IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received 1 December 2023, accepted 27 February 2024, date of publication 29 February 2024, date of current version 8 March 2024.

*Digital Object Identifier 10.1109/ACCESS.2024.3371693*

RESEARCH ARTICLE

# An Efficient Task Scheduling for Cloud Computing Platforms Using Energy Management Algorithm: A Comparative Analysis of Workflow Execution Time

**ADEEL AHMED[1], MUHAMMAD ADNAN[1], SAIMA ABDULLAH[1], ISRAR AHMAD[1], NAZIK ALTURKI[2], AND LEILA JAMEL[2]**

[1]Department of Computer Science, Faculty of Computing, The Islamia University of Bahawalpur, Bahawalpur, Punjab 63100, Pakistan
[2]Department of Information Systems, College of Computer and Information Science, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia

Corresponding author: Adeel Ahmed (adeelmcs@gmail.com)

This research is supported via funding from Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2024R333), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

**ABSTRACT** Cloud computing platform offers numerous applications and resources such as data storage, databases, and network building. However, efficient task scheduling is crucial for maximizing the overall execution time. In this study, workflows are used as datasets to compare scheduling algorithms, including Shortest Job First, First Come, First Served, (DVFS) and Energy Management Algorithms (EMA). To facilitate comparison, the number of virtual machines in the Visual Studio .Net framework environment is used for the implementation. The experimental findings indicate that increasing the number of virtual machines reduces Makespan. Moreover, the Energy Management Algorithm (EMA) outperforms Shortest Job First by 2.79% for the CyberShake process and surpasses the First Come, First Serve algorithm by 12.28%. Additionally, EMA produces 21.88% better results than both algorithms combined. For the Montage process, EMA performs 4.50% better than Shortest Job First and 25.75% superior to the First Come, First Serve policy. Finally, we ran simulations to determine the performance of the suggested mechanism and contrasted it with the widely used energy-efficient techniques. The simulation results demonstrate that the suggested structural design may successfully reduce the amount of data and give suitable scheduling to the cloud.

**INDEX TERMS** Energy management algorithm (EMA), first come first serve (DVFS), shortest job first (RR), makespan, VMs.

## I. INTRODUCTION

Cloud computing is an expanding industry at present. The ‘‘less is more’’ philosophy, which emphasizes acquiring more services while spending less, is becoming increasingly popular. These services are based on hardware, such as the CPU or I/O, or software, such as applications. To achieve minimal average wait times and the highest resource utilization, an efficient scheduling strategy is necessary. Resource allocation is done to Virtual Machines (VMs), and techniques like Energy Management Algorithm (EMA), Priority Scheduling, Shortest Job First (RR), and First Come, First Serve (DVFS) are the energy-efficient [1] versions of RC-GA, AMTS, and E-PAGA.

The use of scheduling techniques is prevalent in cloud computing systems, and their effectiveness has been evaluated through simulation using the cloudsim simulator. Among the investigated algorithms, it has been found that the EMA algorithm provides faster turnaround times and shorter average waiting times compared to others. Energy conservation [2] is critical in virtualized cloud computing systems due to its impact on operating costs, system efficiency, and

The associate editor coordinating the review of this manuscript and approving it for publication was Agostino Forestiero.

environmental protection. However, achieving energy efficiency while maintaining adequate [3] performance within a user-defined timeframe is challenging.

Describe four different kinds of clouds, including private, public, hybrid, and community clouds. The cloud types and their accessibility are shown in Figure 1. To meet the needs of consumers in every field, cloud computing offers a variety of service models. Infrastructure as a Service (IaaS) gives users access to a variety of virtualized computing resources, including RAM, CPU, OS, [4] and other application software. The more sophisticated cloud computing solution is known as PaaS (Platform as a solution). System software, as well as other computing resources, are provided, run, and maintained by it. The system can be installed with greater flexibility thanks to it. Operation and upkeep of operating systems, application software, and other resources are handled [5] through SaaS (Software as a Service). It functions as a user-application interface for web-based applications. It allows accessibility from any place with internet connectivity and eliminates concerns about infrastructure.

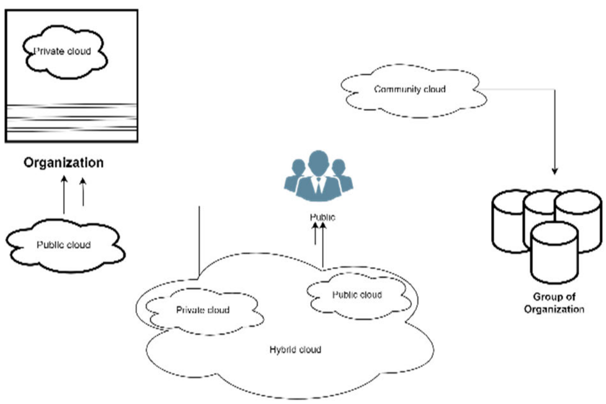



**FIGURE 1. Accessibility and types of cloud.**

The suggested method includes two stages: the first stage assists in achieving work deadlines and cutting down on execution time without taking energy consumption into account; task redistribution scheduling chooses the least energy-intensive site for execution while staying [6] within the deadline. Additionally, we recommend an energy-efficient mechanism for prioritizing tasks to achieve a favorable balance between energy conservation and job scheduling. The simulation findings demonstrate that in comparison to the recommended fix uses, the RC-GA, AMTS, and E-PAGA energy-efficient scheduling techniques are now in use. Aids in reducing considerable energy usage and improving [7] performance by 5%–20% while meeting deadline constraints.

The IoT and cloud computing convergence allows for the importation of data from physical sensors utilized by various applications deployed in various scenarios and subsequent use by all apps. It is past time to lower managed data centers' [8] enormous energy use. Discover and eliminate inefficiencies in the delivery of electric services to IT resources must be done right away. Making physical infrastructure more effective and using effective methods for managing and allocating resources can be accomplished. In the virtualized sensor cloud data center, this article focuses on managing the energy utilization of IT devices. This work is in the infrastructure of the virtualized sensor cloud. Places a strong emphasis on decreasing [9] energy use.

The pace of new technology development in the IT industry has accelerated since cloud computing was introduced. Users are growing quickly along with technology's development. While progress is accelerating, the environment is also regressing since it is deteriorating so quickly. When use rises, degradation becomes apparent but is first undetectable. At this point, research is done to reduce or eliminate the drawbacks. Similar to cloud computing, which has received an embrace from every second person in the thousands, green cloud computing has chosen to stop or decrease environmental destruction. Due to its high water, energy, and pollutant production, the computer sector is a factor in global warming. To reduce power use, the use of green computing, [10] carbon dioxide emissions, and energy usage is being implemented.

These days, the market's use of cloud computing is for a variety of reasons. It is a fantastic replacement for conventional computing and aids in the competitiveness of enterprises. A new method of managing, storing, and dispersing data is called cloud computing. In the world of technology today, it is a very popular trend. Many individuals are still unfamiliar with this technology. A list of [11] cloud computing services will be provided, along with an explanation of what may be expected from each. The most optimum and effective Azure cloud computing methods are what we're trying to identify. Business expenses will be reduced as a result. We observed a considerable improvement in our services' effectiveness and quality of service (QoS) after making the changes to the database algorithm. By doing this, businesses may reduce [12] costs and improve productivity.

Making the cloud is among the difficult and significant jobs involved in cloud computing usable for our purposes by putting various requirements into practice to satisfy the current and growing demands, consume as little energy as possible, and guarantee proper resource usage. It is necessary to put into practice the great mapping technique, sometimes referred to as virtual machine (VM) positioning, that has been developed to map physical [13] machines (PMs) against virtual machines (VMs). The enormous variety of the cloud's abundance of computing power, workloads, and virtualization operations makes consolidation an increasingly challenging, time-consuming, and problematic procedure. An algorithm suggested in the article would consume less energy and allocate resources more efficiently.

Using a Cloud System Model that allows for the mapping of VMs and PMs as well as between VM activities, this approach was created. Additionally, this algorithm's technique encourages reducing the number of PMs that are actively processing jobs and optimizes the overall processing time [14] for a group of Tasks per minute (also called Makespan time). We tested and rated the energy usage and Makespan period when using the Cloud Sim Simulator program. The data is compiled and then graphically compared to other energy-efficient VM placement algorithms that are already in use.

One of the most widely utilized infrastructures for conducting operations utilizing processing units, commonly referred to as virtual machines, is cloud technology. Task planning is one of the most significant inherent problems with cloud computing. Many methods to handle the Problem of non-deterministic optimization in polynomial time (NP) of determining the best [15] scheduling criteria in cloud computing have already been put out by academics. The algorithm known as one of these is the Heterogeneous Earliest Finish Time (HEFT). These techniques are known to give ideal results for job scheduling in a complex setting in a shorter amount of time. According to published research, HEFT [16] produces outstanding outcomes in respect of the timeliness and quality of the timetable.

