflow application, which linearly increases with the increase of the number of workflow tasks. The HH-ECO approach effectively manages the waiting time of each task even when the number of tasks in the workflow is high by applying the C-PSO algorithm with an adaptive mutation on the scheduled tasks. The waiting time ratio of HH-ECO decreases by 56% and 62% over HHWS and DDVM, respectively, when running 30 workflow task numbers. The HH-ECO approach obtains a 37.49% lesser waiting time ratio compared to the DDVM approach. It is because the HH-ECO approach estimates the fitness score for the workflow tasks with the consideration of the task dependency and adaptively mutates the tasks without compromising the SLA constraints.

As for Test 2 (Impact of an average number of active hosts), the total costs of the HH-ECO, HHWS, and DDVM approaches are shown in Figure 6a when varying the average number of active hosts from 2 to 10. Total cost is directly proportional to the total time spent and resource utilization that includes processing, memory, storage, and bandwidth. It reveals the overall performance of the system after completion of the workflow application execution on the remote server. The C-PSO algorithm with the adaptive mutation of the HH-ECO approach substantially reduces the total cost by 10.49% even when there is a need to start many hosts. The existing HHWS and DDVM methods lack in accessing the resources within the optimal utilization range, and thus these approaches rapidly increase the total cost up to 34.8% and 59.16% respectively over HH-ECO, even when utilizing an average count of 2 active hosts. By adaptively controlling the inertia weight and mutating the tasks, the HH-ECO approach effectively utilizes the cloud resources during the execution, which reduces the overall cost.

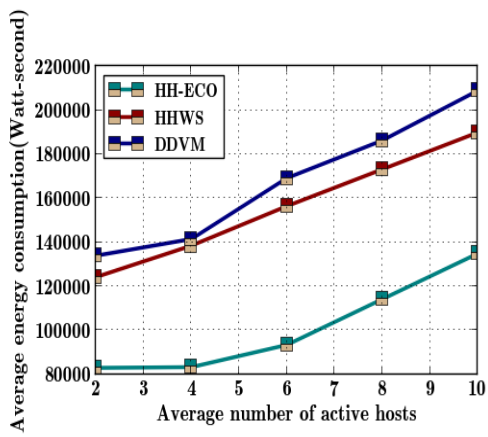
Figure 6b illustrates the DD of the HH-ECO approach in comparison to the HHWS and DDVM approaches. The degree of disparity represents the imbalance of the resources in the cloud environment. HH-ECO manages the degree of disparity value at 3.38, whereas HHWS manages the disparity value at 3.53. The DDVM approach can balance active hosts only when a minimum number of resources are utilized. Hence, DD is less, i.e., under 45.61% in the DDVM approach in comparison to HH-ECO when the average number of active hosts is 2. Moreover, the HH-ECO approach maintains the utilization of the active hosts in the cloud at a certain range of value but, the existing HHWS and DDVM approach obtain the variations on the degree of disparity when increasing the number of active hosts. It is accomplished by the balancing of the overloaded and underloaded resources through the C-PSO based migration with adaptive mutation during the execution of the workflow tasks in the cloud.



