Computers, Materials & Continua
DOI: 10.32604/cmc.2022.026547
Article

Tech Science Press

A New Reliable System For Managing Virtual Cloud Network

Samah Alshathri[1,*], Fatma M. Talaat[2] and Aida A. Nasr[3]

[1]Department of Information Technology, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Saudi Arabia
[2]Faculty of Artificial Intelligence, Kafrelsheikh University, Kafrelsheikh, Egypt
[3]Faculty of Computers and Informatics, Tanta University, Tanta, Egypt
*Corresponding Author: Samah Alshathri. Email: sealshathry@pnu.edu.sa

Received: 29 December 2021; Accepted: 04 March 2022

Abstract: Virtual cloud network (VCN) usage is popular today among large and small organizations due to its safety and money-saving. Moreover, it makes all resources in the company work as one unit. VCN also facilitates sharing of files and applications without effort. However, cloud providers face many issues in managing the VCN on cloud computing including these issues: Power consumption, network failures, and data availability. These issues often occur due to overloaded and unbalanced load tasks. In this paper, we propose a new automatic system to manage VCN for executing the workflow. The new system called Multi-User Hybrid Scheduling (MUSH) can solve running issues and save power during workflow execution. It consists of three phases: Initialization, virtual machine allocation, and task scheduling algorithms. The MUSH system focuses on the execution of the workflow with deadline constraints. Moreover, it considers the utilization of virtual machines. The new system can save makespan and increase the throughput of the execution operation.

Keywords: Virtual network; scheduling; reliability; VM allocation; cloud computing

1 Introduction

Cloud computing, often known as cloud technology, is a new technique of storing, retrieving, and processing data and information. A cloud is an online virtual storage space [1,2]. There is a continuous increase and development in the volume of digital data used over the Internet. According to [3], around 463 Exabyte of data will be created every day by 2025. In many ways, cloud computing varies from traditional IT architecture, such as on-premises or locally installed servers [4]. Cloud computing provides storage and server resources that are scalable and flexible. However, it is cost-effective in terms of expanding or enhancing storage and server resources based on changing needs, whether they are increasing or decreasing. There are three primary types of cloud computing as-a-service solutions, each of which provides you with a different level of management: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a service (SaaS) [5–7]. Virtual machines, virtual storage, servers, and other resources are provided as a service across the network in IaaS. IaaS is a highly scalable on-demand platform that can be scaled up or down depending on the type of workload. Cloud providers supply users with hardware and software tools as a service in PaaS. The SaaS approach for cloud computing distributes software as a service over the internet.

1.1 Issues in Cloud

Although the business world has recognized the growing importance of cloud computing, cloud computing research is still in its early stages. As a result of industrial expansion and progress, new concerns continue to emerge, and many issues remain unaddressed [8,9]. The following are some of the research difficulties and challenges:

i. Security Issue: It looks that saving our data or running our software on someone else's hard drive while using someone else's CPU is quite dangerous. Security issues such as data loss, phishing, and botnet pose a severe threat to an organization's data and software.

ii. Costing Issue: Cloud users must weigh the tradeoffs between computing, communication, and integration. While switching from one cloud model to another lowers infrastructure costs, it adds data communication costs. The cost of moving an organization's data from a community cloud to a public cloud will be higher, as will the cost per unit of computer resources consumed.

iii. SLA Issue: Consumers must ensure the quality, availability, reliability, and performance of the underlying resources while migrating their core business from one cloud model to another. The user must have assurance from the service provider that the service will be delivered. This assurance is given through Service Level Agreements (SLAs).

iv. Predictable and unpredictable workloads: In cloud computing, the CPU, storage, and network are all virtualized resources. The Virtual Machines (VMs) provide the user with a virtualized environment. These VMs are given a variety of varying workloads. Because of the increased demand for the application, the load on (VMs) may dramatically increase. These workloads can be divided into two categories: predictable and unpredictable workloads.

v. Homogenous and Heterogeneous Workload: There are two sorts of workloads in the cloud: homogenous and heterogeneous. Homogeneous workload refers to workloads with similar configurations (number of CPUs required, RAM, storage, and execution time) [10,11]. Different resource vendors manage and host heterogeneous workloads at the same level or at different levels of configuration.

vi. Flexibility: Flexibility in the cloud refers to how dynamically fluctuating resource requirements can be handled. As time passes, the need for resources may increase; these demands must be automatically detected by the cloud and met.

vii. Virtual Machine Migration: The virtual machine migration approach can be used to address insufficient resource handling in the cloud. To accommodate the resources, VMs can be transferred from one host to another using this strategy.

viii. Efficient Energy Allocation: Due to the processing of a huge number of computing resources, carbon emissions in cloud data centers are excessive. As a result, approaches that can cut carbon emissions while also being energy efficient should be used.

1.2 Task Scheduling

The term "cloud scheduling" refers to the technology that allows users to map jobs to a variety of virtual machines or assign virtual machines to use the resources available to meet their needs [12]. In cloud computing, scheduling strategies are used to improve system throughput and load balancing, save expenses, increase resource usage, save energy, and speed up processing [13]. CPU and memory availability are managed by scheduling; a smart scheduling plan maximizes resource use [14]. Task scheduling approaches are used to determine which tasks or activities should be accomplished in what sequence. Its primary focus is on matching user tasks to available resources. Virtualization [15] is the foundation for effective work scheduling. Good scheduling approaches reduce total job completion time, improve device load balancing, and maximize resource use [16]. Task scheduling can be classified into the following group of scheduling algorithms: Static, dynamic, workflow, cloud service, real-time, Heuristic, and opportunistic load balancing (OLB) scheduling algorithms [17].

To understand the scheduling problem, see Fig. 1. Firstly, users send tasks to the Cloud and then tasks must be assigned to the appropriate processor. It's now an issue of allocating processor duties in such a way that the cloud owner obtains the most advantage in the least amount of time [18]. The problem of jobs being assigned to the processor that takes into consideration the other element is solved by developing schedulers.

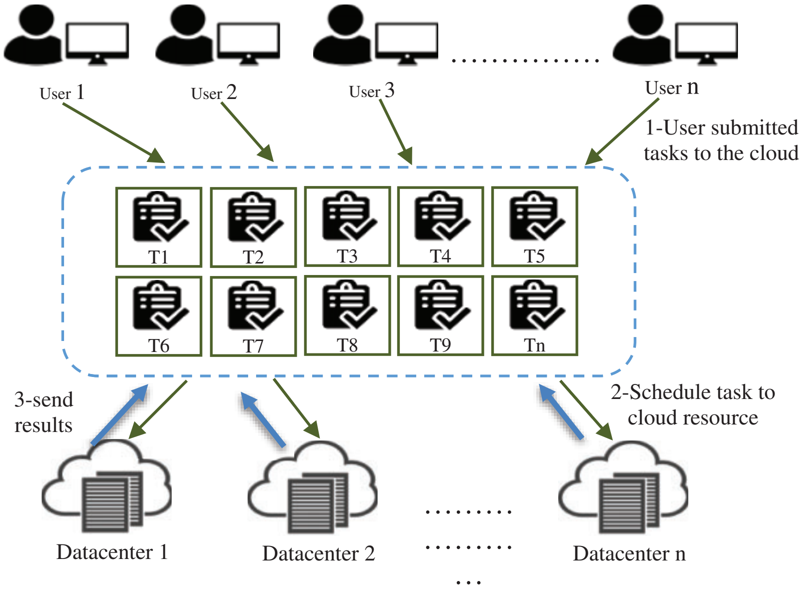


Figure 1: Task Scheduling in CC

1.3 VM Allocation

One of the issues in cloud computing environments, particularly for private cloud designs, is the allocation of virtual machines (VMs). Each virtual machine is mapped to the physical host in this environment based on the resources available on the host computer. Quantifying the performance of scheduling and allocation policies on a Cloud infrastructure for various application and service models with diverse performance metrics and system requirements is a particularly difficult topic to address.

1.4 Power Consumption Issues

One of the essential requirements of IoT of Things (IoT) applications and resources in the cloud is energy efficiency, which is a difficult issue in the Cloud of Things (CoT). One of the major issues is to efficiently use energy at various levels, such as in data centers that host cloud applications [19]. Higher generations of networks face a new level of complexity as a result of energy usage. Energy utilization refers to the ability to reduce energy use while maintaining a sufficient level of energy consumption.

1.5 How We Solve the Previously Mentioned Issues

In this paper, we aim to cloud systems in such a way that it can accommodate both types of workload. A cloud should have a well-functioning resource management system. We aim to handle the most influential issues in CC such as (i) Virtual Machine Migration, (ii) Efficient Energy Allocation, and (iii) Handling Long-Running Jobs.

2 Related Work

This section introduces some of the recent previous efforts in the field of task scheduling algorithms, power consumption, VM allocation, and summarize the advantages and the disadvantages of the most common previous scheduling algorithm in CC.