The first vacant space is chosen; however, in some circumstances, this may not result in a desirable solution. To achieve better outcomes, we here provide updated variants of the HEFT method. We use several methods in the rank construction process for computing ranks, and we change how idle slots are chosen for task scheduling in the processor selection phase. To reduce how long a [17] specific workflow submission lasts on virtual machines, this study proposes improved variants of the HEFT algorithm with user-specified budgetary constraints. Moreover, according to our research, improved HEFT algorithms outperform the standard HEFT algorithm in terms of shorter schedule lengths for running on several virtual machines are workflow issues.

### A. RESEARCH CONTRIBUTIONS

- Here are some contributions we've made to our work:
- This study suggests the user first sends the data center the tasks that require additional steps. The tasks are obtained and sent to the scheduler from a data center is a designated space inside a structure utilized by users to organize, process, and store data.
- We compared three scheduling algorithms based on Makespan: DVFS, Shortest Job First, and the Energy management algorithm (EMA). Using the CyberShake and Montage processes, we ran the algorithms in the Visual Studio Net framework.
- To determine which approach is best for makespan optimization, we computed the results by increasing the number of virtual machines (VMs) from (6 to 20) and executing the chosen algorithms for both workflows.

Organization: The remaining sections are arranged as follows: The related research from various studies is presented in Section II. Algorithms for scheduling are presented architecturally and practically in Section III. Section IV presents the experimental findings, Section V presents the Discussion, and Section VI emphasizes the conclusion and unfinished business.

## II. RELATED WORK

A growingly significant and well-liked Cloud computing is a technology that makes on-demand (as required) possible almost real-time resource provisioning and releases a relatively new paradigm for employing remote computing resources. Task scheduling is one of the most difficult issues in this area, which plays a significant role in cloud computing. As a result, an efficient and reliable job allocation (scheduling) mechanism [18] is needed to establish more effective resource employment. Overall effectiveness, service caliber, and client experience can all be enhanced by employing an effective job scheduling algorithm. There is a large search space as a result of the relationship between job volume and problem complexity.

The category of NP-hard optimization problems includes problems of this type. The goal of this research is to propose a strategy that can reduce the search time and discover a rough (near-optimal) solution to scheduling multi-objective tasks in a [19] cloud environment. In the recommended book, we provide a swarm-intelligence-based method for multi-objective task scheduling called the hybridized bat algorithm. We tested the Cloud Sim toolkit using both real and made-up simultaneous workloads. The resulting results are compared with those of other comparable, metaheuristic-based techniques that were evaluated in the same way. The likelihood of our proposed technique in this field is strongly supported by simulation findings.

Virtual machine-based central cloud facilities have various advantages for lowering scheduling [20] costs and enhancing service accessibility and availability. Because of the integration of online services and security measures, cloud computing is a realistic option. The source and target domains in the task transfer have different feature spaces. Network traffic, which delays data transfers and prevents some important procedures from being completed on time, makes this situation more challenging. This study suggests an effective task-scheduling optimization technique based on a hybrid multi-verse optimizer with a genetic algorithm known as MVO-GA. Based on the burden of cloud resources, the suggested MVO-GA [21] is intended for the effectiveness of job transfers across the cloud network and needs to be improved.

Rescheduling the transfer tasks based on the efficiency weight of the total number of cloud tasks, sufficient transfer judgments must be made. The suggested approach (MVO-GA) takes into account many cloud resource characteristics, including speed, capacity, job size, task count, virtual machine count, and throughput. The suggested [22] approach successfully improves the task scheduling of several tasks (i.e., 100–1200). The efficiency of the suggested MVO-GA is demonstrated by the encouraging results it achieved in reducing the transfer time for large cloud tasks. Utilizing a [23] cloud simulation environment with a MATLAB-distrusted system, the proposed method is assessed.

Individual customers can access a wide range of services using cloud computing, from small to large organizations. The benefits of cloud computing tempt users to switch from traditional platforms to cloud platforms for their operational needs. Cloud computing offers a processing capacity that is far superior to conventional systems. In the cloud, resource requests are viewed as tasks, and the relevant resources are assigned based on user requirements. The cloud, however, finds it difficult to distribute resources due to the number and high demand for queries. In cloud computing, task schedulers are used to [24] and [25] overcome these problems. Numerous research publications have given various job scheduling techniques a better scheduling paradigm is still being sought after. This piece offers a hybrid optimization-based task scheduling system that efficiently schedules tasks with the least amount of waiting time. Other metrics addressed in this study include the total amount of time spent on production, execution, waiting, efficiency, and utilization. The simulation outcomes show that the suggested scheduling approach is effective and outperforms traditional scheduling algorithms based on optimization for ant colonies and particle swarms.

The distinctive qualities of cloud computing, including scalability, elasticity, on-demand service, and security, make it immensely popular. In a cloud system, many operations are carried out concurrently; hence a good task scheduler is required to improve [26] the effectiveness of the cloud system. Customer requirements for Quality of Service (QoS) elements should be a job scheduling algorithm taken into account when determining the order in which jobs are to be completed (as in the execution period and cost). Since it lowers costs and satisfies the prerequisite for green computing, the primary concern with energy strategy is contemporary task scheduling. This study's main objective is to compare and contrast 67 scheduling techniques used utilizing a cloud-based scheduling system to save energy. Given the issues and constraints that are now present, this study enables choosing the best energy scheduling algorithm optimization for the reader. The algorithms further split task scheduling using heuristics and meta-heuristics and other task scheduling into three techniques. Finally, prospective research directions and new advancements in this subject [27] are offered, along with a description of the benefits and drawbacks of the recommended algorithms.

Global energy demand has increased dramatically due to during the past ten years because of rapid urbanization, population growth, and technological improvements. Higher distributed energy system integration into traditional electrical grids is encouraged by the mitigation of environmental effects and socioeconomic benefits related to renewable energy systems. The intermittent and unpredictable nature of the energy management problem is, however, greatly exacerbated by the increase in the production of renewable energy. As a result, achieving a high level of system reliability and operational efficiency demands the application of an ideal energy management plan. As in the article, the various optimization strategies utilized to handle the energy management issues in microgrids are reviewed in a state-of-the-art, systematic manner. This paper identifies and provides a critical analysis of the many optimization strategies applied to energy management issues, with a focus on forecasting, demand management, economic dispatch, and unit commitment. Inferences from the review suggested that mixed integer programming approaches [28] were frequently adopted, given their effectiveness in resolving the energy management issue in microgrids and their ease of use.

Due to the decentralized character of the EMS problem in microgrids and the potential of these strategies to operate effectively in such circumstances, the effectiveness of the system is improved by multi-agent-based techniques and meta-heuristics algorithms over other conventional ways. Furthermore, it was clear that forecasting and demand management [29] were not the only applications for advanced optimization approaches. Arguing for the necessity of more precise scheduling and forecasting algorithms to deal with the microgrids' energy management issue. The necessity for Tran's active/collaborative energy-sharing feature in a community microgrid is a microgrid system with a complete energy management solution is described. The major problem with cloud computing is task scheduling. The aforementioned papers proposed numerous novel algorithms and methods for addressing the work scheduling Problem. Energy use is a significant issue as well. Some studies compare different algorithms to determine which one provides the greatest energy management method. The execution time for all tasks should be optimized while also lowering the cost and energy consumption when they are scheduled in the cloud. We compare many [30] algorithms in our work, including DVFS, Shortest Job First, and an energy management algorithm based on Makespan. For the workflows, CyberShake and Montage, we seek to identify the best method for makespan optimization. A comparison of scheduling criteria-based algorithms is shown in Table 1.

## III. PROPOSED EMA FRAMEWORK

1. The user first sends the data center the tasks that require additional steps.

2. The tasks are obtained and sent to the scheduler from a data center is a designated space inside a structure utilized by users to organize, process, and store data.

3. A scheduler is a piece of software that looks for the most effective way to arrange the work. It then gives the dis-patcher the duties.

4. The process that the Central Processing Unit (CPU) is assigned to is the scheduler chosen by the dispatcher, which is a module. The task is subsequently forwarded to the VM management.