a) Total cost



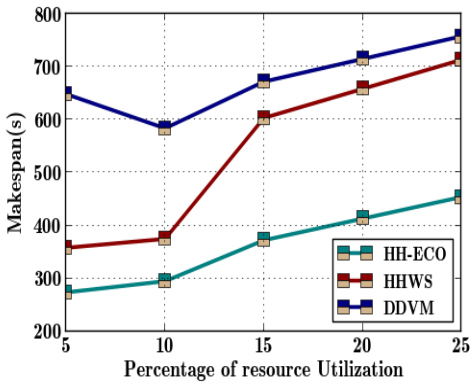
b) Degree of disparity



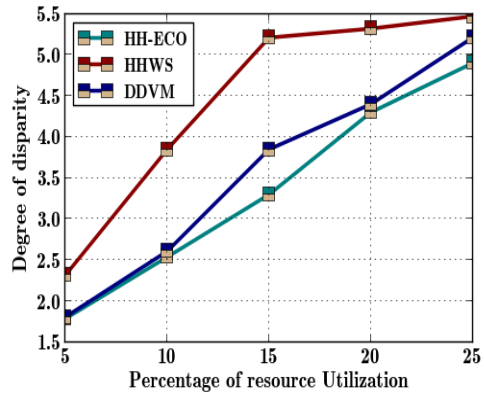
c) Average energy consumption

Figure 6: Performance with a varying average number of active hosts

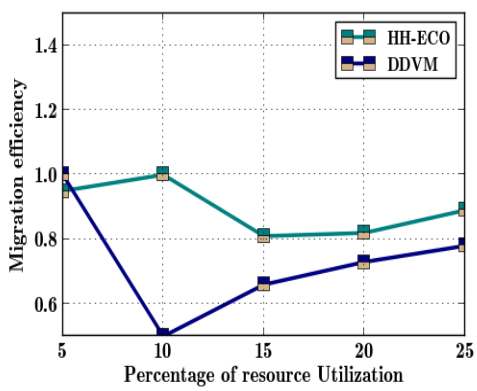
Figure 6c illustrates the average energy consumption of HH-ECO, HHWS, and DDVM approaches. The average energy consumption heavily relies on the number of active hosts during the execution of the workflow tasks in the cloud. The advantage of exploiting the C-PSO algorithm with an adaptive mutation on the workflow task execution creates a significant impact on the application completion time and, consequently, minimizes the overall energy consumption to 28.96% compared to HHWS and 35.4% for DDVM when utilizing an average count of 10 active hosts. Moreover, the HH-ECO approach gradually increases the average energy consumption while increasing the number of active hosts from 2 to 10, whereas DDVM and HHWS approach rapidly increases the average energy consumption. Notably, HH-ECO migrates the tasks when the allocated resources are either in the overloaded or underloaded condition during execution. Moreover, the idle server utilization policy also assists in minimizing energy consumption.



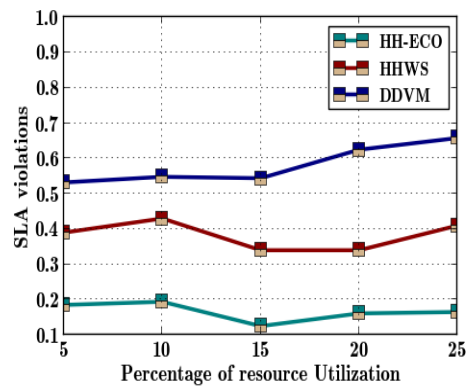
a) Makespan



b) Degree of disparity



c) Migration efficiency



d) SLA violations

Figure 7: Performance with varying percentage of resource utilization

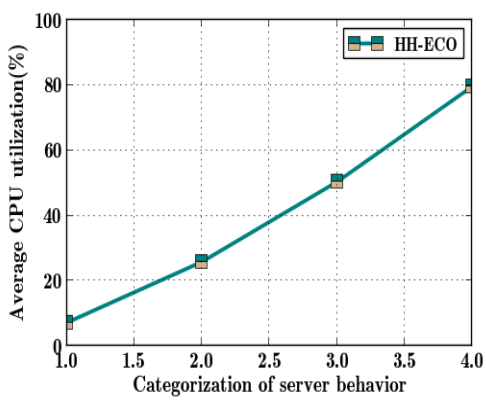
As shown in Figure 7a, the Makespan increases while increasing the percentage of resource utilization by a specific application. Among the three comparative approaches, the HH-ECO approach significantly reduces the Makespan compared to HHWS and DDVM because it focuses on maintaining the optimal utilization of each server at a specific level throughout the execution of the workflow application. Thus, the HH-ECO approach is efficient enough to complete the execution of workflow application in a reasonable time. If there is 25% resource utilization, the proposed approach accomplishes a better performance of 36.34% and 40.07% compared to the performance of HHWS and DDVM approaches, respectively. The proposed HH-ECO approach yields the minimum makespan even when there is an increased amount of resource utilization but, the existing HHWS and DDVM approach linearly increases the energy consumption with the increase of resource

utilization. It is accomplished by the effective utilization of the active resources for the corresponding scheduled workflow tasks with the minimum migrate rate for the execution of the workflow tasks.