Authors in [20] proposed the algorithm's vocalization for humpback whale optimization (VWOA). The VWOA is a cloud system that simulates the vocalization behavior of humpback whales and is used to optimize work schedules. The VWOA scheduler is based on a proposed multi-objective model. It saves time, money, and energy while maximizing resource utilization. The VWOA scheduler outperformed the standard whale optimization algorithm (WOA) and round-robin (RR) algorithms in terms of cost, makespan, degree of imbalance, resource use, and energy consumption. To address the problem of multi-objective task scheduling, authors in [21] suggested an efficient hybrid approach for Distribution Estimation Algorithm and Genetic Algorithm (EDA-GA). The goal of EDA-GA is to reduce task execution time while also improving the environment's ability to balance the load. The proposed method first employs EDA operations to generate a suitable solution, then employs a genetic algorithm to generate additional solutions that are dependent on the optimal solution selected in the previous stage in order to broaden the scope of the search. The data show that the suggested method has a fast convergence rate and a strong search capability. The improved discrete particle swarm optimization technique (IDPSO) by Liu et al. [22] uses a sinusoidal strategy-based dynamic inertia weight optimization approach to make particles adaptive to different phases while searching for the best global solution. In terms of completion time and convergence, the IDPSO technique surpasses the DPSO and FCFS algorithms.

The authors of [23] propose combining dynamic fusion mission planning methodology, ant-colony, and the genetic approach. This reduces the amount of energy used by cloud computing data and storage facilities. The evaluation results show that the suggested task programming approach would significantly reduce cloud computing device time and energy usage. In [24], authors proposed an Efficient Resource Allocation with Score (ERAS) for task scheduling in cloud environments, which takes into account Virtual Machines (VMs) temporary operational availability by suggesting various types of delays and using EFT to set the processor for task scheduling to a standardized score. The results show that the improved dependability of the ERAS algorithm delivers superior efficiency than current systems that just consider EFT for allocations. Sanaj et al. [25] proposed an enhanced Round Robin (ERR) approach to boost efficiency without compromising RR functionality. The proposed technique is implemented and tested using the CloudSim toolbox. In comparison to classical RR, the results show that the overall waiting time for tasks in a certain number of cloudlets in ERR is minimized under identical conditions. To improve the Min-min method's efficiency, Shi et al. [26] devised the BMin algorithm. The proposed technique is evaluated using the cloudsim simulation program, and the results show that it minimizes completion time, maximizes throughput, and improves resource load balancing. Because of resource constraints, one of the major difficulties that IoT faces is power consumption. Combining IoT and cloud computing could result in lower energy use. However, there are still certain issues with the procedures that need to be addressed. The following summarizes the limitations and opportunities of the seven models for an energy efficient IoT in various cloud environments. The simplest model in [27] has saved more than 36% in power. In the other hand, it has low scalability and requires consideration and improvement in run time. The earlier model in [27] was improved by adding a passive optical access network (PON) to the system in [28]. However, PON may cause service interruptions. The best model, when compared to others, is in [29], where the model is suited for smart environments by decreasing 80 percent power and eliminating service delay, and this mechanism could also be utilized in a cloud computing test. In order to reduce computational complexity, researchers are looking into the benefits of distributed optimization techniques for enabling local reactions to fast framework dynamics. The most effective approach for many nodes is the design for green industrial in [30], where the relationship between saved power and the number of nodes is positive. Furthermore, the resource usage of this approach has improved. When using this concept in a smart environment, such as a smart city, this could be advantageous. For mobile cloud computing networks, the technique in [31] does not bring any additional power in terms of data privacy. To ensure that the model is acceptable for a rising number of devices, it should be tested on large scale networks. In the cloud context, the FEM Model [32] has delivered greater energy savings and resource fairness. To test availability, the method should be assessed in a distinct cloud environment. In [33], the E2C2 algorithm achieved electricity savings on the fewest number of services. To improve usability, this technique should be evaluated in a variety of application domains. E2C2 could be improved by ‘sorting’ cloud providers according to the total energy consumption of their respective services. According to Hamdi et al. [34], high energy usage has a negative impact on cost and the environment. They have emphasized VM consolidation strategies in several dimensions. They analyzed some key characteristics of VM consolidation techniques, such as the variable nature of resource requirements, and ideal physical machine power consumption. Wu et al. [35] proposed a power-aware scheduling algorithm that use the utilization threshold approach to pick host physical machines and the smallest utilization gap method to select virtual machines for the VM migration process. Hsieh et al. [36] took into account both current and future resource usage and forecasted future resource utilization more accurately. To make this forecast, they determined the host overload and under load detection. The Comparison of some of the previous common scheduling algorithms in Cloud Computing is shown in Tab. 1.

Table 1: Comparison of some of the previous scheduling algorithms in CC

Ref.	Year	Applied algorithm	Parameters	Tools	pros	Cons
[20]	2019	VWOA algorithm	makespan, cost, and energy and resource utilization	CloudSim	The suggested algorithm outperformed the WOA and RR algorithms in terms of makespan, cost, and degree of imbalance, energy consumption, and resource utilization.	High complexity

(Continued)

Table 1: Continued

Ref.	Year	Applied algorithm	Parameters	Tools	pros	Cons
[21]	2019	EDA-GA algorithm	Convergence, completion time, load balancing	CloudSim	The EDA-GA method improves load balancing by enhancing convergence, search capability, and job execution speed.	Can be trapped into local optima
[22]	2019	IDPSO algorithm	Completion time, converge	CloudSim	In terms of completion time and convergence, IDPSO outperforms both DPSO and FIFO.	It doesn't take into consideration important performance metrics like Average Resource Utilization (ARU) and Load Balancing Level (LBL)
[23]	2020	Combined GA and ACA.	Execution time, energy	CloudSim	The proposed solution minimizes the amount of time and total energy consumed by jobs in cloud computing systems.	It has a difficult theoretical analysis.
[24]	2020	ERAS algorithm	Makespan, Reliability	CloudSim	In comparison to the EFT method for allocation, the ERAS algorithm improves reliability while delivering superior results.	Time consuming
[25]	2020	ERR algorithm	Waiting time, execution time, and residue energy.	CloudSim	The overall waiting time for ERR tasks decreased in comparison to RR under the same conditions.	Complex, and it can be inaccurate
[26]	2020	BMin algorithm	Completion time, throughput, load balancing	CloudSim	When compared to Min-min, the completion time is shorter, and the load balance is better.	inaccurate

The Autonomic Resource Provisioning and Scheduling (ARPS) framework was designed and developed by the authors [37]. The ARPS framework provides the ability to schedule jobs at the optimal resources within the deadline, maximizing both execution time and cost. The spider monkey optimization (SMO) algorithm-based scheduling mechanism is also included in the ARPS architecture.

In [38], the authors propose a resource allocation model for efficiently processing applications as well as a Particle Swarm Optimization (PSO) based scheduling algorithm named PSO-COGENT that not only optimizes execution cost and time, but also reduces the energy consumption of cloud data centres, while keeping the deadline in mind. In [39], the authors give a thorough evaluation and classification of various scheduling approaches, as well as their benefits and drawbacks. We anticipate that our systematic and extensive survey effort will serve as a steppingstone for new cloud computing researchers and will aid in the continued development of scheduling techniques. Fog computing offers a variety of distributed computing models for scheduling in fog environments, such as those described in [40]. The authors of [41] investigated load balancing strategies and found that they are split into two categories: those that support task migration and those that do not support task migration throughout the load balancing process. Each category is further divided into subcategories based on grid resource topology, which covers both flat and hierarchical resource topologies. Based on distinct load balancing features and performance measures, we analyzed and evaluated a variety of dynamic load balancing solutions relevant to these categories. In [42], the authors suggest Cloud Light Weight (CLW), a new load balancing solution that not only balances the burden of virtual machines (VMs) in cloud computing datacenters but also ensures QoS for users. It cuts down on the number of VM migration operations as well as the time it takes to migrate during application execution.

3 Problem Definition

In this section, we describe user application workflow and the cloud model.

3.1 Workflow

The scheduling problem of multi-workflow is known as NP-complete problem [43]. There are two main classes of workflow scheduling: Static Scheduling and Dynamic Scheduling. Static scheduling algorithms use constant strategies which don't change during workflow scheduling. Whereas dynamic algorithms are changing during execution. Each type has cons and pons. Static algorithms are more stable and consistent. In addition, the results of the static algorithms are known previously. However, the results are often poor. With dynamic algorithms, the strategies are changing and being enhanced continuously. This results are in high time complexity. However, the results are better than the static algorithms. There are a set of issues that often occur due to the overloaded and unbalanced load problems [44].