5. To distribute the duties to the hosts, VM management is installed on the server hardware.

6. The resources that have virtual computers to carry out tasks are called hosts, which operate systems that run on

**TABLE 1. Algorithm comparisons based on scheduling criteria.**

| Authors | Algorithm | Environment | Scheduling factor | Objective criteria |
|---|---|---|---|---|
| Garg et al.[3] | Shortest Job First | Cloud Computing | Time | Response time |
| Sahoo et al.[4] | Genetic Algorithm | Cloud Computing | Cost | Makespan |
| S. A. Ali et al.[31] | First come, first serve | Cloud/grid computing | Time | Energy efficiency |
| Anand et al.[28] | Generalized priority algorithm | Cloud Computing | Cost | Resource allocation |
| Jeba et al.[25] | An optimal model for priority-based service scheduling policy for a cloud computing environment | Cloud environment | An array of the workflow instance | Quality of service, service request time |
| A. Ali et al.[15] | Reliable scheduling distributed in cloud computing (RSDC) | Cloud environment | Grouped task | Processing time |
| Moura et al.[22] | Improved cost-based algorithm for task scheduling algorithm | Cloud environment | Unscheduled task group | Cost and performance |
| Petrucci et al.[10] | A priority-based job scheduling algorithm in cloud computing | Cloud environment | A collection of work queues | Priority to each queue |

visitors. The distribution of tasks among VMs is shown in Figure 2.

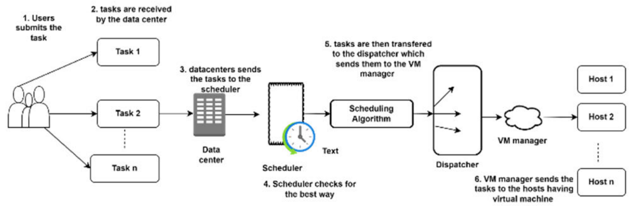



**FIGURE 2. Allocation of jobs to the VMs.**

### A. PROPOSED EMA (AN ENERGY MANAGEMENT ALGORITHM) ALGORITHM

#### 1) ALGORITHM

We have used several algorithms, including First Come, First Serve (DVFS), Shortest Job First (RR), and SJF (Shortest Job First), in our work. Below is a list of definitions for the algorithms.

#### 2) SHORTEST JOB FIRST (RR)

Processes from the ready queue are performed in a first-in, first-out fashion during round-robin scheduling and are allotted a finite amount of CPU power or quantum. When a procedure runs out of CPU time allocated for it, it is stopped and turned over to the next waiting process. Thereafter, the preempted process is reordered so that it is at the bottom of the ready list.

Operating systems employ the Shortest Job First algorithm as a scheduling technique to effectively distribute CPU time across numerous tasks. Each process in a queue is given a set amount of time, or "time quantum," to complete its task. A process is pushed to the back of the queue after finishing its time quantum, and the next process in line is allocated the CPU. Until each process has finished running, this process is repeated. You would need to make a queue to house the processes and a variable to keep track of the time quantum to execute the Shortest Job First algorithm. Additionally, you would need the means to monitor each process' progress and push them to the back of the line when their time quantum is up. You might also require a method to keep track of the overall amount of time each process has been sitting in the queue.

The Shortest Job First method can be implemented using the following steps:

1. Set up a queue to keep the processes waiting.
2. Configure the time quantum.
3. Although the line is not empty

a. Take the procedure after it at the head of the line.
b. Allocate the CPU to the process for one-time quantum.
c. If the process finishes running, take it out of the queue.
d. Put the process at the end of the queue if it doesn't finish running.
g. Extend the process' overall wait time.

4. Determine the overall average wait time for each process.

It's crucial to remember that the algorithm's efficiency greatly depends on the size of the time quantum. A big-time quantum can result in the system being slow to respond, and a small-time quantum can result in the overhead of switching operations being too high. It's also crucial to keep in mind that the Shortest Job First algorithm is a pre-emptive scheduling method, which means that when the time quantum runs out, the process will be stopped and put to the back of the queue.

**Algorithm 1** *Round Robin*

```
Input: BT = Burst Time
       AT = Arrival Time
       NT = Number of Task
       TQ = Time Quantum
Output: ST = StartTime
       CT = Completion Time
       TAT = Turn Around Time
       Wt = Waiting Time
       RT = Response Time
While RQ ≠ Null do
       ∑Tn/NT              //RQ=Ready Queue
end
for i = 1 to NT do
   if BT < TQ then
       TQ → BT
       BT → TQ
       send the task to waiting list
       i = i + +
else
       BT → TQ
       send the task to waiting list
endif
end
if waiting list ≠ empty then
       send tasks from waiting list to ready queue
       goto 1
else
   finish
endif
```

### 3) FIRST COME, FIRST SERVE (DVFS)

Processes on First Come and First Serve are carried out according to the time of their arrival. The largest duty must wait until the shortest job at the back of the line is finished.

Operating systems use the First Come; First Served (DVFS) algorithm as a scheduling technique to distribute CPU time across numerous processes. It operates by carrying out processes in the order that they enter the system. You would need to construct a queue to store the processes to implement the DVFS algorithm. As the processes enter the system, they are added to a queue, and they are then executed in the order that they are added to the queue.

The steps you can take to implement the DVFS algorithm are as follows:

First come, first serve

Input: ti = task to be scheduled

Output: $r_{list}$ = sorted list

For i = 1 to ti do

$r_{list} \leftarrow$ sort (R(ti), $b_{w(r)}$ ) // R(ti) is the ready queue and $b_w$ is the born time of each

Task

End

Return $r_{list}$

1. Set up a queue to keep the processes waiting.
2. Although the line is not empty

a. Take the procedure after it at the head of the line.
b. Provide the CPU to the process.
c. Watch for the procedure to finish running.
d. Discard the procedure from the waiting list.
a. Include the process's waiting time.

3. Determine the overall average wait time for each process.

It's crucial to keep in mind that DVFS is a non-preemptive scheduling method, which means that once a task begins running, it is not stopped until it has finished. Therefore, if a process enters the system after another process has begun execution, it must wait for the earlier process to complete before continuing. One of the main drawbacks of the DVFS algorithm is that if other processes arrive before it and are using a lot of CPU time, it may cause a process to get trapped in the queue for a very long time. This issue is referred to as "starvation."

### 4) ENERGY MANAGEMENT ALGORITHM (EMA)

A scheduling strategy known as the "Energy Management Algorithm" chooses the process with the quickest execution time to run next. It is possible to start with the shortest task first, whether it is preemptive or not. The shortest task is initially regarded as ideal because of its simplicity. It also cuts down on the typical amount of time other procedures take to execute. Operating systems use the Energy Management Algorithm (EMA) algorithm as a scheduling technique to distribute CPU time among numerous activities. It operates by starting operations in the sequence in which they should finish. The procedure that takes the least amount of time to complete is carried out first, then the procedure that takes the next-smallest amount of time, and so forth. You would need to construct a queue to store the processes to implement the EMA algorithm. As the processes enter the system, they are added to a queue and processed in the order of their anticipated execution time.

The steps you can take to build the EMA algorithm are as follows:

1. Set up a queue to keep the processes waiting.
2. Although the line is not empty:

a. Sort the processes in the queue according to how long they should take to complete.

b. Take the subsequent process in the queue with the quickest execution time.
c. Provide the CPU to the process.
d. Watch for the process to finish running.
e. Discard the procedure from the waiting list.
f. Include the process's waiting time.
3. Determine the overall average wait time for each process.

Preemptive and non-preemptive implementations of the EMA algorithm are also possible. In the pre-emptive version, a new process will start executing if it arrives and has a shorter execution time than the process that is already running.