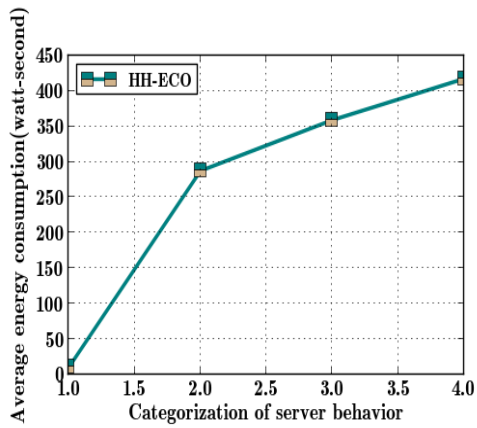
Figure 7b illustrates the value of DD for the HH-ECO approach in comparison to the HHWS and DDVM techniques. An increase in the percentage resource utilization escalates DD substantially over 41.89% in HH-ECO when increasing resource utilization from 5% to 10%. However, in HHWS and DDVM, the percentage DD has a drastic increase of over 65.94% and 44.19%. Significantly, the HH-ECO approach manages a negligible degree of disparity than both the HHWS and DDVM approaches. In contrast to the HHWS and DDVM, the HH-ECO approach performs the adaptive mutation during the VM resource allocation, task scheduling, and migration, which leads to the performance improvement with the advantage of the C-PSO algorithm on the workflow tasks.

Figure 7c shows the performance improvement in the Migration efficiency of HH-ECO compared with the DDVM approach when varying the percentage resource utilization. The HH-ECO approach maintains the migration efficiency throughout the application execution in the range of 89.4%. Although DDVM has a fair migration efficiency until it reaches a specific percentage of resource utilization, it shows a sudden drop of over 50% migration efficiency, and beyond a specific level of resource utilization, it gradually reaches a level of 73.4%. This is because the DDVM applies shut down and freeze, which leads to providing an inaccurate solution when the load percentage is high. Moreover, the HH-ECO approach applies the chaotic theory-based PSO algorithm during the migration of the overloaded or underloaded resources, which prevents the premature convergence in the decision-making of the optimal solution, and thus, ensures the optimal migration efficiency even when increasing the percentage of resource utilization.

Figure 7d demonstrates that stress or overloaded resources cause any SLA violations. When the percentage of resource utilization is high, the HH-ECO approach reschedules the tasks within a reasonable time to complete the execution of workflow application based on the C-PSO algorithm. The HHWS approach applies the PSO and GSA algorithms on two different sets of the tasks of one application. Thus the independent identification of the optimal solution is not useful in all cases because it is likely to misclassify while identifying the optimal solution within a set of tasks and resources, which is likely to extend the overall processing time resulting in many violations of time and energy. DDVM lacks in scheduling the tasks on the optimal resources as the allocation is based on the first-fit heuristic algorithm, which dynamically allocates the resources based on the current availability of resources in which the local solution appears as the optimal global solution. It leads to a violation of SLA in HHWS and DDVM of 21.6% and 41.54%, respectively, higher than that of the HH-ECO.



a) Average CPU utilization



b) Average energy consumption

Figure 8: Performance while varying Categorization of server behaviour

The evaluation results in terms of Average CPU utilization and Average energy consumption for different server behaviours are shown in Figures 8a and 8b. The server behaviours such as idle, under load, imbalanced, and overload are graphically represented as 1, 2, 3, and 4. The average CPU utilization of the allocated resources under various server behaviours is illustrated in Figure 8a. The HH-ECO approach optimally utilizes the CPU resources by applying an adaptive mutation method when there is a dominance of one solution over the other. This process averts the local convergence of the solutions as it determines the global solution in both the workflow application and cloud resources. When there are sudden changes in the allocated resources such as the state of underload or overload during the execution of workflow application, the proposed approach efficiently handles the tasks without violating the SLAs through adaptive mutation and task migration among the available resources. By effectively exploiting the optimal level of resource utilization, the HH-ECO approach obtains average CPU utilization for idle, under load, imbalanced, and overloaded servers as 7.16%,25.82%,50.38%, and 79%, respectively during application execution.