We can represent the workflow applications using Directed Acyclic Graphs (DAG) G = (N, E, W, C). Where N is the set of nodes/tasks N = {n_1, n_2, n_3, . . . , n_n}, and E is the set of edges (i.e., dependences) between the nodes. For an example: Edge e(i,k) represents the precedence constraint between n_i and n_k. Where node n_i is the parent of n_k, and n_k is the child of n_i. Otherwise n_i should be executed before task n_k can be started. Node with no parents is named root and Node with no child is named leaf. Numbers on the edges represent the communication costs between the nodes. Fig. 2 shows an example of the workflow application. Tab. 2 shows the time taken to execute each task at each VM.

3.2 Virtual Network Model on Cloud Computing

In this paper, we consider cloud computing model consists of two layers: The manager layer and the virtual network layer. The manager layer is responsible for managing virtual network for executing user workflow with considering the constraints (see Section 4). It represents the MUSH system, while the virtual network is owned by the organization as a private cloud. Some of organizations preferred using virtual network than individual VMs for sharing resources like files and applications. This way save money and make the organization devices as a one unit.

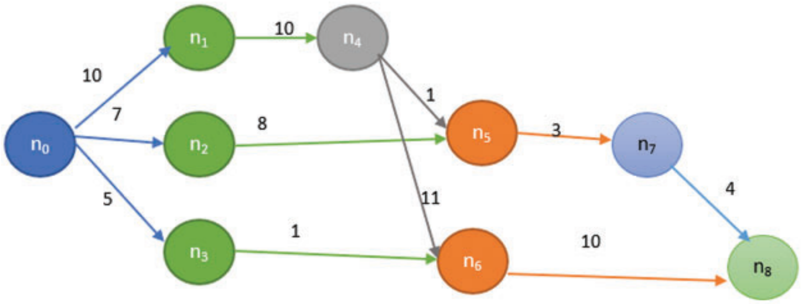


Figure 2: Example of workflow applications with execution weights

Table 2: Examples of set of nodes

Node	VM_1	VM_2	VM_3
n_1	60	30	22
n_2	50	25	10
n_3	100	55	20
n_4	70	40	28
n_5	100	60	40
n_6	50	25	14
n_7	90	47.5	25
n_8	30	17.5	11
n_9	40	27.5	20

Many cloud providers such as Microsoft and google provides this type of network to users and organizations. There are two classes of virtual network: Homogenous network and heterogeneous network. With homogeneous networks, users can choose VMs with equal specifications while heterogeneous networks consist of VMs with different specifications. In this paper, we use a heterogeneous virtual network. We schedule multi-workflow for different Users considering the deadline time and available resources. The scheduling model consists of four components: users, workflow queue, scheduler system, and cloud infrastructure. Fig. 3 shows the system model. Cloud users can submit their workflows for executing over the internet to the cloud. All submitted workflow are ordered on a queue according to the deadline of each workflow. The executed system is responsible for managing the cloud resources to execute workflows with considering workflow deadline, power consumption, and reliability. Cloud infrastructures are responsible for executing the workflows.

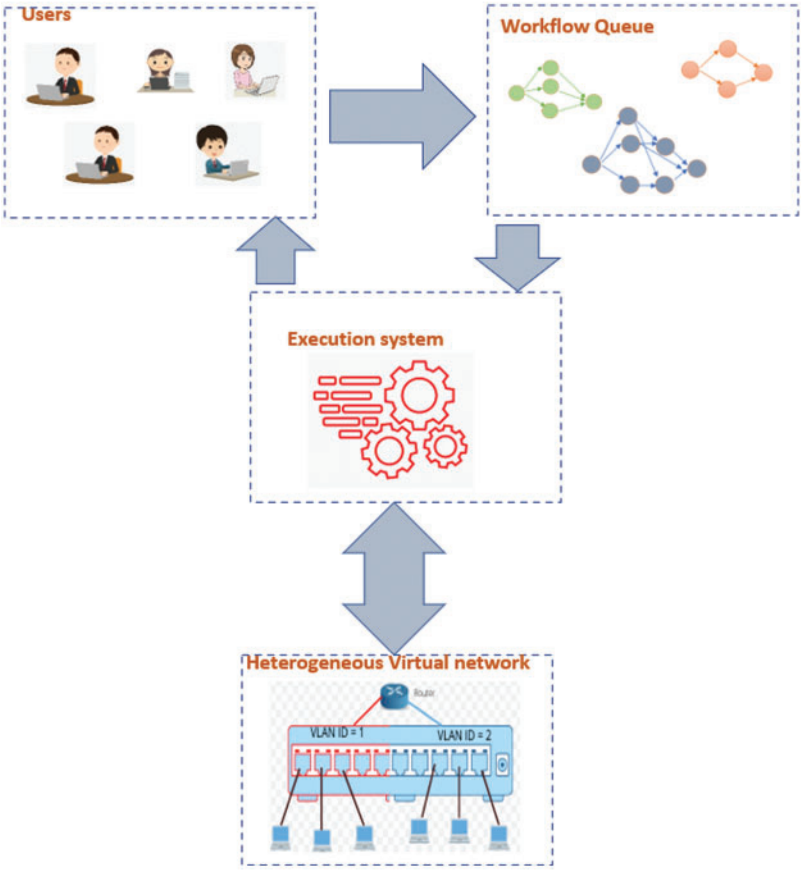


Figure 3: The system model

4 The Proposed Scheduling Framework

The proposed Multi-User Hybrid Scheduling (**MUHS**) framework considers the QoS and provider profits. This occurs by taking into account the QoS parameters, such as the makespan, deadline, cost, and the provider profits such as power consumption and utilization. There is no scheduling algorithm that considers the previous parameters at the same time. Thus, we developed the new MUSH framework to solve the trade-off between the user needs and the provider profits. The MUSH framework consists of three stages which are: **(i) Initialization stage**, **(ii) allocation stage**, and **(iii) power-consuming stage**. Fig. 4 shows the structure of the MUSH framework.

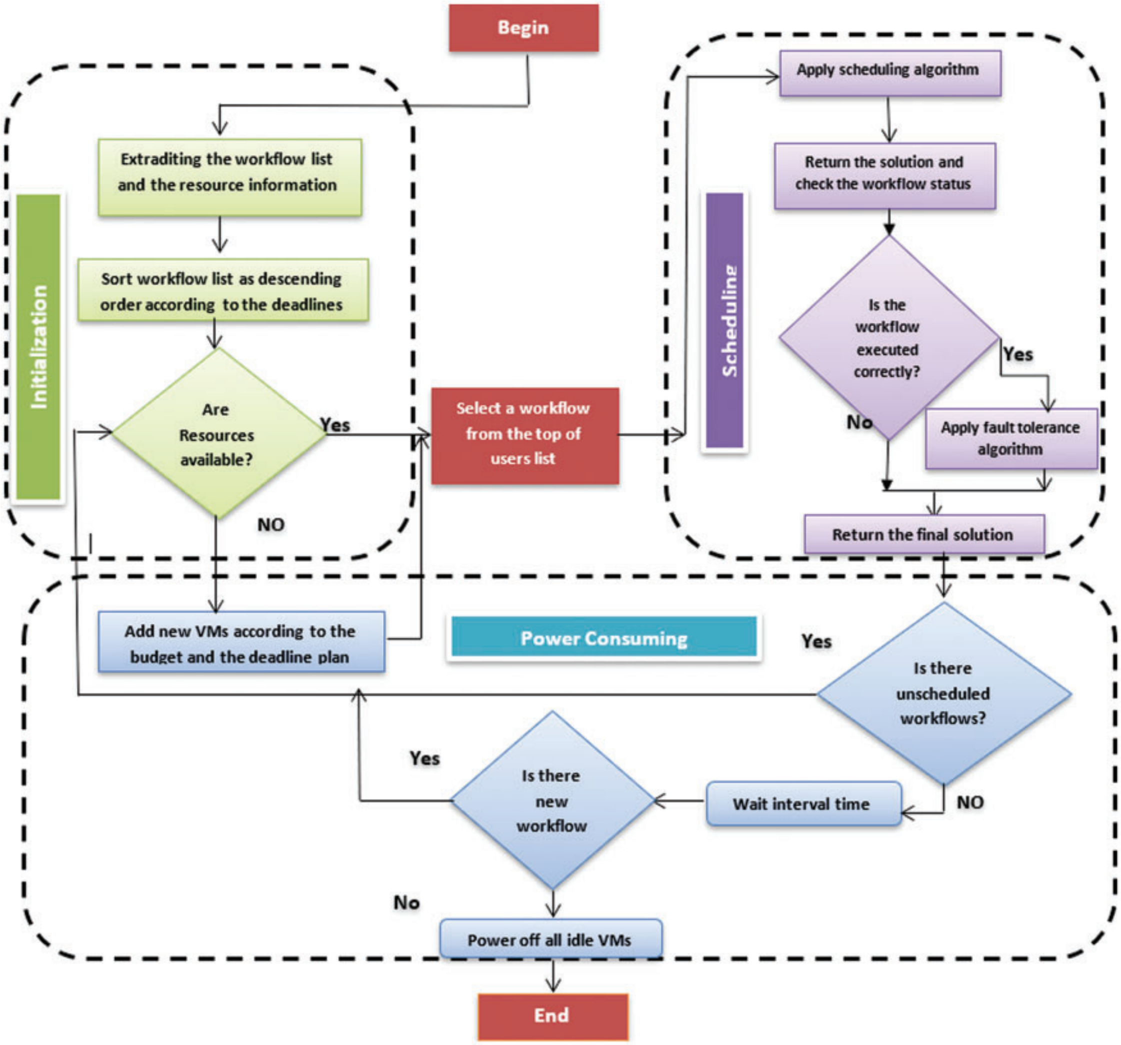

Begin
Initialization
Extraditing the workflow list and the resource information
Sort workflow list as descending order according to the deadlines
Are Resources available?
Yes
NO
Select a workflow from the top of users list
Scheduling
Apply scheduling algorithm
Return the solution and check the workflow status
Is the workflow executed correctly?
Yes
No
Apply fault tolerance algorithm
Return the final solution
Add new VMs according to the budget and the deadline plan
Power Consuming
Yes
Is there unscheduled workflows?
NO
Wait interval time
Yes
Is there new workflow
No
Power off all idle VMs
End