It's crucial to remember that one of the EMA algorithm's greatest difficulties is precisely calculating how long each step will take to complete. When the estimate is too high or too low, the algorithm may not be able to fully utilize the CPU or may cause a process [32] to become stuck in the queue for an extended period. Another issue is that if fresh, shorter processes keep coming in, this algorithm may result in a situation known as "starvation," where a process with a longer execution time may never have the chance to run.

```
Energy management Algorithm
  Input: CloudletList
  Output: SortedList, Makespan
  1. Define CloudletList
  2. TotalCloudlet = CloudletList.size ( )
        avgCloudlet = Cloudletlength / totalCloudlet
  3. CloudletList = tempList
  4. for i =1 to totalCloudlet
        smallerCloudlet = tempList.get(i)
        checkCloudlet = tempList
        if smallerCloudlength > checkCloudlet.Length
        smallerCloudlet = checkCoudlet
        sortedList = smallerCloudlet
        else sortedList = smallerCloudlet
        end if
        end for
  5. if tempList = Null
        get sortedList
        else goto step4
        end if
  6. for j = 1 to sortedList.size ( )
        if CloudletLength < avgCloudlet&CloudletLength <
VM[j].siz e
        CloudletList(i) -> VM[j]
        Else
        QoudletList(i) -> vm[j+ I]
        End if
  7. Calculate the execution time of each task
  8. Completion time = sume(execution time)
  9. Makespan = max (completion time)
  10. End
```

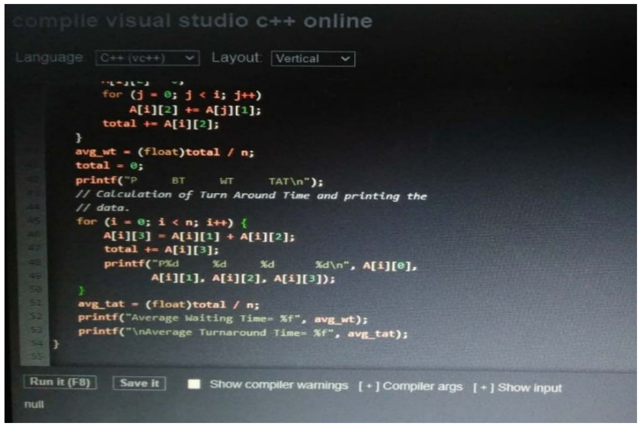



**FIGURE 3.** Visual studio 2013 net framework GUI.

### 5) SIMULATION SETUP

The resources we used in our work are listed below.

#### a: PROGRAMMING TOOL

Visual Studio 2013 Net framework was the development tool used in this model for the implementation of the algorithm. It enables the failure and overhead of heterogeneous systems and is an expanded version of Cloud Sim. It is mostly used to execute Visual Studio Net frame operations, which Cloud Sim does not support. Figure 3 displays the GUI for the Visual Studio Net framework.

#### b: RESOURCE MODELING

Resource modeling consists of the following components, as shown in Table 2.

**TABLE 2.** Configuration of the data center, host, and virtual machines.

| | Datacenter configurati on | Host configurati on | Virtual machine configuration | |
|---|---|---|---|---|
| | | | VM1 | VM2 |
| Cost of | 3.0 | - | - | - |
| using | 0.05 | 2048 | 512 | 512 |
| processin | 0.1 | 1000,000 | 10,00 | 10,00 |
| g | 0.1 | 10,000 | 0 | 0 |
| Memory | - | 2000 | 1000 | 1000 |
| (MB) | | | 1000 | 800 |
| Storage | | | | |
| (MB) | | | | |
| Bandwid | | | | |
| th (MB) | | | | |
| MIPS | | | | |

#### c: APPLICATION MODELING

In our work, we have used the following two workflows as a dataset:

- CyberShake (40, 60, 110, 1500)
- Montage (35, 65, 100, 1500)

#### 6) EVALUATION MATRIC

The ensuing parameter is taken into account when assessing the performance of the chosen algorithms.

##### *a: MAKESPAN*

It is the entire amount of time required by the resources to complete the execution of all tasks.

$$Makespan = \sum(finish(Ti)) - \sum(start(Ti)) \ldots\ldots\ldots [33]$$

## IV. RESULTS AND SIMULATIONS

Energy Management Algorithms (EMAs) are designed to optimize energy consumption in computer systems and other devices. When comparing EMA with two latest models instead of Shortest Job First and DVFS (First-Come, First-Serve), it's important to consider the context and the specific models you want to compare. Here, I'll provide a general comparison with two popular scheduling algorithms, which could be considered "latest models" in energy management:

Shortest Job First (SJF):

Objective: Minimize the energy consumption while maximizing system performance by prioritizing shorter jobs.

Comparison: EMA typically focuses on broader system-level energy optimization, considering variables like CPU load, memory utilization, and device power states. SJF, on the other hand, is a task-level scheduling algorithm that aims to reduce energy consumption by executing shorter jobs first, thus reducing idle times. The choice between EMA and SJF depends on the specific system and its workload characteristics.

Dynamic Voltage and Frequency Scaling (DVFS):

Objective: Optimize energy efficiency by dynamically adjusting the CPU's voltage and frequency according to the workload.

Comparison: EMA can incorporate DVFS as one of its strategies. DVFS is a power management technique that reduces the CPU's operating frequency and voltage when the system load is low, which can save energy. EMA, on the other hand, considers a broader range of system components and their interactions, not just the CPU. It can take into account factors like memory, storage, and network devices. Therefore, EMA can be seen as a more comprehensive approach to energy management.

In the comparison between EMA and these two models:

EMA is a holistic energy management approach, considering various aspects of system components and their interactions.

SJF focuses on minimizing energy consumption at the task level by prioritizing shorter jobs.

DVFS concentrates on optimizing energy efficiency by dynamically adjusting the CPU's voltage and frequency.

The choice between EMA, SJF, or DVFS depends on the specific requirements and characteristics of the system and workload. EMA is often used when a comprehensive energy management strategy is needed, whereas SJF and DVFS are more specific and task-oriented approaches.

The findings of the simulation evaluation of our suggested model in various experimental settings are covered in this section. The suggested model assesses various performance algorithms' points of view. The protocols DVFS and SJF, and EMA were contrasted in this study. The Visual Studio Net framework, an expanded version of cloudsim that allows large-scale scheduling, was used to run three algorithms, including DVFS, Shortest Job First, and the Energy management algorithm (EMA), in our experimental findings. By executing two processes from the Visual Studio Net Framework [34] (Montage and CyberShake) and increasing the number of VMs used, we were able to calculate the Makespan of each workflow (5–15). We utilized the formula given in to get the Makespan.

$$Makespan = \sum(finish(Ti)) - \sum(start(Ti))[33]$$

where finish (Ti) and start (Ti) are functions that, respectively, return the start time and completion time of the first and final tasks planned, the outcomes of processes employing the chosen algorithms—First Come, First Serve (DVFS), Shortest Job First, and Energy management algorithm (EMA) —are shown below. The number of virtual machines is plotted on the X-axis, and the Makespan is displayed on the Y-axis.

### *A. CYBERSHAKE WORKFLOW*

Algorithms like Dynamic Voltage and Frequency Scaling (DVFS), Shortest Job First, and Energy Management (EMA) are used to carry out the tasks (40, 60, 110, and 1500) in the CyberShake workflow. To determine which algorithm is optimum for makespan optimization, these algorithms are compared using the CyberShake process based on Makespan. We've [35] labeled the Y-axis makespan in seconds, and the X-axis is where we've taken the number of virtual machines.

The graphs for CyberShake 40, CyberShake 60, CyberShake 110, and CyberShake 1500, respectively, are shown in Figures 4a–c and d. With the addition of more VMs, the graph demonstrates how EMA has optimized the Makespan. Because more VMs are completing a given number of tasks, the Makespan gets shorter as the number of virtual machines increases [36]

The outcomes of the CyberShake workflow for the jobs (40, 60, 110, and 1500) are shown in Figure 5. The X-axis displays the number of virtual machines, and the Y-axis displays the Makespan in seconds. The increase in virtual machines can be shown to shorten the Makespan. This is because many VMs can complete a given number of tasks more quickly. Because of the parallel nature of the CyberShake workflow, the EMA algorithm can handle more tasks because it distributes them across the VMs based on task arrival time. EMA outscored DVFS by 2.79%, outperformed Shortest Job First by 12.28%, and outperformed SJF and DVFS by 21.88%. (Table 3).

#### 1) MONTAGE WORKFLOW

Using algorithms like First Come, First Serve (DVFS), Shortest Job First, [37] and Energy Management Algorithms,