Figure 8b illustrates the increase in average energy consumption of the HH-ECO approach while varying the server categorization from the idle state to overload conditions. The average energy consumption varies concerning the server category. When the server is in an idle state, the energy consumption of the server is minimum regardless of the workload. The HH-ECO approach consumes reasonable energy as it considers the limited access level of the server resources rather than utilizing the entire available resources. The consideration of idle server restarts during the execution of workflow application facilitates less consumption of energy. The average energy consumption of the HH-ECO approach under the various server states of idle, under load, balanced, and overload is 2.49%, 68.89%, 86.03%, and 100%, respectively. Moreover, the adaptive mutation-based resource allocation, task scheduling, and migration in the HH-ECO approach leverage the energy-efficient execution of the workflow tasks in the cloud by considering the task dependencies and the objective factors.

9. Conclusions

A hybrid workflow scheduling algorithm, HH-ECO, for energy-efficient cloud computing has been proposed in this paper. The approach focuses on accomplishing multi-objectives via resource allocation, task scheduling, and resource migration phases underpinned by the application of the C-PSO algorithm with an adaptive mutation. It builds the system as an energy-efficient green cloud system through optimal allocation and avoids the dominant solution among other workflow tasks that ensure the optimal global solution rather than local convergence solutions. Moreover, the scheduling of the workflow tasks is performed dynamically based on the scheduling plan, workflow model, and the optimal global solution. Also, the HH-ECO migrates the overloaded or underloaded resources during the execution of workflow application, facilitating an improved quality of service and an energy-efficient cloud environment. The key achievements of the proposed methodology are:

- The Hybrid Heuristic Workflow algorithm provides an energy-efficient virtualized environment with high reliability and efficiency in resource allocation.
- Hybrid Heuristic algorithms allocate heterogeneous VMsbased on the capability of the host CPU and memory and assist to successfully satisfy the user demands by minimizing the cost of the schedulers.
- The C-PSO plans the scheduling procedure of workflow application to utilize the advantages of a green cloud environment; the mutation process improves the potential profit without violating SLAs.

- The Chaotic iterations in PSO avoid earlier standstill of particles before reaching global optima to provide a near-global optimal solution to promise user demands and achieve optimal energy consumption.
- C-PSO with adaptive mutation characterizes the resource utilization of VMs and generates the globally best migration plan to optimize the Makespan and cost.

The experimental results of the HH-ECO approach demonstrate a fair performance, accomplishing an optimal Makespan of 38.27% and significant energy conservation of 38.06% when compared with existing cloud computing approaches such as HHWS and DDVM. In future work, VM migration among various data centers needs to be focused along with the network latency. Additionally, instead of considering CPU and RAM as the source of performance and energy consumption alone, other resources such as cooling systems and network racks such as routers and switches can be considered, and suitable algorithms developed to improve performance.

References

[1] Hameed. A, Khoshkbarforoushha. A, Ranjan. R, Jayaraman. P. P, Joanna. K, Balaji. P and Zeadally. S, "A survey and taxonomy on energy-efficient resource allocation techniques for cloud computing systems," *Computing.*, Vol. 98, No.7, pp.751-774, 2016.

[2] Elghoneimy. E, Bouhali. O and Alnuweiri. H, "Resource allocation and scheduling in cloud computing," *IEEE International Conference on Networking and Communications (ICNC).,* pp.309-314, 2012.

[3] Singh. S and Chana. I, "A survey on resource scheduling in cloud computing: Issues and challenges," *Journal of Grid Computing*., Vol.14, No.2, pp.217-264, 2016.

[4] Kalra. M and Singh. S, "A review of metaheuristic scheduling techniques in cloud computing," *Egyptian Informatics Journal.*, Vol.16, No.3, pp.275-295, 2015.