Figure 4: The structure of the MUSH framework

4.1 Initialization Stage

In this stage, the framework prepares information about the system before starting the allocation process. The MUSH framework gathers information about available resources and submitted workflows, such as the number of submitted workflows, the information about each workflow, the number of available VMs, and the deadline and the budget user plan. According to the deadline constraint, the MUSH framework sorts the workflow list by descending order. The initialization stage is changing continually during the running of the framework because it is used for the next stages: Allocation and power-consuming. Also, the MUSH workflow considers the initial user plan for defining the budget and the deadline of each submitted workflow. The MUSH workflow achieves the user needs and the provider profits according to the user plan. Firstly, the user should define the number of VMs of his IaaS model. After that he detects the maximum number of the rented VMs over specific duration. According to the maximum number of VMs (h), the provider can compute the maximum budget.

(MB) per hour as shown in Eq. (1):

$$MB = \sum_{v=1}^{h} C(VM_v) \tag{1}$$

where C value is the cost per hour for each VM. According to the deadline D of each submitted workflow, the provider adds and removes the VMs. The overall steps of the Initialization stage are shown in Algorithm 1:

Algorithm 1: Initialization Stage

```
Extraditing the workflow list and the available VMs
Sort the submitted workflow according to the user deadline by ascending order
If there are no enough VMs
    Apply the VM allocation stage
      End if
```

4.2 VM Allocation Stage

In this stage, the MUSH framework allocates a number of VMs to the organization to meet the deadline constraint and the maximum budget are shown in Algorithm 2. Firstly, the algorithm computes the Total Deadline (TD) by using Eq. (2):

$$TD = \sum_{j=1}^{W} D_j \tag{2}$$

where W is the number of submitted workflows. The algorithm also calculates the Expected Completion Time (ECT) of all workflows by using Eq. (3).

$$ECT = \left(Q * \sum_{j=1}^{W} \sum_{i=1}^{n} MI(t_i)\right) \Big/ \sum_{v=1}^{m} MIPS_v \tag{3}$$

where Q is the control parameter that is used to close the gap between the ECT and the real completion time, and $\left(\sum_{j=1}^{W} \sum_{i=1}^{n} MI(t_i)\right)$ is the total Million instructions (TMI) of all workflows for all tasks.

The algorithm checks if ECT <= TD. If the inequality is true, that means the available VMs will meet the deadlines constraint. If it is false, that means the system needs new VMs. The VM allocation algorithm defines the needed number of VMs (NV) according to Eq. (4).

$$NV = NC/VAC \tag{4}$$

where NC = TMI/TD is the needed capacity to meet the deadline, and VAC is the VM available capacity of the extra homogeneous VMs. We can compute the VAC by using Eq. (5).

$$VAC = \left(\sum_{j=1}^{W} \sum_{i=1}^{n} MI(t_i)\right) \Big/ TD \tag{5}$$

The overall steps of the VM Allocation stage are shown in Algorithm 2:

Algorithm 2: VM Allocation

```
Compute the total deadline TD of all submitted workflows
Compute ECT
If ECT $< =$ TD
    The resources are enough
Else
        Compute VAC
        Compute NV
        Create new VM equals the NV
            End if
```

4.3 Task Scheduling Stage

The task scheduling stage is the process of assigning the tasks of selected workflow into the fit VMs to be executed. In this stage, the MUSH framework follows the budget and the deadline plan of the user to schedule the tasks. Firstly, the system divides the workflow into levels and sorts the levels into list ascending according to the root direction. Secondly, the system selects each level and schedules its tasks to different free VMs for execution. The algorithm repeats the previous steps until all tasks are scheduled and executed. The system checks the execution status of the workflows every time interval to assure that the execution process moves to right way. If the system explores an error in execution process, it solves the issue and migrates the suspended workflows to new VMs.

4.4 The Outline of the MUSH Framework

The MUSH framework is a new framework, which is used by the provider to manage the rented resources in a perfect way considering the scalability feature. It consists of three stages: Initialization stage, task scheduling stage, and VM allocation stage. The organization, which requests the IaaS model, selects constant IaaS model features according to the provider offers. After that it defines maximum budget plan by selecting a number of homogenous VMs for adding to its IaaS or removing from it during the rent duration. According to the features of the IaaS, the MUSH framework schedules the uploaded workflows adds, and removes the VMs to the organization IaaS. The main aims of our framework are: (i) Achieve the scalability feature in a perfect way. (ii) Achieve the QoS by meeting the budget and the deadline constraints. (iii) Achieve the maximum utilization of the cloud resources by managing the resources under all constraints. (iv) Consume the power. The overall steps of the MUSH framework are shown in Algorithm 3:

Algorithm 3: VM Allocation

```
Begin
Apply the initialization stage
Select a workflow from the workflow list
Schedule tasks for each workflow according to level strategy
Return the final solution and execute the workflow according to this solution
If there are unscheduled workflows
        If there are enough resources
                Select a workflow from the workflow list
```

(Continued)

Algorithm 3: Continued

```
          Apply the task allocation stage
         Return the final solution and execute the workflow according to this solution
         Go to step 6
     Else
         Apply the VM allocation stage
         Go to step 9
     End if
 Else
    Waite the interval time
    Check all running VMs by sending a message for all
    If there is a VM does not response in 10 s
      Check the problem of the VM
      Migrate the task to working free VM
    If there are new submitted workflows
       Go to step 1
    Else
       Free the extra VMs
       If the server utilization < 25%
          Migrate all server VMs to low loaded server
          Shutdown the server
       End if
    End if
   End if
 End
```

5 Results and Discussion

In our experiments, we used a heterogeneous virtual network, where different VMs are used in the network. These VMs are generated randomly with MIPS range ∈ [1000–10000] and number of cores from 2 to 12. The memory has size from 8 giga to 64 giga for each VM. We used laptop core i5, windows 7 64bit and Ram equal 12 Giga bytes to obtain the results. Workflowsim [45] is the toolkit simulator that is used to develop the cloud environment and management system.

5.1 Workflow Application Dataset

To evaluate the performance of the virtual network on cloud computing, various scientific workflow applications are used as data set. Some examples of these scientific workflows are shown in Fig. 5. It shows small Montage workflow, CyberShake workflow, Sipht workflow, Epigenomics workflow and LIGO workflow [46]. All workflow applications represent real-life applications. Montage represents a workflow of tasks that is used for processing sky images. CyberShake is a data-intensive workflow application with extremely large memory and high computing capacity used in seismology to classify earthquake hazards in an area. It is used by the Southern California Earthquake Center while Sipht is used in NCBI database for analyzing RNAs in bacterial replicons.

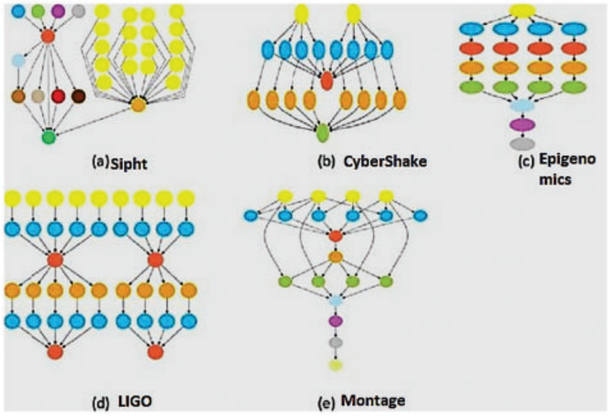


Figure 5: Workflow dataset

In gravitational science, the more suitable workflow is LIGO. It is used for the detection of waveforms generated by the various events in the universe. Epigenomics is one of the famous workflows that is used for cancer diagnosis. In addition, we used random workflows also. Each user can submit one or more workflows.

