Workflow Scheduling Algorithm for Budget Constraint Green Cloud Computing

Medara Rambabu[1](✉), Robin Kurana[2], K. Praveen Kumar[3], and Srinivasa Rao Bendi[1]

[1] Gandhi Institute of Technology and Management, Visakhapatnam 530045, India
rmedara@gitam.edu

[2] Amazon, Gurugram 122002, Delhi, India

[3] Department of Information Technology, Vignan's Foundation for Science, Technology & Research University, Guntur, India

Abstract. Cloud computing is a promising platform for various applications and use cases, providing organizations with cost-effective, scalable, and flexible computing resources to support their needs. Therefore, businesses, research institutions, and educational organizations drive the adoption of cloud computing. Mainly cloud is the most affordable environment for executing scientific workflow applications. These applications demand high computing power and need to run for long times to complete. The cost and energy associated with workflows operating in data centers are significant, making it an important research topic in recent years. Achieving optimal scheduling of workflows in a cloud data center is a challenging NP-hard problem. This paper presents an efficient heuristic to reduce energy utilization in data centers while executing workflow applications without sacrificing performance. The proposed algorithm was evaluated using the WorkflowSim toolkit and benchmarked with various scientific workflows. The experiments' results demonstrate the proposed algorithm's effectiveness in optimizing energy consumption.

Keywords: Cloud computing · Data center · Monetary cost · Energy consumption · Deadline constraints

1 Introduction

Cloud computing has become critical in scientific research, providing a highly scalable, cost-effective, and flexible platform for executing workflows, managing data, and enabling collaboration across scientific sectors. The adoption of cloud services by scientific sectors for economic and operational benefits has been growing [1], and end-user spending on public clouds is projected to increase by 20.4% and 21.3% in 2022 and 2023, respectively [2, 3] as shown in Fig. 1. This growth has led to the establishment of massive data centers by tech giants globally, consuming a significant amount of electrical energy and water resources while also releasing harmful gases into the environment, posing a threat to the environment.

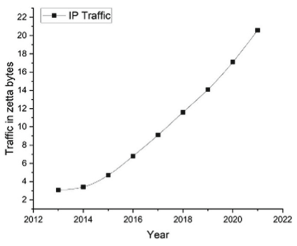

Fig. 1. End-user spending on public cloud worldwide [2]

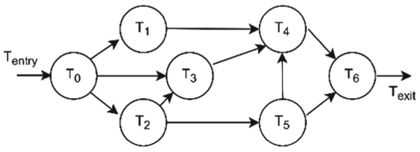

Fig. 2. A sample DAG

Major cloud service providers, including Amazon EC2, Google App Engine, Alibaba Cloud, and Microsoft Azure Services Platform, charge customers on an hourly basis, which may lead to overpaying when applications run for less than an hour. Energy consumption is a significant concern in cloud computing, with research indicating that server power and cooling account for half of data center management costs, while idle machines consume over 60% of energy usage. Developing energy-aware algorithms is, therefore, crucial for environmental and financial reasons [4].

This article extends previous work [5], which proposed the CEAS algorithm for cost and energy optimization in workflow scheduling with deadline constraints in cloud data centers. This article introduces an efficient approach to energy optimization while preserving performance by utilizing idle server time between the user-specified deadline and initial task completion time, resulting in significant energy savings (Figs. 1 and 2).

The article is structured as follows: Section 2 discusses related works, Sect. 3 presents problem formation and system models, Sect. 4 discusses the proposed implementation, Sect. 5 presents experimental settings and result analysis, and Sect. 6 concludes the article.

2 Related Works

Workflow scheduling problem in clouds is a broad studying topic. Many researchers proposed different techniques [6–9, 11] for scheduling workflows in a cloud environment. However, these works do not consider cost and energy together. The authors in [10] studied the trade-off between energy and reliability. Another energy-aware technique in [11] not considered the monetary cost of the application.