[5] Beloglazov. A, Buyya. R, Lee. Y.C and Zomaya. A, "A taxonomy and survey of energy-efficient data centers and cloud computing systems," *Advances in computers*., Vol.82, pp.47-111, 2011.

[6] Barroso. L.A and Holzle. U, "The case for Energy-Proportional Computing," *Computer.,* pp. 33-37, 2007.

[7] Ahmad. R.W, Gani. A, Hamid S.H, Shiraz. M, Yousafzai. A and Xia. F, "A survey on virtual machine migration and server consolidation frameworks for cloud data centers," *Journal of Network and Computer Applications*., Vol.30, No.52, pp.11-25, 2015.

[8] Fakhfakh. F, Kacem. H.H and Kacem. A.H, "Workflow Scheduling in Cloud Computing: A Survey," *IEEE 18th International Enterprise Distributed Object Computing Conference Workshops and Demonstrations.,* pp.372-378, 2014.

[9] Usman, M.J., Ismail, A.S., Abdul-Salaam, G., Chizari, H., Kaiwartya, O., Gital, A.Y., Abdullahi, M., Aliyu, A. and Dishing, S.I., "Energy-efficient Nature-Inspired techniques in Cloud computing datacenters", Telecommunication Systems, Vol.71, No.2, pp.275-302, 2019

[10] Wu, Q., Ishikawa, F., Zhu, Q. and Xia, Y., "Energy and migration cost-aware dynamic virtual machine consolidation in heterogeneous cloud datacenters", IEEE transactions on Services Computing, Vol.12, No.4, pp.550-563, 2016

[11] Sharma, N.K. and Reddy, G.R.M., "Multi-objective energy efficient virtual machines allocation at the cloud data center", IEEE Transactions on Services Computing, Vol.12, No.1, pp.158-171, 2016

[12] Lakra. A.V and Yadav. D.K, "Multi-objective tasks scheduling algorithm for cloud computing throughput optimization," *Procedia Computer Science*., Vol.48, pp.107-113, 2015.

[13] Wu Kai, "A tunable workflow scheduling algorithm based on particle swarm optimization for cloud computing", 2014

[14] Mirzayi. S and Rafe. V, "A hybrid heuristic workflow scheduling algorithm for cloud computing environments," *Journal of Experimental and Theoretical Artificial Intelligence*., Vol.27, No.6, pp.721-735, 2015.

[15] Wei-Neng. C andZhang. J, "A set-based discrete PSO for cloud workflow scheduling with user-defined QoS constraints,"*IEEE International Conference on Systems, Man, and Cybernetics (SMC*)., pp.773-778, 2012

[16] Manojit. G, Verma. P, Karmakar. S and Sahu. A, "Energy efficient scheduling of scientific workflows in cloud environment," *IEEE 19th International Conference on High Performance Computing and Communications, IEEE 15th International Conference on Smart City, IEEE 3rd International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*., pp.170-177, 2017.

[17] Juan. D. J, Nae. V and Prodan. R, "Multi-objective energy-efficient workflow scheduling using list-based heuristics," *Future Generation Computer Systems*., Vol.36, pp.221-236, 2014.

[18] Zhaomeng. Z, Zhang. G, Miqing. L and Liu. X, "Evolutionary multi-objective workflow scheduling in cloud," *IEEE Transactions on parallel and distributed Systems*., Vol.27, No.5, pp.1344-1357, 2016.

[19] Rehman. A, Hussain. S.S, ur Rehman. Z, Zia. S and Shamshirband. S, "Multi‐objective approach of energy efficient workflow scheduling in cloud environments," *Concurrency and Computation: Practice and Experience*., p.e4949, 2018.

[20] Li. Z, Ge. J, Hu.H, Song.W, Hu.H and Luo.B, "Cost and energy aware scheduling algorithm for scientific workflows with deadline constraint in clouds," *IEEE Transactions on Services Computing*., Vol.11, No.4, pp.713-726, 2018.