5.2 Performance Metrics

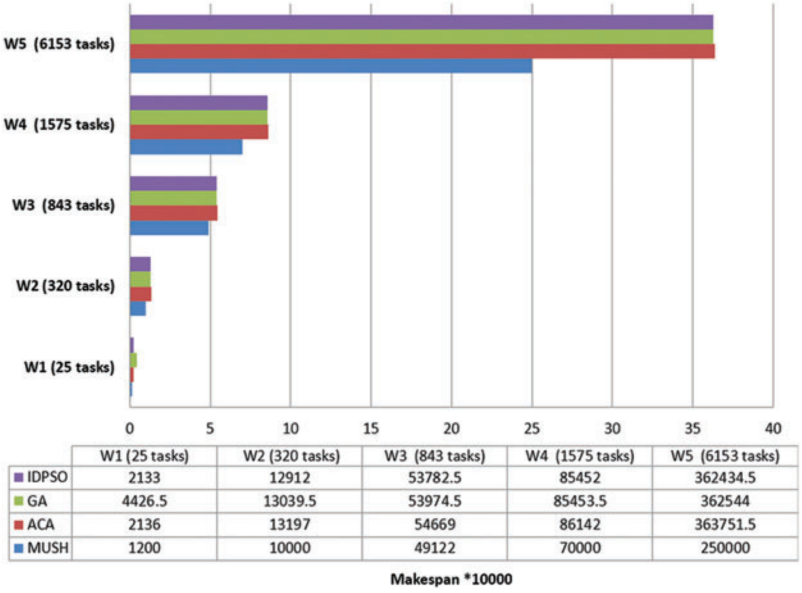
To measure the performance of the MUSH system, we use some metrics such as: makspan, throughput, reliability, and power consumption. Tab. 3 shows the definition and the objective of each metric.

Table 3: The Performance metrics to evaluate the MUSH framework

Metric	Definition	Objective
Makespan	• **Makespan** is the final execution time of the workflow. • It is an important parameter for comparing different algorithms.	• **Makespan** always should be minimized.
Throughput	• **Throughput** represents the number of tasks that are finished in the unit time	• **Measure** the effect of the new system on the response time.
Reliability	• **Reliability** is the quality of performing and working consistently well.	• **Measure** the degree of the consistency of the system
Power consumption	• **Power consumption** is the total power consumed during the execution operation running	• **Measure** the amount of power-saving

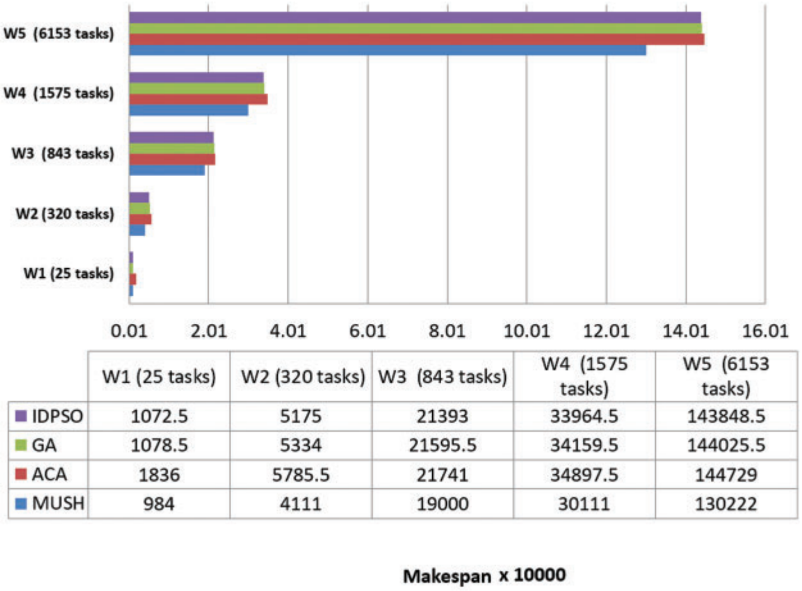
5.2.1 Makespan

It is an important parameter for comparing different algorithms. In Figs. 6–8, and 9 we compare the new system MUSH with GA, ACA, and IDPSO. The symbol W1, W2, W3, W4, and W5 refer to user workflows. From the figures, we note that the MUSH system has a lower makespan than the GA, ACA, and IDPSO algorithms. It can finish 6153 tasks at 10 VMs after four minutes. The new system can manage a various number of workflows with a various number of nodes/tasks. It decreases the makespan for all users and saves time. Because users pay for their usage time, we find that the MUCH system also decreases the cost.



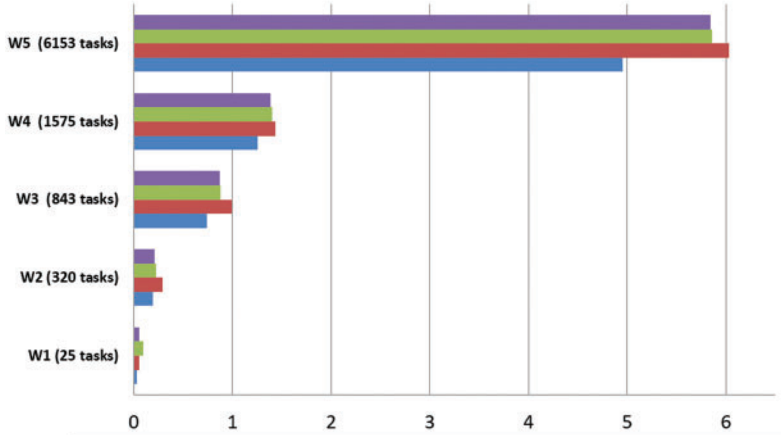
	W1 (25 tasks)	W2 (320 tasks)	W3 (843 tasks)	W4 (1575 tasks)	W5 (6153 tasks)
🟪 IDPSO	2133	12912	53782.5	85452	362434.5
🟩 GA	4426.5	13039.5	53974.5	85453.5	362544
🟥 ACA	2136	13197	54669	86142	363751.5
🟦 MUSH	1200	10000	49122	70000	250000

Figure 6: Makespan of different workflows at virtual network with 50 VMs



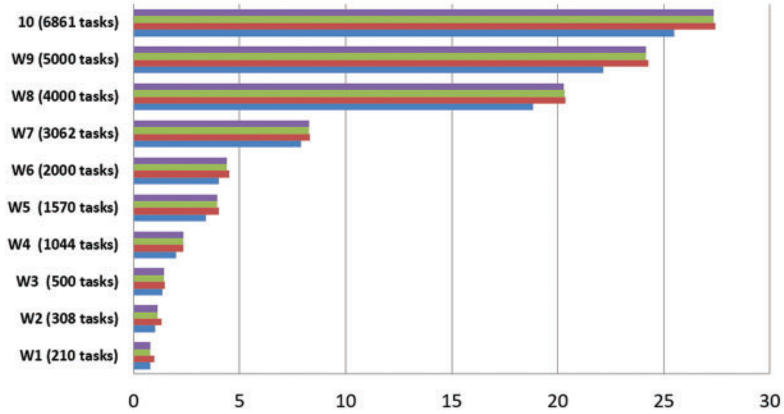
	W1 (25 tasks)	W2 (320 tasks)	W3 (843 tasks)	W4 (1575 tasks)	W5 (6153 tasks)
🟪 IDPSO	1072.5	5175	21393	33964.5	143848.5
🟩 GA	1078.5	5334	21595.5	34159.5	144025.5
🟥 ACA	1836	5785.5	21741	34897.5	144729
🟦 MUSH	984	4111	19000	30111	130222

Figure 7: Makespan of different workflows at virtual network with 100 VMs



	W1 (25 tasks)	W2 (320 tasks)	W3 (843 tasks)	W4 (1575 tasks)	W5 (6153 tasks)
🟪 IDPSO	517.5	2130	8728.5	13852.5	58474.5
🟩 GA	958.5	2290.5	8817	14023.5	58566
🟥 ACA	565.5	2925	9960	14379	60322.5
🟦 MUSH	300	1945	7444	12546	49562