Further many works proposed to deal with this problem with different objectives makespan and cost. A real-time approach for scheduling multiple workflows called RMWS proposed in [12] optimizes application cost under varying deadline constraints.

Another work called MFGA [13] minimizing makespan and cost under deadline constraints. MFGA adopts a modified fuzzy GA (genetic algorithm) to achieve maximum resource utilization under budget. However, these works not consider energy optimization objective.

There have been several proposals to optimize energy utilization in data centers, as discussed in [14], [15]. One such proposal, discussed in [5], is a cost and energy-aware scheduling algorithm for scientific workflows in clouds with deadline constraints. The authors call this algorithm CEAS and it involves five steps aimed at reducing the cost and energy consumption of workflows in the context of cloud computing. Another approach to reducing energy consumption and cost, discussed in [9], involves identifying power inefficient virtual servers and removing them from the schedulable list. This approach employs techniques for finding common time between a set of virtual machines and evaluates a parameter ratio of effectiveness to accomplish this task.

A reliability and energy-aware scheduling method proposed by Gard et al. [16] aims to optimize both the lifetime reliability of an application and its energy consumption while ensuring that user-specified quality-of-service (QoS) requirements are met. The proposed algorithm operates in four distinct phases: priority calculation, task clustering, target time allocation, and assignment of clusters to processing elements with suitable voltage/frequency levels. However, this work does not consider cost and deadline constraints.

All these works perform well in saving valuable energy. However, these works are not effectively utilized the user-defined deadline for completion of the workflow tasks. A HEFT-based scheduling algorithm proposed in [17] maximizes the dormant virtual servers by identifying power inefficient machines.

3 System Model

The work aims to optimize energy utilization in data centers for workflow applications by considering various system models including data center, application, energy, and cost, which are described in the following subsections.

3.1 Data Center Model

This work assumed a cloud data center that contains heterogeneous computing resources. It has a set of virtual servers VMs each can be described by various computing parameters like computing capacity, memory, storage, bandwidth, etc. Another consideration is that all the VMs are DVFS enabled hence they can run at different frequencies and voltage levels depending on the load.

3.2 Application Model

This work considers workflow tasks modeled as directed acyclic graphs (DAGs). A workflow W is a set of dependent tasks $T = t_1, t_2, \ldots., t_n$ with set of communications edges $E = e_1, e_2, \ldots., e_m$ among the dependent tasks. A task without a predecessor is an entry or start task t_{start} and a task without a successor is an exit task t_{exit}. Hence a workflow DAG W is represented as W = (T, E). A sample DAG with seven nodes or tasks depicted in Fig. 2. Once a task t_i finishes its execution then the data generated by it is communicated to its successors if any.

In cloud services, a data center provides various VM instances to customers with different pricing models. This study assumes an hourly pricing model, as described in the cost model (section III C). It also assumes that there is no capping on provisioning of a VM instance type. Therefore, a separate VM instance is assigned to every task at the start. Before discussing the algorithm, certain parameters need to be defined, such as the computation time of a task on vm_k, which is the sum of execution time and communication time, as shown in Eq. (1).

$$T(t_i, vm_k) = \frac{L_{t_i}}{S_{vm_k}} + \frac{I_{t_i}}{B} \tag{1}$$

In the given formula, B represents the bandwidth, S_{vm_k} denotes the speed of vm_k, and L_{t_i} and I_{t_i} refer to the length and input size of task t_i, respectively. The effective execution time is calculated as $\frac{L_{t_i}}{S_{vm_k}}$, while the communication time (i.e., data transfer time) is calculated as $\frac{I_{t_i}}{B}$. To determine the minimum makespan of the workflow, the algorithm computes the earliest finish time (EFT) of the exit task t_{exit}. To do so, it is required to estimate the earliest start and finish times i.e., EST and EFT of t_i, which are determined using Eqs. (2) and (3).