[21] Choudhary. A, Gupta. I, Singh. V and Jana. P.K, "A GSA based hybrid algorithm for bi-objective workflow scheduling in cloud computing," *Future Generation Computer Systems*., Vol.83, pp.14-26, 2018.

[22] Garg. R and Mittal. M, "Reliability and energy efficient workflow scheduling in cloud environment," *Cluster Computing*., pp.1-15, 2019.

[23] Stavrinides. G.L and Karatza. H.D, "An energy-efficient, QoS-aware and cost-effective scheduling approach for real-time workflow applications in cloud computing systems utilizing DVFS and approximate computations," *Future Generation Computer Systems*., Vol.96, pp.216-226, 2019.

[24] Sardaraz, M., and Tahir, M. "A Hybrid Algorithm for Scheduling Scientific Workflows in Cloud Computing", IEEE Access, Vol.7, pp.186137-186146, 2019

[25] Shirvani, M. H. "A hybrid meta-heuristic algorithm for scientific workflow scheduling in heterogeneous distributed computing systems", Engineering Applications of Artificial Intelligence, 90, 103501, 2020

[26] Gu, Y. and Budati, C., "Energy-aware workflow scheduling and optimization in clouds using bat algorithm", Future Generation Computer Systems, Vol.113, pp.106-112, 2020

[27] Adhikari, M., Amgoth, T. and Srirama, S.N., "Multi-objective scheduling strategy for scientific workflows in cloud environment: A Firefly-based approach", Applied Soft Computing, p.106411, 2020

[28] Saeedi, S., Khorsand, R., Bidgoli, S.G. and Ramezanpour, M., "Improved many-objective particle swarm optimization algorithm for scientific workflow scheduling in cloud computing", Computers & Industrial Engineering, Vol.147, p.106649, 2020

[29] Gao. Y, Guan. H, Qi. Z, Hou. Y and Liu. L, "A multi-objective ant colony system algorithm for virtual machine placement in cloud computing," *Journal of Computer and System Sciences*., Vol.79, No.8, pp.1230-1242, 2013.

[30] Rai. R, Sahoo. G and Mehfuz. S, "Effect of VM Selection Heuristics on Energy Consumption and SLAs During VM Migrations in Cloud Data Centers," *Springer, Advances in Computational Intelligence: International Conference on Computational Intelligence.*, pp.189-199, 2017.

[31] Singh. S, Chana. I and Buyya. R, "STAR: SLA-aware autonomic management of cloud resources," *IEEE Transactions on Cloud Computing.,* Vol. 4, pp. 1-8, 2017.

[32] Mohammadhossein. M, and Kara. N, "Multi-objective ACO virtual machine placement in cloud computing environments," *IEEE Globecom Workshops* (GC Wkshps)., pp.112-116, 2014.

[33] Tighe. M, Keller. G, Bauer. M and Lutfiyya. H, "A distributed approach to dynamic VM management,"

IEEE 9th International Conference on Network and Service Management (CNSM 2013)., pp.166-170, 2013.

[34] Xiao. Z, Song. W and Chen. Q, "Dynamic resource allocation using virtual machines for cloud computing environment," *IEEE Transactions on parallel and distributed systems*., Vol.24, No.6, pp.1107-1117, 2013.

[35] Shojafar. M, Cordeschi. N and Baccarelli. E, "Energy-efficient adaptive resource management for real-time vehicular cloud services," IEEE Transactions on Cloud computing., Vol. 7, No. 1, pp. 196-209, 2016.

[36] Mehiar. D, Hamdaoui. B, Guizani. M and Rayes. A, "Energy-efficient resource allocation and provisioning framework for cloud data centers," *IEEE Transactions on Network and Service Management*., Vol.12, No.3, pp.377-391, 2015.

[37]Yuyang. P, Kang. D, Al-Hazemi. F and Youn. C, "Energy and QoS aware resource allocation for heterogeneous sustainable cloud datacenters," *Optical Switching and Networking*., Vol.23, pp.225-240, 2017.