Figure 8: Makespan of different workflows at virtual network with 150 VMs

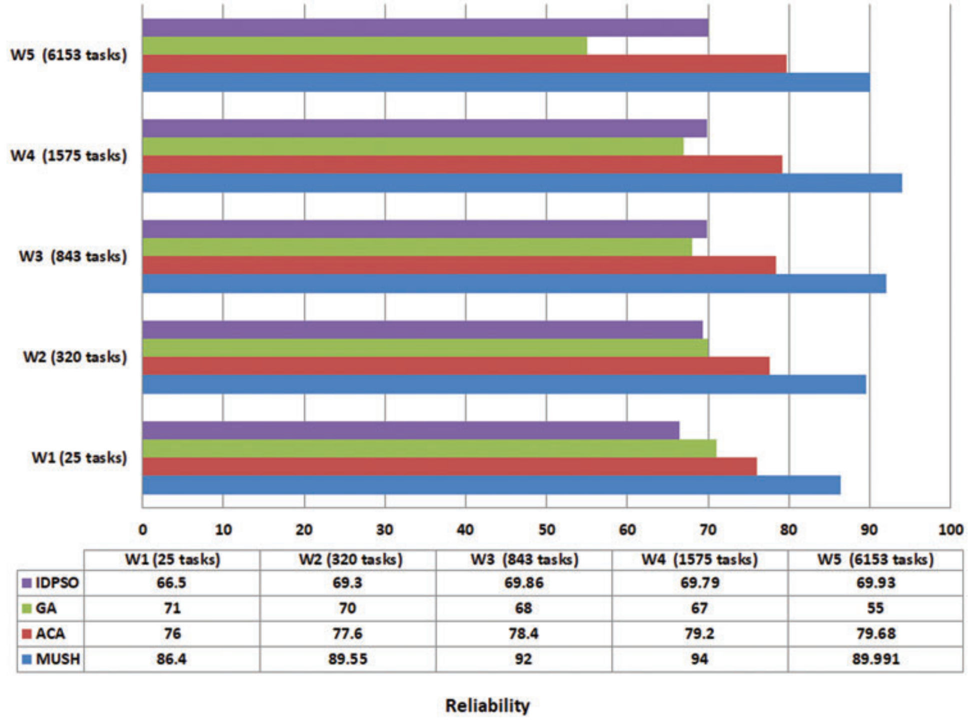


	W1 (210 tasks)	W2 (308 tasks)	W3 (500 tasks)	W4 (1044 tasks)	W5 (1570 tasks)	W6 (2000 tasks)	W7 (3062 tasks)	W8 (4000 tasks)	W9 (5000 tasks)	10 (6861 tasks)
🟪 IDPSO	7919	11348	14435	23339	39362	43975	82677	202979	241623	273285
🟩 GA	7923	11370	14455	23511	39362	44057	82769	203044	241696	273449
🟥 ACA	9943	13312	14852	23449	39989	45025	83234	203398	242501	274186
🟦 MUSH	7789	10000	13455	20000	34165	39987	78894	188452	221455	254788

Figure 9: Makespan of large number of workflows at virtual network with 500 VMs

5.2.2 Reliability

The previous metric is measured user satisfaction. They measure the time and cost according to the time. In this section, we achieve cloud provider satisfaction by measuring system reliability and power consumption in the next section. By measuring system reliability, the cloud provider can know the percentage of consistency of the system. Reliability is calculated as the number of workflows executed divided by the total number of submitted workflows. Fig. 10 shows the average reliability percentage of the new system *vs.* the systems which use GA, ACA, and IDPSO algorithms. From the figure, we find that the degree of new system consistency is higher than the other algorithms. It can work more time without problems. By using a checking point to check out all running VMs with continuous form, the system can solve running problems and the execution operation continuous without errors. From the figure, we find that the MUSH works with reliability of more than 85% for all cases.



	W1 (25 tasks)	W2 (320 tasks)	W3 (843 tasks)	W4 (1575 tasks)	W5 (6153 tasks)
🟪 IDPSO	66.5	69.3	69.86	69.79	69.93
🟩 GA	71	70	68	67	55
🟥 ACA	76	77.6	78.4	79.2	79.68
🟦 MUSH	86.4	89.55	92	94	89.991

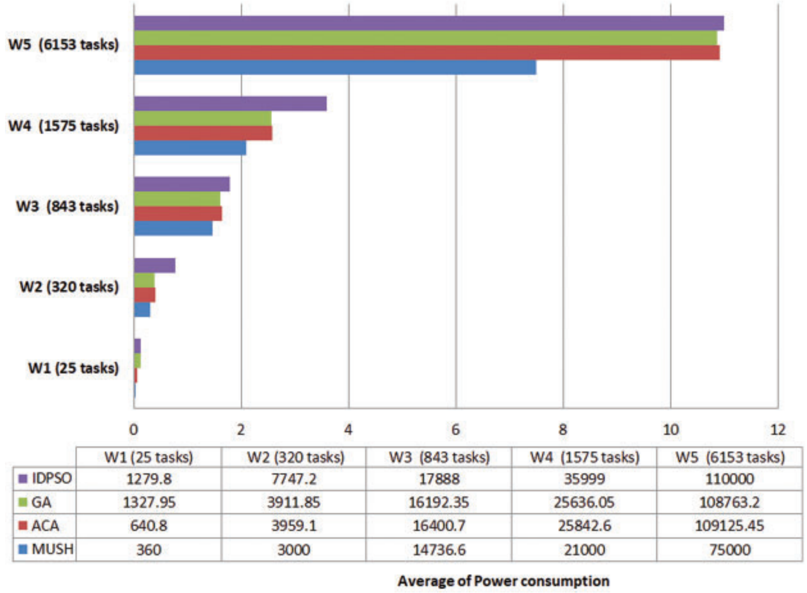
Figure 10: Reliability results

5.2.3 Power Consumption

One of the main issues of cloud datacenters running is a power consumption issue. MUSH system uses an algorithm to manage the running of virtual networks. It moves VMs from low-loaded servers to suitable servers to save power. Fig. 11 shows the average of power consumption of 5 workflows. From the figure, we noted the results of the new system are better than the others. MUSH system can save power and manage the execution operation in a good way.

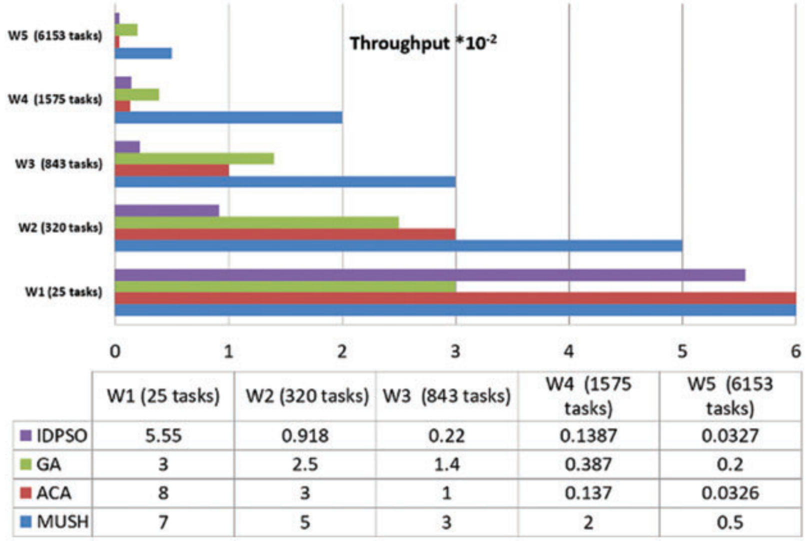
5.2.4 Throughput

User satisfaction is measured by the response time from the system and the number of tasks completed in a short time. Throughput is the metric that is used for measuring user satisfaction. Figs. 12–14, and 15 show the results of the throughput parameters for the MUCH and GA, IDPSO, and ACA algorithms.



	W1 (25 tasks)	W2 (320 tasks)	W3 (843 tasks)	W4 (1575 tasks)	W5 (6153 tasks)
🟪 IDPSO	1279.8	7747.2	17888	35999	110000
🟩 GA	1327.95	3911.85	16192.35	25636.05	108763.2
🟥 ACA	640.8	3959.1	16400.7	25842.6	109125.45
🟦 MUSH	360	3000	14736.6	21000	75000

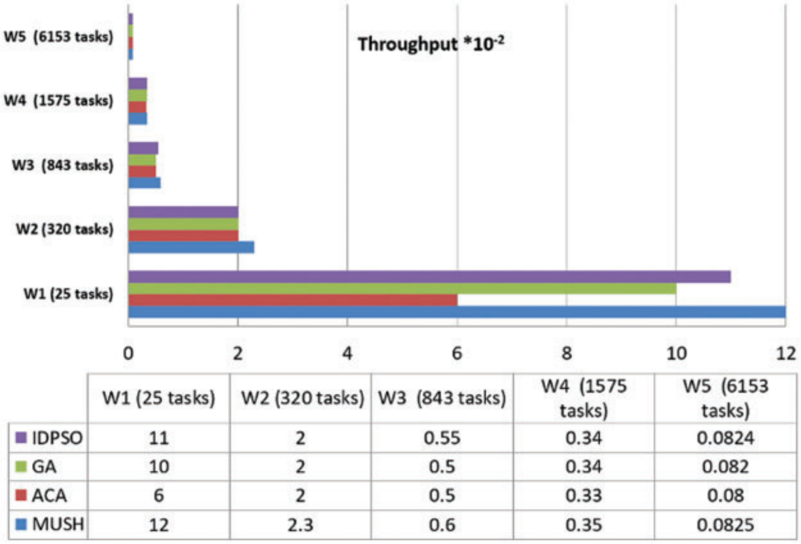
Figure 11: Power consumption results



	W1 (25 tasks)	W2 (320 tasks)	W3 (843 tasks)	W4 (1575 tasks)	W5 (6153 tasks)
🟪 IDPSO	5.55	0.918	0.22	0.1387	0.0327
🟩 GA	3	2.5	1.4	0.387	0.2
🟥 ACA	8	3	1	0.137	0.0326
🟦 MUSH	7	5	3	2	0.5