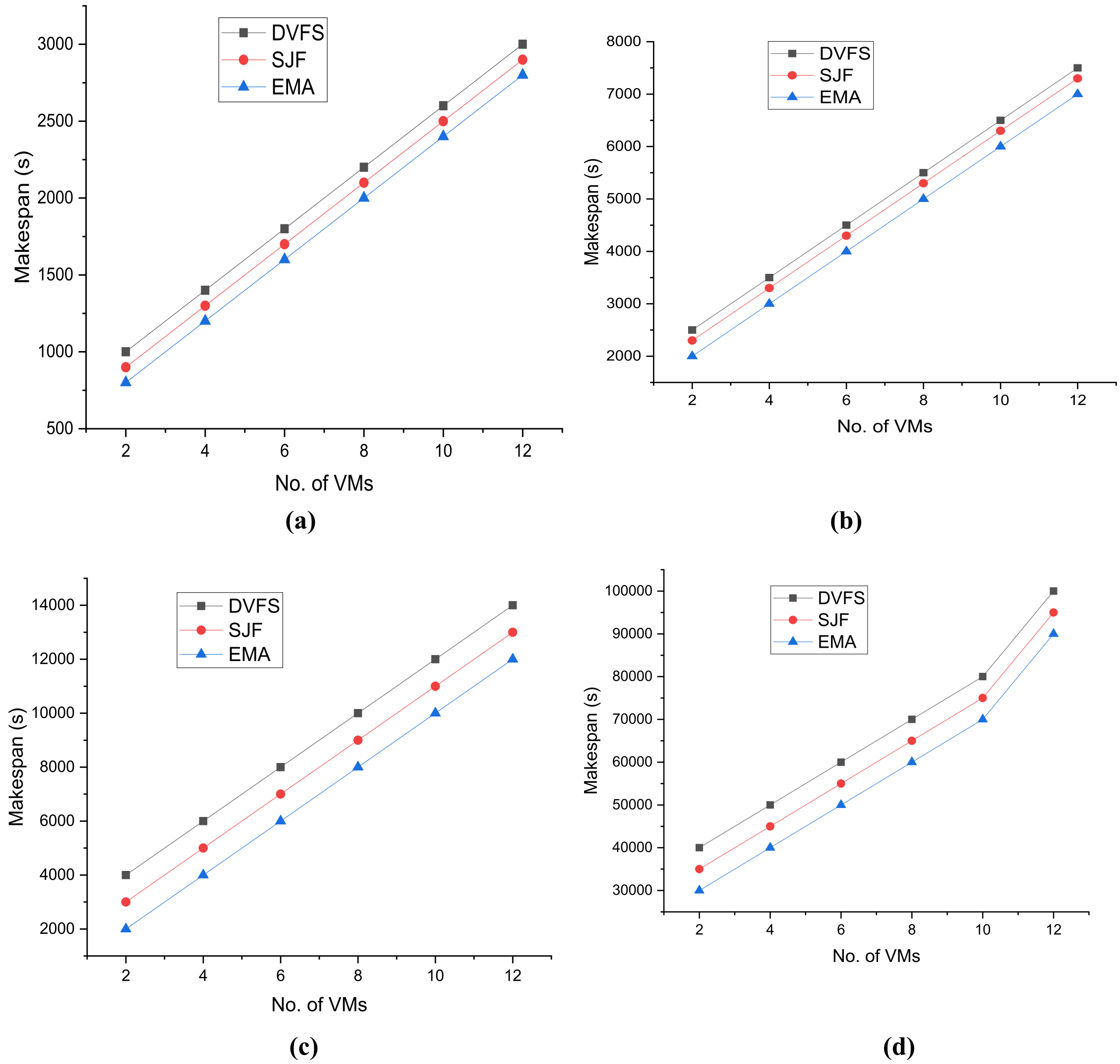



**FIGURE 4.** Makespan of (a) CybeShake_40 (b) CybeShake_60 (c) CybeShake_110 (d) CyberShake_1500.

**TABLE 3.** Results of cybershake and montage workflow.

| Workflow | No of tasks | EMA | DVFS | Shortest Job First |
|---|---|---|---|---|
| CyberShake | 40 | 18,804.7 | 18,536.7 | 20,178.3 |
| | 60 | 40,310.7 | 42,274.8 | 43,259.5 |
| | 110 | 36,964.6 | 43,390.3 | 65,880.9 |
| | 1500 | 551,192.3 | 579003.1 | 649872.1 |

Montage workflow with jobs (40, 60, 110, and 1500) is carried out (EMA). To compare various algorithms and determine the one that produces the least amount of Makespan, a Montage workflow is used. The number of virtual machines is on the X-axis, while the Makespan in seconds is denoted on the Y-axis.

The graphs for Montage 35, Montage 65, Montage 100, and Montage 1500 are shown in Figures 6a–d, respectively. By shortening the Makespan, max-min has produced the best outcomes for montage workflow. Here, we have increased the number of virtual machines (VMs) used to run the algorithms, and the Makespan decreases as the number of VMs increases since the burden is distributed over more virtual machines. The output of the Montage process for tasks (35, 65, 100, and 1500) is shown in Figure 7. The X-axis displays the number of virtual machines, and the Y-axis displays the Makespan in seconds. As the VMs advance, it is clear that the Makespan is decreasing. The pipeline structure of the montage approach is why max-min has worked successfully for this workflow. Max-min runs minor tasks concurrently while running the bigger jobs on the fastest machine. EMA has surpassed Shortest Job First by 17.73%, outperformed DVFS by 25.73%, and produced 4.63% better results than EMA.

To determine which algorithm performed the best overall, we also made a comparison between the CyberShake and Montage workflows. For the CyberShake and Montage procedures, respectively, Figures 8a and b compare the

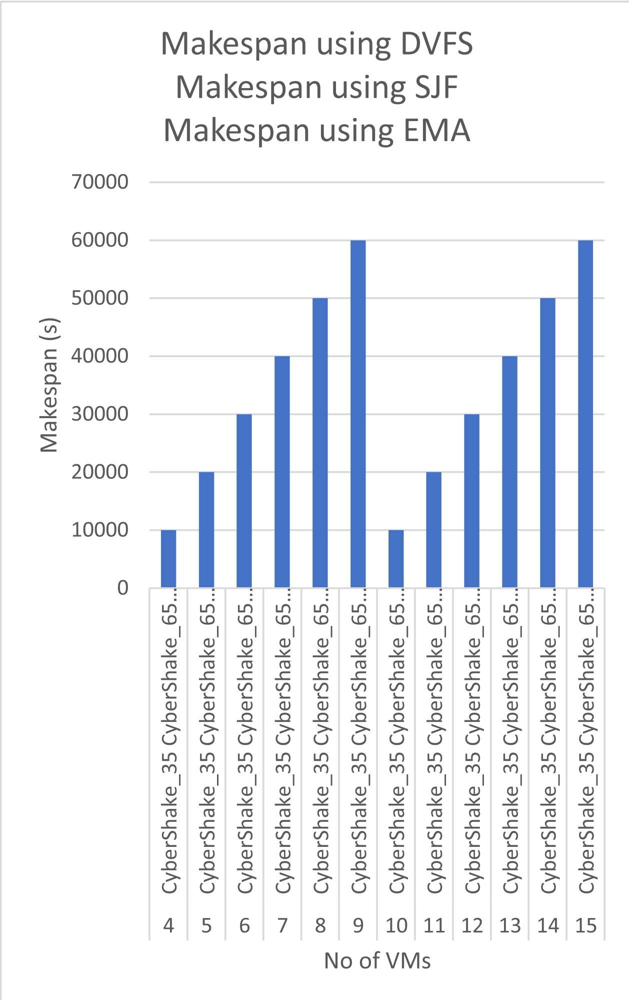



**FIGURE 5.** Makespan of cybershake workflow.

algorithms. By providing the shortest Makespan with the increase in [38] VMs for the CyberShake workflow, EMA has performed successfully. Due to the parallel nature of the CyberShake workflow, if there are more tasks than the number of VMs available, the EMA algorithm can still complete them. This is because it assigns the tasks to the VMs based on the tasks' arrival time. Because Montage's process is pipelined and involves minor jobs, the max-min algorithm has been successful in allocating these tasks to the quickest machines.