$$EST(t_i) = \begin{cases} 0, \textit{if}\ t_i = t_{start} \\ \max_{t_{p \in parent(t_i)}} EFT(t_p), \textit{Otherwise} \end{cases} \tag{2}$$

$$EFT(t_i) = EST(t_i) + T(t_i, vm_k) \tag{3}$$

To calculate the optimal makespan ($minT_M$) of W, every task must be mapped to a virtual machine (VM) with the high processing power or the shortest makespan. The sub-makespan of every t_i is specified as in Eq. (4).

$$T_{sub}(t_i) = \frac{T(t_i, vm_k) * T_D}{minT_M} \tag{4}$$

Here, T_D represents the deadline of the workflow, and $T(t_i, vm_k)$ represents the execution time of task t_i on VM vm_k. The idea behind this approach is that if each task is executed within its sub-makespan, then the entire workflow can be completed within the given deadline. This can be easily understood using Eq. (4).

3.3 Cost Model

Assuming VM instances are charged on an hourly basis, this work considers fixed-time unit leases for end-users. The cost of task is determined by the execution time on vm_k, which is calculated using Eq. (5).

$$C(t, vm_k) = \lceil T(t, vm_k) \rceil * c_k \tag{5}$$

where $C(t, vm_k)$ is cost t_i on vm_k and c_k is hourly price of vm_k.

3.4 Energy Model

The energy consumption of a virtual machine $\mathrm{vm_k}$ executing a task t is defined as a product of its power consumption and its active time $\mathrm{P_k}$*T(t, $\mathrm{vm_k}$). The power consumption is determined by the CPU cycles that can be represented in terms of virtual machine frequency. Equation (6) expresses power consumption.

$$P_k = (v_k^j)^2 f_k^j \tag{6}$$

Here, $(v_k^j)^2$ and f_k^j are the voltage and frequency of virtual machine $\mathrm{vm_k}$ operating at level j. The voltage and frequency levels depend on whether the VM is in an active or idle state. For an active VM, the power consumption can be calculated using Eq. (7), while for an idle VM, it can be calculated using Eq. (8).

$$\mathrm{E_{active}} = (v_k^j)^2 * f_k^j * T(t, vm_k) \tag{7}$$

$$\mathrm{E_{idle}} = (v_k^{min})^2 * f_k^{min} * T_{idle}(vm_k) \tag{8}$$

where $T_{idle}(vm_k)$ is the idle period of the leased server vm_k. Finally, the overall energy is calculated as in Eq. (9).

$$E_{total} = E_{active} + E_{idel} \tag{9}$$

4 Proposed Work

Proposed work is an improvement on CEAS algorithm [5]. CEAS algorithm consists of VM selection, sequence and parallel task merging, VM reuse, and task slaking algorithms to optimize cost and energy utilization of workflows in clouds. VM selection chooses cost-optimal VMs, while sequence and parallel task merging minimize cost and energy by clustering tasks on the same VM. VM reuse effectively utilizes idle periods of leased VMs to minimize cost and energy. The task-slacking algorithm reduces energy utilization by reclaiming slack periods of workflow tasks. However, we found the algorithm ineffective in utilizing leased VMs and deadline to minimize energy. Section 3 presents our proposed problem formulation and system model.

We have improved the algorithm by extending sub makespan of workflow tasks to address the above issue. It saved more energy compared to the CEAS algorithm without compromising cost and performance. The proposed algorithm in this work is the improved version of the CEAS (MCEAS). The pseudo-code of the proposed MCEAS algorithm is shown in Fig. 3. Further, the detailed description of the improvement over CEAS i.e. makespan extension approach in MCEAS is presented in the following subsection.

```
The MCEAS algorithm
Begin
        Call VM selection algorithm
        Call Task clustering algorithm
        Call VM reuse algorithm
        While the application is not complete
            Map tasks to VMs

            Call slack algorithm
            End if
            Call makespan extension algorithm
        End while
End
```

Fig. 3. Pseudo-code of the proposed MCEAS algorithm

4.1 Makespan Extension Algorithm

The goal of this algorithm is to optimize energy while meeting the deadline constraint without affecting the cost metric. While the above steps do not increase the makespan of the workflow, they do not effectively utilize the user-defined deadline to optimize energy. Therefore, the proposed algorithm aims to utilize the gap between the deadline and actual finishing time to extend the makespan under deadline constraints, reducing energy consumption without increasing monetary costs. The makespan algorithm is as follows:

In the proposed algorithm, *Step-1* involves arranging all the tasks of a workflow in a topological order.