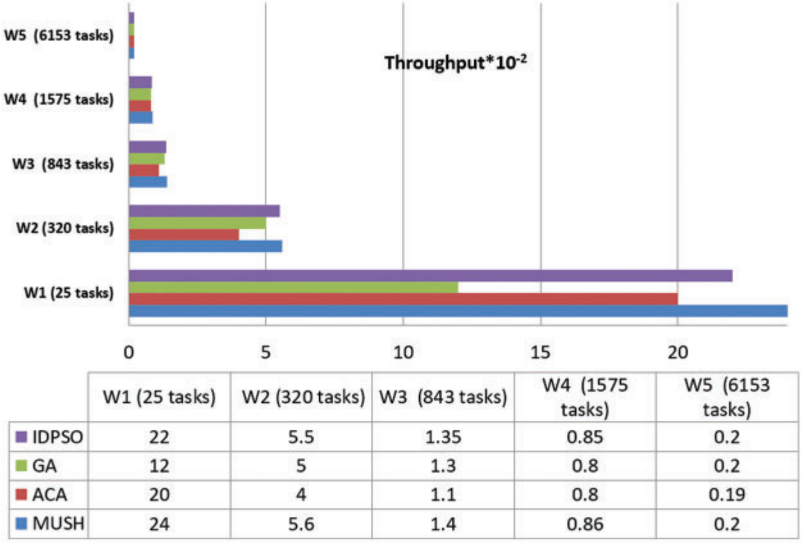
Figure 12: Throughput results at 50 VMs

From the findings, we note that the MUCH algorithm increases the number of finished tasks in the unit time. This means that the response time also decreases. The system is fit for many workflows. It gives a high number of finished tasks in unit time. This appears in Fig. 15, the system finished the execution of 10 tasks for user 1 in one second, and it finished 29 tasks for all users.



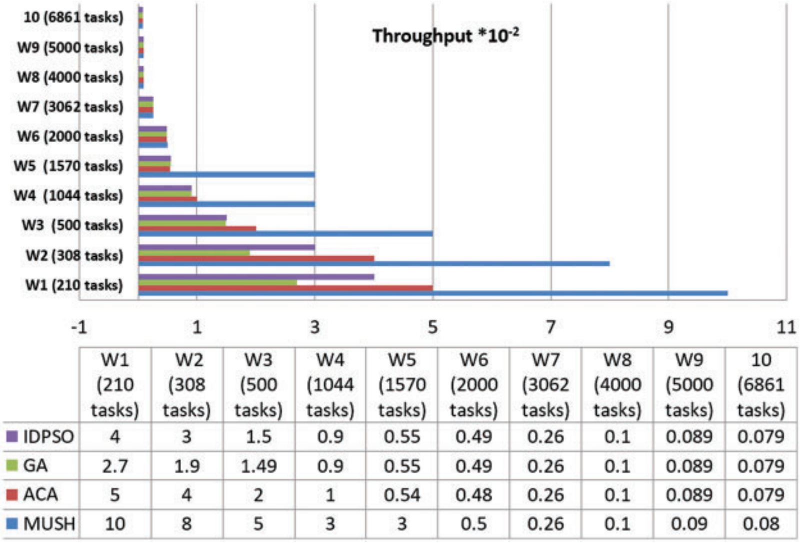
	W1 (25 tasks)	W2 (320 tasks)	W3 (843 tasks)	W4 (1575 tasks)	W5 (6153 tasks)
🟪 IDPSO	11	2	0.55	0.34	0.0824
🟩 GA	10	2	0.5	0.34	0.082
🟥 ACA	6	2	0.5	0.33	0.08
🟦 MUSH	12	2.3	0.6	0.35	0.0825

Figure 13: Throughput results at 100 VMs



	W1 (25 tasks)	W2 (320 tasks)	W3 (843 tasks)	W4 (1575 tasks)	W5 (6153 tasks)
🟪 IDPSO	22	5.5	1.35	0.85	0.2
🟩 GA	12	5	1.3	0.8	0.2
🟥 ACA	20	4	1.1	0.8	0.19
🟦 MUSH	24	5.6	1.4	0.86	0.2

Figure 14: Throughput results at 150 VMs



	W1 (210 tasks)	W2 (308 tasks)	W3 (500 tasks)	W4 (1044 tasks)	W5 (1570 tasks)	W6 (2000 tasks)	W7 (3062 tasks)	W8 (4000 tasks)	W9 (5000 tasks)	10 (6861 tasks)
🟪 IDPSO	4	3	1.5	0.9	0.55	0.49	0.26	0.1	0.089	0.079
🟩 GA	2.7	1.9	1.49	0.9	0.55	0.49	0.26	0.1	0.089	0.079
🟥 ACA	5	4	2	1	0.54	0.48	0.26	0.1	0.089	0.079
🟦 MUSH	10	8	5	3	3	0.5	0.26	0.1	0.09	0.08

Figure 15: Throughput results at 500 VMs

6 Conclusions and Future Work

In this paper, we proposed a new system called MUSH to manage a heterogeneous virtual networks-based cloud computing model. MUSH system can schedule tasks from different workflows in VNC for being executed in low time. In addition, the new system can save power by applying a new VM migration algorithm with chick point to treat VMs failure. MUSH system is compared with the system that uses GA, ACA, and IDPSO algorithms to evaluate performance. We measure makespan, throughput, reliability, and power consumption for comparison. From the results, we find that the new system can manage virtual networks in a better way than GA, ACA, IDPSO algorithms. It gives scheduling solutions with low makespan and high throughput values. Moreover, it uses new algorithms for power saving with VMs migration idea and the reliable algorithm built on task migration. The new system has reliable percentages of more than 85% in different cases. It is planned to discuss and reduce the time complexity in the future work.

Acknowledgement: Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2022R197), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Funding Statement: Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2022R197), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] S. A. Bello, L. O. Oyedele, O. O. Akinade, M. Bilal, J. M. Delgado *et al.*, "Cloud computing in construction industry: Use cases, benefits and challenges," *Automation in Construction*, vol. 122, pp. 1–18, Feb 2021.

[2] A. Sunyaev, "Cloud computing," in *Internet Computing*, Springer, Cham, 2020. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-34957-8_7.

[3] J. Desjardins, "How much data is generated each day?," Retrieved from https://www.weforum.org/agenda/2019/04/how-much-data-is-generated-each-day-cf4bddf29f/. 2022.

[4] K. Dubey, M. Y. Shams, S. C. Sharma, A. Alarifi, M. Amoon *et al.,* "A management system for servicing multi-organizations on community cloud model in secure cloud environment," *IEEE Access*, vol. 7, pp. 159535–159546, 2019.

[5] M. M. Sadeeq, N. M. Abdulkareem, S. R. Zeebaree, D. M. Ahmed, A. S. Sami *et al.,* "IoT and cloud computing issues, challenges and opportunities: A review," *Qubahan Academic Journal*, vol. 1, no. 2, pp. 1–7, 2021.

[6] A. Nasr, D. Kalka, N. A. El-Bahnasawy, S. C. Sharma, G. Attiya *et al.,* "HPFE: A new secure framework for serving multi-users with multi-tasks in public cloud without violating SLA," *Neural Computing & Applications*, vol. 32, no. 11, pp. 6821–6841, 2020.

[7] R. Kollolu, "Infrastructural constraints of cloud computing," *International Journal of Management*," *Technology and Engineering*, vol. 10, pp. 255–260, 2020.

[8] A. Alarifi, K. Dubey, M. Amoon, T. Altameem, F. E. Abd El-Samie *et al.,* "Energy-efficient hybrid framework for green cloud computing," *IEEE Access*, vol. 8, pp. 115356–115369, 2020.

[9] D. A. Amer, G. Attiya, I. Zeidan and A. A. Nasr, "Elite learning harris hawks optimizer for multi-objective task scheduling in cloud computing," *The Journal of Supercomputing*, vol. 78, no. 2, pp. 1–26, 2021.

[10] M. Alam and A. K. Varshney, "A new approach of dynamic load balancing scheduling algorithm for homogeneous multiprocessor system," *International Journal of Applied Evolutionary Computation (IJAEC)*, vol. 7, no. 2, pp. 61–75, 2016.

[11] M. U. Bokhari, M. Alam and F. Hasan, "Performance analysis of dynamic load balancing algorithm for multiprocessor interconnection network," *Perspectives in Science*, vol. 8, pp. 564–566, 2016.

[12] L. M. Haji, S. Zeebaree, O. M. Ahmed, A. B. Sallow, K. Jacksi *et al.,* "Dynamic resource allocation for distributed systems and cloud computing," *TEST Engineering & Management*, vol. 83, pp. 22417–22426, 2020.