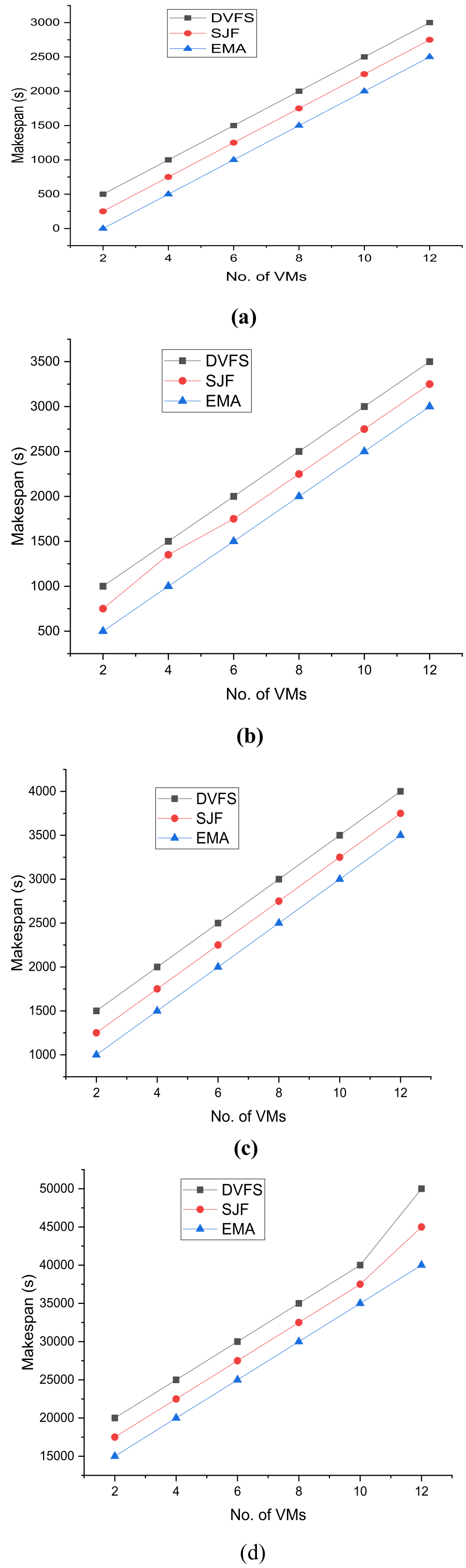



**FIGURE 6.** Makespan of (a) Montage_35 (b) Montage_65 (c) Montage_100 (d) Montage_1500.

## V. DISCUSSIONS

These days, the market's use of cloud computing is for a variety of reasons. It is a fantastic replacement for conventional computing and aids in the competitiveness of enterprises. A new method of managing, storing, and dispersing data is called cloud computing. In the world of technology today, it is a very popular trend. Many individuals are still unfamiliar with this technology. A list of cloud computing services will be provided, along with an explanation of what may be expected from each. The most optimum and effective [39] Azure cloud computing methods are what we're trying to

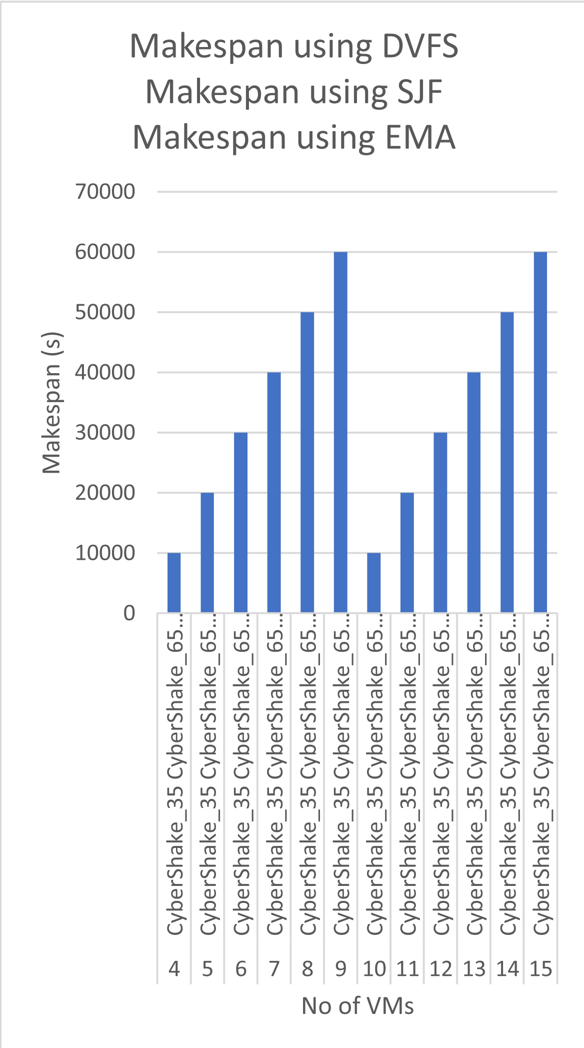



**FIGURE 7.** **Makespan of montage workflow.**

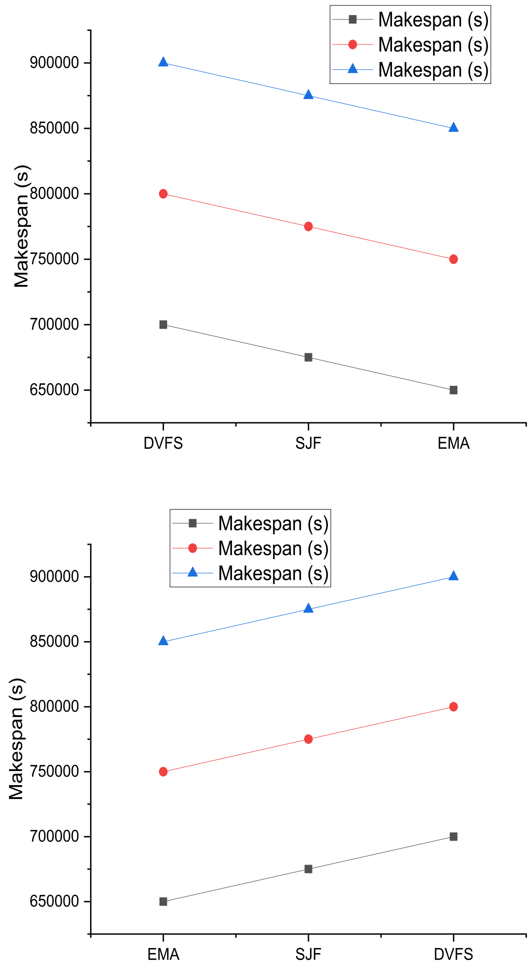



**FIGURE 8.** **(a) CyberShake workflow (b) Montage workflow.**

identify. Business expenses will be reduced as a result. We observed a considerable improvement in our services' effectiveness and quality of service (QoS) after making the changes to the database algorithm. By doing this, businesses may reduce costs and improve productivity.

In this study, a brand-new task-scheduling algorithm is introduced to outline a dynamic decision-based approach to handling energy usage and time execution that is energy-efficient. The suggested approach easily accommodates mobile device energy and time computations as well as cloud computing workloads. In addition, we offer a brand-new task scheduling server that offloads computing enhancing the mobile device's capacity for making decisions via the cloud [40] decisions improved computational efficiency when tasks are offloaded.

The task scheduling process uses the suggested empirical algorithm. Due to the efficient job scheduling made possible by this study's findings, energy consumption is greatly decreased. In heterogeneous cloud computing systems, the task scheduling issue of dependent tasks is addressed by a novel task prioritization strategy, and Techniques for task duplication are provided in this study. The original aspect is the introduction of a novel list scheduling technique, along with a novel way for work prioritizing and the use of [41] useful job duplication strategies.

Making the cloud is among the most difficult and crucial duties in cloud computing usable for our purposes by putting various requirements into practice, satisfying the current and growing demands, consuming as little energy as possible, and guaranteeing proper resource usage. It is necessary to put into practice the great mapping technique, sometimes referred to as virtual machine (VM) positioning, that has been developed to map physical machines (PMs) against [42] virtual machines (VMs). The enormous variety of the cloud's abundance of computing power, workloads, and virtualization operations makes consolidation an increasingly challenging, time-consuming, and problematic procedure.

An algorithm suggested in the article would consume less energy and allocate resources more efficiently. Using a Cloud System Model that allows for the mapping of VMs and PMs as well as between VM activities, this approach was created. Additionally, this algorithm's technique encourages reducing the number of PMs that are actively processing [43] jobs and optimizes the overall processing time for a group of Tasks per minute (also called Makespan time). We tested and rated the energy usage and Makespan period when using the Cloud Sim Simulator program. The data is compiled and then graphically compared to existing energy-saving VM placement methods used. The long-held goal of computing as a utility, known as cloud computing, has the power to completely change the IT sector by increasing the appeal of software as a service and influencing the design and procurement of IT hardware. The huge capital expenditures are no longer necessary for developers with creative concepts for new Internet services in hardware to set up their service or the cost of hiring people to run it. They don't have to worry about overprovisioning for a service whose demand turns out to be lower than expected, [44] squandering expensive resources, or under provisioning for one that takes off and loses out on prospective clients and income. In addition, since using 1000 servers for an hour costs no more than using one server for 1000 hours, businesses with huge batch-oriented operations can get results as fast as their programs can expand. This resource elasticity is unheard of in the history of IT, and it comes without paying a price for huge size. The world's most significant and interconnected technology is the Internet of Things (IoT), which is made up of sensor devices. The internet is seamlessly transitioning from an Internet of People to an Internet of Things, allowing different items to wirelessly link to one another. The IoT routing protocol's energy usage may have an impact on the network's lifespan. Furthermore, because tiny sensors are typically difficult to recharge after they are deployed, the massive volume of data created by IoT will lead to transmission collision, security vulnerabilities, and energy dissipation owing to increasing data redundancy. Data aggregation [45] often eliminates data redundancy at each node to conserve energy by putting some nodes in sleep mode and others in wake mode. Blockchain technology and the Internet of Things (IoT) have drawn a lot of attention lately from academics and businesses looking to develop a robust, safe, and secure communication platform. It can be difficult to decide how blockchain should be used in current IoT scenarios with the fewest possible ramifications. This paper proposes a message schedule for a blockchain-based architecture that separates incoming messages into non-critical and critical access levels. The researchers' proposed work splits the fog layer into two categories: blockchain fog clusters and action clusters. The action cluster [46] and the main cloud data centre collaborate for critical message requests, much like the three-layered IoT architecture.