[38]Quan. D. M, Mezza. F, Sannenli. D and Giafreda. R, "T-Alloc: a practical energy-efficient resource allocation algorithm for traditional data centers," *Future Generation Computer Systems*., Vol.28, No.5, pp.791-800, 2012.

[39] Kansal. N. J and Chana. I, "Energy-aware Virtual Machine Migration for Cloud Computing-A Firefly Optimization Approach," Journal of Grid Computing., Vol.14, No.2, pp.327-345, 2016.

[40] Quang-Hung. N, Thoai. N and Son. N.T, "Epobf: energy-efficient allocation of virtual machines in high performance computing cloud," Transactions on Large-Scale Data- and Knowledge-Centered Systems., pp.71-86, 2014.

[41] Zhang. Z, Xiao. L, Chen. X and Peng. J, "A Scheduling Method for Multiple Virtual Machines Migration in Cloud," *Springer, IFIP International Conference on Network and Parallel Computing*., pp.130-142, 2013.

[42] Rybina. K, Dargie. W, Umashankar. S and Schill. A, "Modelling the live migration time of virtual machines," *Springer International Publishing, OTM Confederated International Conferences on the Move to Meaningful Internet Systems*., pp.575-593, 2015.

[43] Alarifi, A., Dubey, K., Amoon, M., Altameem, T., Abd El-Samie, F.E., Altameem, A., Sharma, S.C. and Nasr, A.A., "Energy-Efficient Hybrid Framework for Green Cloud Computing", IEEE Access, Vol.8, pp.115356-115369, 2020

[44] Al-Mahruqi. A.A.H., Athinarayanana. V, Morison. G. and Stewart. B.G, "A Proposed Energy and Performance Aware Cloud Framework for Improving Service Level Agreements (SLAs) in Cloud Datacenters," *International Journal of Applied Engineering Research*., Vol.13, No.16, pp.12917-12922, 2018.

[45] Netjinda. N, Sirinaovakul. B, and Achalakul. T, "Cost optimal scheduling in IaaS for dependent workload with particle swarm optimization," The Journal of Supercomputing., Vol.68, No.3, pp.1579-1603, 2014.

[46] Thiago. G. A, Pietri. I, Sakellariou. R, . Bittencourt. L. F, and Madeira. E. RM, "A particle swarm optimization approach for workflow scheduling on cloud resources priced by cpu frequency*," Proceedings of the 8th International Conference on Utility and Cloud Computing*., pp.237-241, 2015.

[47] Xuejun. L, Xu. J, and Yang. Y, "A chaotic particle swarm optimization-based heuristic for market-oriented task-level scheduling in cloud workflow systems," *Computational Intelligence and Neuroscience*., Vol.81, 2015.

[48] Mohammed. A, Ngadi. Md. A, and Dishing. S. I, "Chaotic symbiotic organisms search for task scheduling optimization on cloud computing environment*," IEEE 6th ICT International Student Project Conference (ICT-ISPC).,* pp.1-4, 2017.

[49] Topcuoglu. H, Hariri. S And Wu. M.Y, "Performance Effective and Low-Complexity Task Scheduling for Heterogeneous Computing," *IEEE Transactions on Parallel and Distributed Systems*., Vol.13, No.3, pp. 260-274, 2002.

[50] Beloglazov. A, "Energy-efficient Management of virtual machines in data centers for cloud computing," *Dissertetion*, 2013.

[51] Cao. J, Yihua. W and Minglu. L, "Energy-efficient allocation of virtual machines in cloud computing environments based on demand forecast," *International conference on grid and pervasive computing*., pp. 137-151, 2012.

[52] Khoshkholghi. M.A, Derahman. M.N, Abdullah. A, Subramaniam. S and Othman. M, "Energy-efficient algorithms for dynamic virtual machine consolidation in cloud data centers," *IEEE Access*., Vol.5, pp.10709-10722, 2017.

[53] Beloglazov. A, Abawajy. J and Buyya. R, "Energy-aware resource allocation heuristics for efficient management of data centers for Cloud computing," *Future Generation Computer Systems*., Vol. 28, no. 5, pp. 755–768, 2012.