In *Step-2*, each task is assigned a number in the following manner: first, all numbers for every task of W are initialized to zero.

Step-3: Iterate over the topological order of the tasks in a reverse manner and update the number for every task which is the predecessor of the currently picked task $t_{current}$ as given by Eq. (10):

$$\mathrm{num}_{t_i} = (\mathrm{num}_{t_i}, \mathrm{num}_{t_{current}} + 1) \tag{10}$$

where num_{t_i} is number on t_i.

In *Step-4* of the algorithm, the tasks are sorted based on the numbering assigned in *Step-2*. This numbering ensures that tasks with the same number can be considered independent.

In *Step-5,* the extension is calculated as the difference between the deadline and the makespan. Tasks with the same number are marked as independent, and for each set of independent tasks, an extension constraint is applied to extend the task without increasing the cost. This extension is done based on the idle time available on the assigned virtual machine. If a task is under the reuse category and has a following task, the extended sub-makespan can be calculated using Eq. (11), otherwise, it can be calculated using Eq. (12).

$$T'_{sub} = T_{sub}(t_i) + \min(\text{extension}, EST(t_{i-next} - EFT(t_i)) \tag{11}$$

$$T'_{sub} = T_{sub}(t_i) + \min(\text{extension}, T_{idle}(t_i)) \tag{12}$$

Step–6: Then update the sub-makespan as in *Step-5*, supply frequency and voltage as described by Eqs. (13) and (14) respectively.

$$f_{t_i} = \frac{f_{t_i} * T_{sub}(t_i)}{T'_{sub}(t_i)} \tag{13}$$

$$v_{t_i} = \frac{v_{t_i} * T_{sub}(t_i)}{T'_{sub}(t_i)} \tag{14}$$

where f_{t_i} and v_{t_i} are the frequency and voltage the server respectively where task t_i is executing. Finally, the makespan extension is updated as described in Eq. (15).

$$\text{extention} = \text{extension} - \max_{t_i \in t_n} T'_{sub}(t_i) - T_{sub}(t_i) \tag{15}$$

5 Performance Discussion

The performance of the MCEAS algorithm was evaluated on WofklowSim [18] using four scientific workflows Montage, CyberShake, LIGO, and Sipht which are from different areas. The physical structures of the four scientific workflows used in this paper are depicted in Fig. 4. The complete characterization of these workflows is presented in [19]. The various performance parameters of VM instances taken from [5].

The MCEAS outperformed the other two algorithms in terms of energy-saving. In Figs. 5–8 the energy utilization is reduced by decreasing a deadline. The EHEFT algorithm performed poorly compared with the other two approaches. Whereas CEAS performed well compared to EHEFT but was less energy-saving compared to MCEAS. Further, we have evaluated the proposed algorithm for cost savings. It performed on par with CEAS algorithms hence the experimental results were not included in the article. We can conclude that our proposed approach is better in saving valuable energy without increasing the costs to the customer.