## VI. CONCLUSION AND FUTURE WORK

In this study, we compared scheduling methods, including DVFS, Shortest Job First, and the Energy Management Algorithm (EMA) based on Makespan. CyberShake and Montage workflows are used to run the algorithms in the Visual Studio Net framework. The results are calculated by increasing the number of virtual machines (VMs) from (5 to 15) and then running the chosen algorithms for both processes to find the most effective technique for makespan optimization. The results of the experiment demonstrated that EMA performed well for the CyberShake process because of its parallelism and also because of the way it performs tasks, which decreases the Makespan by performing the tasks as they come in. Energy Management Algorithm (EMA) surpassed Shortest Job First by 2.79%, first come, first serve by 12.28%, and produced 21.88% better outcomes. Due to the pipeline layout of the Montage workflow, EMA has worked successfully for this workflow. EMA runs tiny tasks concurrently while doing the shortest task on the quickest machine. EMA outperformed DVFS by 25.73%, Shortest Job First by 17.73%, and both algorithms by 4.50%.

In the future, different models can be implemented to improve describing the accuracy of patterns used via malicious traffic in the cloud computing environment. To improve the accuracy of detecting known and unknown threats, different relevant features can be utilized. Furthermore, multiclass classification will apply to the algorithm.

## ACKNOWLEDGMENT

This research is supported via funding from Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2024R333), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

## REFERENCES

[1] A. Oukil, A. El-Bouri, and A. Emrouznejad, "Energy-aware job scheduling in a multi-objective production environment—An integrated DEA-OWA model," *Comput. Ind. Eng.*, vol. 168, Jun. 2022, Art. no. 108065.

[2] A. P. U. Siahaan, "Comparison analysis of CPU scheduling: FCFS, SJF and Round Robin," *Int. J. Eng. Dev. Res.*, vol. 4, no. 3, pp. 124–132, 2016.

[3] R. Sissodia, M. S. Rauthan, and V. Barthwal, "A multi-objective optimization scheduling method based on the genetic algorithm in cloud computing," *Int. J. Cloud Appl. Comput.*, vol. 12, no. 1, pp. 1–21, Jul. 2022.

[4] S. A. Ali, M. Affan, and M. Alam, "A study of efficient energy management techniques for cloud computing environment," in *Proc. 9th Int. Conf. Cloud Comput., Data Sci. Eng.*, Jan. 2019, pp. 13–18.

[5] A. Rashid and A. Chaturvedi, "Cloud computing characteristics and services a brief review," *Int. J. Comput. Sci. Eng.*, vol. 7, no. 2, pp. 421–426, Feb. 2019.

[6] A. Petrucci, G. Barone, A. Buonomano, and A. Athienitis, "Modelling of a multi-stage energy management control routine for energy demand forecasting, flexibility, and optimization of smart communities using a recurrent neural network," *Energy Convers. Manag.*, vol. 268, Sep. 2022, Art. no. 115995.

[7] A. Ali, M. M. Iqbal, H. Jamil, F. Qayyum, S. Jabbar, O. Cheikhrouhou, M. Baz, and F. Jamil, "An efficient dynamic-decision based task scheduler for task offloading optimization and energy management in mobile cloud computing," *Sensors*, vol. 21, no. 13, p. 4527, Jul. 2021.

[8] R. NoorianTalouki, M. Hosseini Shirvani, and H. Motameni, "A heuristic-based task scheduling algorithm for scientific workflows in heterogeneous cloud computing platforms," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 34, no. 8, pp. 4902–4913, Sep. 2022.

[9] R. Ghafari, F. H. Kabutarkhani, and N. Mansouri, "Task scheduling algorithms for energy optimization in cloud environment: A comprehensive review," *Cluster Comput.*, vol. 25, no. 2, pp. 1035–1093, Apr. 2022.

[10] M. Karuppasamy, M. Jansi Rani, and M. Prabha, "An efficient resource allocation mechanism using intelligent scheduling for managing energy in cloud computing infrastructure," in *Information and Communication Technology for Competitive Strategies*. Berlin, Germany: Springer, 2023, pp. 81–86.

[11] M. Andrei, P. Thollander, and A. Sannö, "Knowledge demands for energy management in manufacturing industry—A systematic literature review," *Renew. Sustain. Energy Rev.*, vol. 159, May 2022, Art. no. 112168.

[12] W. Shu, K. Cai, and N. N. Xiong, "Research on strong agile response task scheduling optimization enhancement with optimal resource usage in green cloud computing," *Future Gener. Comput. Syst.*, vol. 124, pp. 12–20, Nov. 2021.

[13] T. Kaur and A. Kumar, "Virtual migration in cloud computing: A survey," in *Proc. Int. Conf. Innov. Comput. Commun.*, 2022, pp. 785–796.

[14] N. Kortas and H. Youssef, "Performance impact of the MVMM algorithm for virtual machine migration in data centres," *Int. J. Grid Utility Comput.*, vol. 13, no. 4, pp. 333–346, 2022.

[15] W. Akram, K. Mahmood, X. Li, M. Sadiq, Z. Lv, and S. A. Chaudhry, "An energy-efficient and secure identity based RFID authentication scheme for vehicular cloud computing," *Comput. Netw.*, vol. 217, Nov. 2022, Art. no. 109335.

[16] B. Magotra, D. Malhotra, and A. K. Dogra, "Adaptive computational solutions to energy efficiency in cloud computing environment using VM consolidation," *Arch. Comput. Methods Eng.*, vol. 30, no. 3, pp. 1789–1818, Apr. 2023, doi: 10.1007/s11831-022-09852-2.

[17] H. Zhang, Y. Wu, and Z. Sun, "EHEFT-R: Multi-objective task scheduling scheme in cloud computing," *Complex Intell. Syst.*, vol. 8, no. 6, pp. 4475–4482, Dec. 2022.

[18] N. Garg, D. Singh, and M. S. Goraya, "Energy and resource efficient workflow scheduling in a virtualized cloud environment," *Cluster Comput.*, vol. 24, no. 2, pp. 767–797, Jun. 2021.

[19] S. Iftikhar, M. M. M. Ahmad, S. Tuli, D. Chowdhury, M. Xu, S. S. Gill, and S. Uhlig, "HunterPlus: AI based energy-efficient task scheduling for cloud–fog computing environments," *Internet Things*, vol. 21, Apr. 2023, Art. no. 100667.

[20] A. Katal, S. Dahiya, and T. Choudhury, "Energy efficiency in cloud computing data centers: A survey on software technologies," *Cluster Computing*, vol. 26, no. 3, pp. 1845–1875, 2022, doi: 10.1007/s10586-022-03713-0.

[21] R. Mandal, M. K. Mondal, S. Banerjee, G. Srivastava, W. Alnumay, U. Ghosh, and U. Biswas, "MECpVmS: An SLA aware energy-efficient virtual machine selection policy for green cloud computing," *Cluster Comput.*, vol. 26, no. 1, pp. 651–665, Feb. 2023.

[22] S. Sharma and C. Rana, "The scope of energy efficient techniques in cloud computing," in *Proc. 10th Int. Conf. Rel., INFOCOM Technol. Optim.*, Oct. 2022, pp. 1–4.

[23] H. Liu, P. Yao, S. Latif, S. Aslam, and N. Iqbal, "Impact of green financing, FinTech, and financial inclusion on energy efficiency," *Environ. Sci. Pollut. Res.*, vol. 29, no. 13, pp. 18955–18966, Mar. 2022, doi: 10.1007/s11356-021-16949-x.

[24] R. Choudhary and S. Perinpanayagam, "Applications of virtual machine using multi-objective optimization scheduling algorithm for improving CPU utilization and energy efficiency in cloud computing," *Energies*, vol. 15, no. 23, p. 9164, Dec. 2022, doi: 10.3390/en15239164.

[25] J. Zeng, D. Ding, K. Kang, H. Xie, and Q. Yin, "Adaptive DRL-based virtual machine consolidation in energy-efficient cloud data center," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 11, pp. 2991–3002, Nov. 2022.

[26] D. A. Amer, G. Attiya, I. Zeidan, and A. A. Nasr, "Elite learning Harris hawks optimizer for multi-objective task scheduling in cloud computing," *J. Supercomput.*, vol. 78, no. 2, pp. 2793–2818, Feb. 2022.