[13] D. A. Hasan, B. K. Hussan, S. R. Zeebaree, D. M. Ahmed, O. S. Kareem *et al.,* "The impact of test case generation methods on the software performance: A review," *International Journal of Science and Business*, vol. 5, no. 6, pp. 33–44, 2021.

[14] K. H. Sharif, S. R. Zeebaree, L. M. Haji and R. R. Zebari, "Performance measurement of processes and threads controlling, tracking and monitoring based on shared-memory parallel processing approach," in *The 3rd Int. Conf. on Engineering Technology and its Applications (IICETA)*, Najaf, Iraq, pp. 62–67, 2020.

[15] Y. S. Jghef and S. Zeebaree, "State of art survey for significant relations between cloud computing and distributed computing," *International Journal of Science and Business*, vol. 4, no. 12, pp. 53–61, 2020.

[16] C. M. Mohammed and S. R. Zebaree, "Sufficient comparison among cloud computing services: Iaas, paas, and saas: A review," *International Journal of Science and Business*, vol. 5, no. 2, pp. 17–30, 2021.

[17] Y. Yu and S. Yu, "Cloud task scheduling algorithm based on three queues and dynamic priority," in *Int. Conf. on Power, Intelligent Computing and Systems (ICPICS)*, IEEE, Shenyang, China, pp. 278–282, 2019.

[18] A. Yazdeen, S. R. Zeebaree, M. M. Sadeeq, S. F. Kak, O. M. Ahmed *et al.,* "FPGA implementations for data encryption and decryption via concurrent and parallel computation: A review," *Qubahan Academic Journal*, vol. 1, no. 2, pp. 8–16, 2021.

[19] T. Wang, H. Luo, W. Jia, A. Liu and M. Xie, "MTES: An intelligent trust evaluation scheme in sensor-cloud-enabled industrial internet of things," *IEEE Trans Indust Inform*, vol. 16, pp. 2054–62, 2019.

[20] A. Alzaqebah, R. Al-Sayyed and R. Masadeh, "Task scheduling based on modified grey wolf optimizer in cloud computing environment," in *2nd Int. Conf. on new Trends in Computing Sciences (ICTCS)*, Amman, Jordan, pp. 1–6, 2019.

[21] S. Pang, W. Li, H. He, Z. Shan and X. Wang, "An EDA-GA hybrid algorithm for multi-objective task scheduling in cloud computing," *IEEE Access*, vol. 7, pp. 146379–146389, 2019.

[22] S. Liu and Y. Yin, "Task scheduling in cloud computing based on improved discrete particle swarm optimization," in *2nd Int. Conf. on Information Systems and Computer Aided Education (ICISCAE)*, IEEE, Dalian, China, pp. 594–597, 2019.

[23] Z. Zong, "An improvement of task scheduling algorithms for green cloud computing," in *15th Int. Conf. on Computer Science & Education (ICCSE)*, IEEE, Delft, Netherlands, pp. 654–657, 2020.

[24] V. A. Lepakshi and C. S. R. Prashanth, "Efficient resource allocation with score for reliable task scheduling in cloud computing systems," in *2nd Int. Conf. on Innovative Mechanisms for Industry Applications (ICIMIA)*, IEEE, Bangalore, India, pp. 6–12, 2020.

[25] M. S. Sanaj and P. J. Prathap, "An enhanced round robin (ERR) algorithm for effective and efficient task scheduling in cloud environment," in *Advanced Computing and Communication Technologies for High Performance Applications (ACCTHPA)*, IEEE, Cochin, India, pp. 107–110, 2020.

[26] Y. Shi, K. Suo, S. Kemp and J. Hodge, "A task scheduling approach for cloud resource management," in *Fourth World Conf. on Smart Trends in Systems, Security and Sustainability (WorldS4)*, IEEE, London, UK, pp. 131–136, 2020.

[27] Z. T. Al-Azez, A. Q. Lawey, T. E. El-Gorashi and J. M. Elmirghani, "Virtualization framework for energy efficient IoT networks," in *4th Int. Conf. on Cloud Networking (CloudNet)*, IEEE, Niagara Falls, ON, Canada, pp. 74–77, 2015.

[28] Z. T. Al-Azez, A. Q. Lawey, T. E. H. El-Gorashi and J. M. H. Elmirghani, "Energy efficient IoT virtualization framework with passive optical access networks," in *18th Int. Conf. on Transparent Optical Networks (ICTON)*, IEEE, Trento, Italy, pp. 1–4, 2016.

[29] M. Barcelo, A. Correa, J. Llorca, A. M. Tulino, J. L. Vicario *et al.,* "IoT-Cloud service optimization in next generation smart environments," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 4077–4090, 2016.

[30] K. Wang, Y. Wang, Y. Sun, S. Guo and J. Wu, "Green industrial internet of things architecture: An energy-efficient perspective," *IEEE Communications Magazine*, vol. 54, no. 12, pp. 48–54, 2016.

[31] M. Bahrami, A. Khan and M. Singhal, "An energy efficient data privacy scheme for IoT devices in mobile cloud computing," in *Int. Conf. on Mobile Services (MS)*, IEEE, San Francisco, CA, USA, pp. 190–195, 2016.

[32] G. Xing, X. Xu, H. Xiang, S. Xue, S. Ji *et al.,* "Fair energy-efficient virtual machine scheduling for internet of things applications in cloud environment," *International Journal of Distributed Sensor Networks*, vol. 13, no. 2, pp. 1–11, 2017.

[33] T. Baker, M. Asim, H. Tawfik, B. Aldawsari and R. Buyya, "An energy-aware service composition algorithm for multiple cloud-based IoT applications," *Journal of Network and Computer Applications*, vol. 89, pp. 96–108, 2017.

[34] N. Hamdi and W. Chainbi, "A survey on energy aware VM consolidation strategies," *Sustainable Computing Informatics and Systems*, vol. 23, pp. 80–87, 2019.

[35] X. Wu, Y. Zeng and G. Lin, "An energy efficient VM migration algorithm in data centers," in *16th Int. Symp. on Distributed Computing and Applications to Business, Engineering and Science*, IEEE, AnYang, China, pp. 27–30, 2017.

[36] S. Y. Hsieh, C. S. Liu, R. Buyya and A. Y. Zomaya, "Utilization-prediction-aware virtual machine consolidation approach for energy-efficient cloud data centers," *Journal of Parallel and Distributed Computing*, vol. 139, pp. 99–109, 2020.

[37] M. Kumar, A. Kishor, J. Abawajy, P. Agarwal, A. Singh *et al.,* "ARPS: An autonomic resource provisioning and scheduling framework for cloud platforms," *IEEE Transactions on Sustainable Computing*, https://doi.org/10.1109/TSUSC.2021.3110245, 2021.

[38] M. Kumar and S. C. Sharma, "PSO-COGENT: Cost and energy efficient scheduling in cloud environment with deadline constraint," *Sustainable Computing: Informatics and Systems*, vol. 19, pp. 147–164, 2018.

[39] M. Kumar, S. C. Sharma, A. Goel and S. P. Singh, "A comprehensive survey for scheduling techniques in cloud computing," *Journal of Network and Computer Applications*, vol. 143, pp. 1–33, 2019.

[40] H. Siar and M. Izadi, "Offloading coalition formation for scheduling scientific workflow ensembles in fog environments," *Journal of Grid Computing*, vol. 19, no. 3, pp. 1–20, 2021.

[41] S. Khan, B. Nazir, I. A. Khan, S. Shamshirband and A. T. Chronopoulos, "Load balancing in grid computing: Taxonomy, trends and opportunities," *Journal of Network and Computer Applications*, vol. 88, pp. 99–111, 2017.

[42] M. Mesbahi, A. M. Rahmani and A. T. Chronopoulos, "Cloud light weight: A new solution for load balancing in cloud computing," in *Int. Conf. on Data Science & Engineering (ICDSE)*, IEEE, Kochi, India, pp. 44–50, 2014.

[43] A. Nasr, N. A. El-Bahnasawy, G. Attiya and A. El-Sayed, "Cost-effective algorithm for workflow scheduling in cloud computing under deadline constraint," *Arabian Journal for Science and Engineering*, vol. 44, no. 4, pp. 3765–3780, 2019.

[44] F. M. Talaat, M. S. Saraya, A. I. Saleh, H. A. Ali and S. H. Ali, "A load balancing and optimization strategy (LBOS) using reinforcement learning in fog computing environment," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 11, pp. 1–16, 2020.

[45] W. Chen and E. Deelman, "Workflowsim: A toolkit for simulating scientific workflows in distributed environments," in *8th Int. Conf. on E-Science*, IEEE, Chicago, IL, USA, pp. 1–8, 2012.

[46] T. Ghafarian, B. Javadi and R. Buyya, "Decentralised workflow scheduling in volunteer computing systems," *International Journal of Parallel Emergent and Distributed Systems*, vol. 30, no. 5, pp. 343–365, 2015.