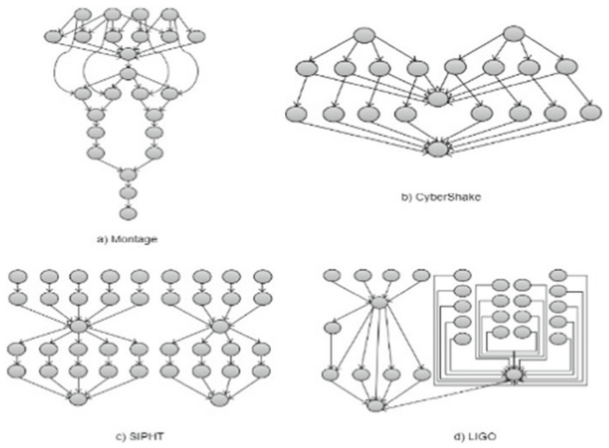

Fig. 4. Structure of various scientific workflows

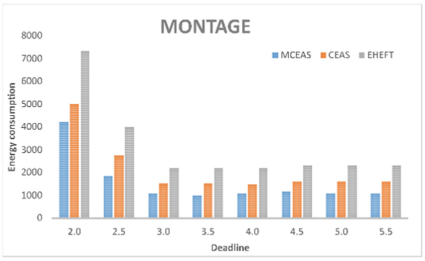

Fig. 5. Energy utilization on Montage workflow

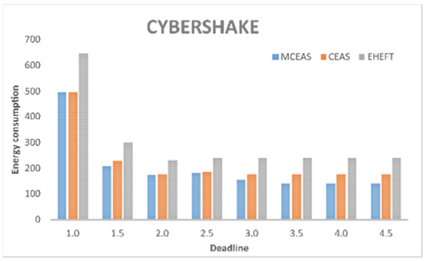

Fig. 6. Energy utilization on CyberShake workflow

The experiment results of our MCEAS algorithm compared with the two other algorithms CEAS [4] and EHEFT [16]. We have evaluated our algorithm for energy consumption using Montage, CyberShake, Sipht, and LIGO as shown in Figs. 5, 6, 7 and 8 respectively by varying workflow deadlines. It is clear from Figs. 5, 6, 7 and 8 that the proposed algorithm is efficient in reducing energy consumption irrespective of diverse workloads. The proposed algorithm performs well when choosing the VMs for scheduling tasks and switches of inefficient machines as in EHEFT and CEAS which can save valuable energy and as well monetary costs. Along with this, the MCEAS implements the makespan extension without compromising the deadline constraints to reduce energy consumption further.

In terms of energy utilization savings, the proposed approach outperformed the existing algorithms CEAS and EHEFT. Specifically, it achieved energy savings of 26% and 32% with the Montage workload, 9% and 24% with the CyberShake workload, 1% and 30% with the Sipht workload, and 6% and 23% with the LIGO workload, respectively.

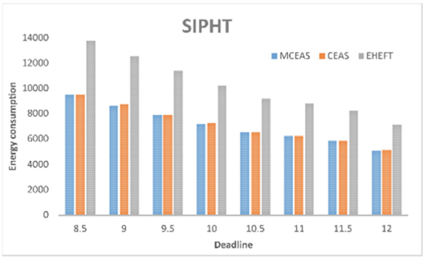

Fig. 7. Energy utilization on Sipht workflow

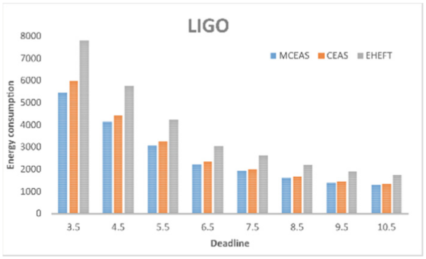

Fig. 8. Energy utilization on LIGO workflow

6 Conclusion

This study proposes a modified cost and energy-efficient scheduling technique for cloud schedulers to reduce energy usage while meeting user deadlines. The technique consists of five sub-algorithms: a cost-based VM selection algorithm, cost, and energy-aware clustering methods, a cost-aware VM reuse policy, an energy-aware slacking algorithm using DVFS, and a makespan extension algorithm that reduces energy usage. All sub-algorithms run in polynomial time suitable for commercial use. Experiment results show good energy utilization compared to existing algorithms. Future work will focus on generalizing the algorithm to handle combined workflows, considering actual electricity costs for scheduling, and updating the energy model to include RAM and hard drive energy usage.