[27] S. Vinoth, H. L. Vemula, B. Haralayya, P. Mamgain, M. F. Hasan, and M. Naved, "Application of cloud computing in banking and e-commerce and related security threats," *Mater. Today, Proc.*, vol. 51, pp. 2172–2175, Jan. 2022.

[28] H. Wadhwa and R. Aron, "TRAM: Technique for resource allocation and management in fog computing environment," *J. Supercomput.*, vol. 78, no. 1, pp. 667–690, Jan. 2022, doi: 10.1007/s11227-021-03885-3.

[29] K. Siddesha, G. V. Jayaramaiah, and C. Singh, "A novel deep reinforcement learning scheme for task scheduling in cloud computing," *Cluster Comput.*, vol. 25, no. 6, pp. 4171–4188, Dec. 2022.

[30] R. Agrawal and S. Bansal, "Cloud computing: Security with educational usage," *J. ISMAC*, vol. 4, no. 3, pp. 165–173, Sep. 2022.

[31] Y. Mansouri, V. Prokhorenko, F. Ullah, and M. A. Babar, "Resource utilization of distributed databases in edge–cloud environment," *IEEE Internet Things J.*, vol. 10, no. 11, pp. 9423–9437, Jun. 2023.

[32] M. S. A. Khan and R. Santhosh, "Task scheduling in cloud computing using hybrid optimization algorithm," *Soft Comput.*, vol. 26, no. 23, pp. 13069–13079, Dec. 2022.

[33] N. Mansouri and R. Ghafari, "Cost-efficient task scheduling algorithm for reducing energy consumption and makespan of cloud computing," *Comput. Knowl. Eng.*, vol. 5, no. 1, pp. 1–12, 2022.

[34] M. Hussain, L.-F. Wei, A. Lakhan, S. Wali, S. Ali, and A. Hussain, "Energy and performance-efficient task scheduling in heterogeneous virtualized cloud computing," *Sustain. Comput., Informat. Syst.*, vol. 30, Jun. 2021, Art. no. 100517, doi: 10.1016/j.suscom.2021.100517.

[35] L. Hamid, A. Jadoon, and H. Asghar, "Comparative analysis of task level heuristic scheduling algorithms in cloud computing," *J. Supercomput.*, vol. 78, no. 11, pp. 12931–12949, Jul. 2022, doi: 10.1007/s11227-022-04382-x.

[36] R. Kaur and V. Laxmi, "Performance evaluation of task scheduling algorithms in virtual cloud environment to minimize makespan," *Int. J. Inf. Technol.*, vol. 14, no. 1, pp. 79–93, Feb. 2022.

[37] A. Javadpour, A. Nafei, F. Ja'fari, P. Pinto, W. Zhang, and A. K. Sangaiah, "An intelligent energy-efficient approach for managing IoE tasks in cloud platforms," *J. Ambient Intell. Humanized Comput.*, vol. 14, no. 4, pp. 3963–3979, Apr. 2023.

[38] L. Abualigah and M. Alkhrabsheh, "Amended hybrid multi-verse optimizer with genetic algorithm for solving task scheduling problem in cloud computing," *J. Supercomput.*, vol. 78, no. 1, pp. 740–765, Jan. 2022.

[39] B. M. P. Moura, G. B. Schneider, A. C. Yamin, H. Santos, R. H. S. Reiser, and B. Bedregal, "Interval-valued fuzzy logic approach for overloaded hosts in consolidation of virtual machines in cloud computing," *Fuzzy Sets Syst.*, vol. 446, pp. 144–166, Oct. 2022.

[40] A. S. Al-Haboobi, "Improving max-min scheduling algorithm for reducing the makespan of workflow execution in the cloud," *Int. J. Comput. Appl.*, vol. 975, p. 8887, Jan. 2022.

[41] K. Baghele, R. Barskar, and U. Chourasia, "Adoption of green cloud computing—A review," in *Proc. Int. Conf. Cogn. Intell. Comput.*, 2022, pp. 915–925.

[42] M. Gong, Y. Yoo, and S. Ahn, "Vehicular cloud forming and task scheduling for energy-efficient cooperative computing," *IEEE Access*, vol. 11, pp. 3858–3871, 2023.

[43] P. K. Bal, S. K. Mohapatra, T. K. Das, K. Srinivasan, and Y.-C. Hu, "A joint resource allocation, security with efficient task scheduling in cloud computing using hybrid machine learning techniques," *Sensors*, vol. 22, no. 3, p. 1242, Feb. 2022.

[44] M. Armbrust, A. Fox, and R. Griffith, "Above the clouds: A Berkeley view of cloud computing," Univ. California, Berkeley, CA, USA, Tech. Rep., 013, 2009, doi: 10.1145/1721654.1721672.

[45] A. Ahmed, S. Abdullah, M. Bukhsh, I. Ahmad, and Z. Mushtaq, "An energy-efficient data aggregation mechanism for IoT secured by blockchain," *IEEE Access*, vol. 10, pp. 11404–11419, 2022, doi: 10.1109/ACCESS.2022.3146295.

[46] I. Ahmad, S. Abdullah, M. Bukhsh, A. Ahmed, H. Arshad, and T. F. Khan, "Message scheduling in blockchain based IoT environment with additional fog broker layer," *IEEE Access*, vol. 10, pp. 97165–97182, 2022, doi: 10.1109/ACCESS.2022.3205592.

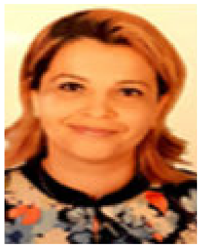

**ADEEL AHMED** received the M.S. degree in computer sciences from the Virtual University of Pakistan, and the Ph.D. degree in computer Science from The Islamia University of Bahawalpur, Punjab, Pakistan. His main research interests include edge computing, the IoT systems, energy efficiency, fuzzy logic, wireless networks, data aggregation in IoT, security, and blockchain. He serves as a reviewer for international journals and conferences.

**MUHAMMAD ADNAN** received the bachelor's and master's degrees in computer science from The Islamia University of Bahawalpur, Pakistan, where he is currently pursuing the M.S. degree. His main research interests include the IoT systems, cloud computing, and energy management algorithms.

**SAIMA ABDULLAH** received the Ph.D. degree from the Department of Computer Science and Electronic Engineering, University of Essex, U.K. She is currently an Assistant Professor with the Department of Computer Science and Information Technology, The Islamia University of Bahawalpur, Pakistan. She is a member of the Multimedia Research Group, DCS, where she has been involved in efficient and secure communication of multimedia data over future generation network technologies. Her main research interests include wireless networks and communications, future internet technology, and network performance analysis. She has authored around ten papers in the above research areas. She serves as a reviewer for international journals.

**ISRAR AHMAD** received the bachelor's and master's degrees in computer science from the Virtual University of Pakistan. He is currently pursuing the Ph.D. degree with The Islamia University of Bahawalpur, Pakistan. His main research interests include the IoT systems, fog computing, high availability, and blockchain.

**NAZIK ALTURKI** is currently an Esteemed Researcher affiliated with the Department of Information Systems, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia. With a deep passion for advancing the field of information systems, she has made significant contributions in areas, such as data management, decision support systems, and information security. Her expertise lies in the application of data analytics and machine learning techniques to extract meaningful insights from large datasets. She is also dedicated to exploring the challenges and opportunities presented by emerging technologies, including cloud computing, the Internet of Things (IoT), and blockchain. Her work has been published in prestigious journals and presented at international conferences, solidifying her reputation as a respected scholar. Furthermore, she actively participates as a reviewer for reputable academic journals, ensuring the high quality and rigor of scientific publications. Driven by her passion for research and her commitment to academic excellence, she continues to contribute to the field of information systems, making valuable contributions that advance knowledge, and drive innovation. Her research interests include leveraging technology to enhance organizational processes and improve decision-making efficiency.



**LEILA JAMEL** received the Engineering degree in computer sciences and the Ph.D. degree in computer sciences and information systems. She was the Program Leader of the IS Program and the ABET and NCAAA Accreditation Committees, CCIS, Princess Nourah bint Abdulrahman University (PNU), Saudi Arabia. She was the HOD of Information Systems Security of the Premier Ministry of Tunisia. She is currently an Assistant Professor with the College of Computer and Information Sciences, PNU. She is a Researcher with the RIADI Laboratory, Tunisia. Her research interests include business process modeling, business process management/re-engineering and quality, context-awareness in business models, data sciences, ML, process mining, e-learning, and software engineering. She was a member of the Steering and Scientific Committees of the IEEE International Conference on Cloud Computing. She is a reviewer of many international journals and conferences.