References

1. Medara, Rambabu, and Ravi Shankar Singh. “A Review on Energy-Aware Scheduling Techniques for Workflows in IaaS Clouds.” *Wireless Personal Communications* 125.2 (2022): 1545–1584.
2. Vailshery, L.S. Public cloud services market size 2017–2023. https://www.statista.com/statistics/273818/global-revenue-generated-with-cloud-computing-since-2009/ (2022). [Online; accessed 15-Jan-2023].
3. Medara, Rambabu, and Ravi Shankar Singh. “Dynamic Virtual Machine Consolidation in a Cloud Data Center Using Modified Water Wave Optimization.” *Wireless Personal Communications* (2023): 1–19.

4. Wang, Shangguang, et al. “Particle swarm optimization for energy-aware virtual machine placement optimization in virtualized data centers.” *2013 International Conference on Parallel and Distributed Systems*. IEEE, 2013.
5. Z. Li, J. Ge, H. Hu, W. Song, H. Hu and B. Luo, “Cost and Energy Aware Scheduling Algorithm for Scientific Workflows with Deadline Constraint in Clouds,” in *IEEE Transactions on Services Computing*, vol. 11, no. 4, pp. 713–726, 1 July–Aug. 2018.
6. Alkhanak, Ehab Nabiel, and Sai Peck Lee. “A hyper-heuristic cost optimisation approach for scientific workflow scheduling in cloud computing.” *Future Generation Computer Systems* 86 (2018): 480–506.
7. Belgacem, Ali, and Kadda Beghdad-Bey. “Multi-objective workflow scheduling in cloud computing: trade-off between makespan and cost.” *Cluster Computing* 25.1 (2022): 579–595.
8. Gupta, Swati, et al. “User defined weight based budget and deadline constrained workflow scheduling in cloud.” *Concurrency and Computation: Practice and Experience* 33.24 (2021): e6454.
9. Medara, Rambabu, Ravi Shankar Singh, and Mahesh Sompalli. “Energy and cost aware workflow scheduling in clouds with deadline constraint.” *Concurrency and Computation: Practice and Experience* 34.13 (2022): e6922.
10. Medara, Rambabu, and Ravi Shankar Singh. “Energy efficient and reliability aware workflow task scheduling in cloud environment.” *Wireless Personal Communications* 119.2 (2021): 1301–1320.
11. Medara, Rambabau, et al. “Energy efficient virtual machine consolidation using water wave optimization.” *2020 IEEE congress on evolutionary computation (CEC).* IEEE, 2020.
12. Ma, Xiaojin, et al. “Real-time multiple-workflow scheduling in cloud environments.“ *IEEE Transactions on Network and Service Management* 18.4 (2021): 4002–4018.
13. Rizvi, Naela, et al. “A workflow scheduling approach with modified fuzzy adaptive genetic algorithm in IaaS clouds.” *IEEE Transactions on Services Computing* (2022).
14. Liu, Xing, et al. “Energy-aware task scheduling with time constraint for heterogeneous cloud datacenters.” *Concurrency and Computation: Practice and Experience* 32.18 (2020): e5437.
15. Safari, Monire, and Reihaneh Khorsand. “Energy-aware scheduling algorithm for time-constrained workflow tasks in DVFS-enabled cloud environment.” *Simulation Modelling Practice and Theory* 87 (2018): 311–326.
16. Garg, Ritu, Mamta Mittal, and Le Hoang Son. “Reliability and energy efficient workflow scheduling in cloud environment.” *Cluster Computing* 22.4 (2019): 1283–1297.
17. Thanavanich, Thanawut, and Putchong Uthayopas. “Efficient energy aware task scheduling for parallel workflow tasks on hybrids cloud environment.” *2013 International Computer Science and Engineering Conference (ICSEC).* IEEE, 2013.
18. Chen, Weiwei, and Ewa Deelman. “Workflowsim: A toolkit for simulating scientific workflows in distributed environments.” *2012 IEEE 8th international conference on E-science.* IEEE, 2012.
19. Juve, Gideon, et al. “Characterizing and profiling scientific workflows.” *Future generation computer systems* 29.3 (2013): 682–692.