Hindawi Publishing Corporation
Scientific Programming
Volume 2016, Article ID 8239239, 9 pages
http://dx.doi.org/10.1155/2016/8239239

Research Article

Adaptive Cost-Based Task Scheduling in Cloud Environment

Mohammed A. S. Mosleh,[1] G. Radhamani,[1] Mohamed A. G. Hazber,[2] and Syed Hamid Hasan[3]

[1]*School of IT & Science, Dr. GR Damodaran College of Science, Coimbatore, India*
[2]*International School of Software Engineering, Wuhan University, Wuhan, China*
[3]*Information Systems Department, King Abdulaziz University, Jeddah, Saudi Arabia*

Correspondence should be addressed to Mohammed A. S. Mosleh; mohammed.mosleh@grd.edu.in

Received 22 June 2016; Revised 19 September 2016; Accepted 20 October 2016

Academic Editor: Frank De Boer

Task execution in cloud computing requires obtaining stored data from remote data centers. Though this storage process reduces the memory constraints of the user's computer, the time deadline is a serious concern. In this paper, Adaptive Cost-based Task Scheduling (ACTS) is proposed to provide data access to the virtual machines (VMs) within the deadline without increasing the cost. ACTS considers the data access completion time for selecting the cost effective path to access the data. To allocate data access paths, the data access completion time is computed by considering the mean and variance of the network service time and the arrival rate of network input/output requests. Then the task priority is assigned to the removed tasks based data access time. Finally, the cost of data paths are analyzed and allocated based on the task priority. Minimum cost path is allocated to the low priority tasks and fast access path are allocated to high priority tasks as to meet the time deadline. Thus efficient task scheduling can be achieved by using ACTS. The experimental results conducted in terms of execution time, computation cost, communication cost, bandwidth, and CPU utilization prove that the proposed algorithm provides better performance than the state-of-the-art methods.

1. Introduction

Cloud computing is a promising technology that provides efficient services to the customers in a distant virtual platform on a pay-per-use model. The definition for cloud computing given by NIST [1] is as follows: cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to shared computing resources which can be provisioned and provided with minimal interaction. Cloud computing provides different types of services such as infrastructure, software, and platform to the requested users with a specific price for the services. Cloud services use the internet and the central remote servers to maintain the data and applications. Cloud computing allows consumers and businesses to use applications without installation and access their personal files at any computer with internet access. This approach improves the computing processes such as data storage and processing. Cloud is deployed in different models: public cloud, private clouds, hybrid cloud, community cloud, and distributed cloud are some examples.

Service oriented architecture is the basic principle of the cloud computing which considers everything on the cloud as a service [2]. Infrastructure-as-a-service (IaaS) is the service of providing the physical machines (PM) or virtual machines (VM) to the user for processing resources, data partitioning, scaling, security, and backup processes. Platform-as-a-service (PaaS) provides the vendors with the platforms for development of applications including databases, web servers, and developmental tools. Software-as-a-service (SaaS) provides services for the e-mails, virtual desktops, communication processes, and gaming applications. The services are normally paid services whose price is fixed by the service providers based on the usage level of the customers. The price of the cloud services is very less compared to the other installed services.

In cloud computing, the tasks are performed in the physical machines (PMs) or the VMs as per the task requirements. The data required for the execution of the tasks and services are stored at multiple distant storage locations called the data centers which are also used with specific cost [3]. When the

tasks are performed in the processing machines, the required data are requested and obtained from the data centers. The data from the data centers has to reach the VM within the particular time which is always the access completion time. The problem with this process is that the data is accessed through certain paths which are bound by the computation and storage costs. So it is possible that either one of the two situations arises: in order to obtain the data in time, the cost has to be sacrificed or, in order to reduce the cost, the delay in data access has to be accepted. This problem reduces the overall scheduling performance.

In order to overcome the data access problem, an adaptive cost-based task scheduling (ACTS) is proposed in this paper so that the data is obtained at the required time without delay and through affordable cost paths. The proposed approach estimates the completion time for accessing the data [13] that are required by the VM machines during the particular task executions. Then the cost of each possible path is estimated by the sum of computation, communication, and storage costs [14] of the path. Using the completion time for data access the priority of the tasks is assigned. The paths with high cost but with quick data access are assigned to tasks with high priority and the paths with low cost are assigned to the low priority tasks. Thus the data paths can be adaptively selected to reduce the overall cost and effectively deliver the data at the required time.

The remainder of the paper is summarized as follows: Section 2 explains the related researches briefly and presents the analysis of scheduling schemes. Section 3 presents the methodologies utilized in the paper. Section 4 provides the experimental results and their discussions. Section 5 concludes the research.

2. Related Works

A cloud scheduler is a cloud-enable distributed resource manager. It manages virtual machines on clouds to create an environment for job execution. The FIFO scheduler in Hadoop MapReduce, fair scheduler in Facebook, and capacity scheduler in Yahoo are typical examples that serve the cloud systems with efficient and equitable resource management, but none of these schedulers satisfies QoS (quality of service) constraints. Therefore, they are not applicable to soft real-time needed applications and services, which are becoming more and more important and necessary in the hybrid cloud environment. The main objective of this section is not to propose methodologies to overcome all of the current issues in cloud task scheduling but to study and analyze some of the current methodologies and focus on finding their drawbacks.

Sahni and Vidyarthi [4] presented a cost-effective deadline constraint dynamic scheduling algorithm for the scientific workflows. The workflow scheduling algorithms in the grid and clusters are efficient but could not be utilized effectively in the cloud environment because of the on-demand resource provisioning and pay-as-you-go pricing model. Hence the scheduling using a dynamic cost-effective deadline-constrained heuristic algorithm has been utilized to exploit the features of cloud by considering the virtual machine performance variability and instance acquisition delay to determine the time scheduling. The problem with the approach is that VM failures may adversely affect the overall workflow execution time.

Tsai et al. [5] proposed hyper-heuristic scheduling algorithm (HHSA) for providing effective cloud scheduling solutions. The diversity detection and improvement detection operators are utilized in this approach to dynamically determine the better low-level heuristic for the effective scheduling. HHSA can reduce the makespan of task scheduling and improves the overall scheduling performance. The drawback is that the approach has high overhead of connection which reduces the importance of scheduling and thus reduces the overall performance.

Zhu et al. [6] proposed an agent-based dynamic scheduling algorithm named ANGEL for effective scheduling of tasks in the virtualized clouds. In this approach, a bidirectional announcement-bidding mechanism and the collaborative process are performed to improve the scheduling performance. To further improve the scheduling, elasticity is considered to dynamically add VMs. The calculation rules are generated to improve the bidding process that in turn reduces the delay. The problem with this approach is that it reduces the performance as it does not consider the communication and dispatching times.

Zhu et al. [7] presented an evolutionary multiobjective (EMO) workflow scheduling approach to reduce the workflow scheduling problem such as cost and makespan. Due to the specific properties of the workflow scheduling problem, the existing genetic operations, such as binary encoding, real-valued encoding, and the corresponding variation operators are based on them in the EMO. The problem is that the approach does not consider monetary costs and time overheads of both communication and storage.

Zhang et al. [8] proposed a fine-grained scheduling approach called phase and resource information-aware scheduler for MapReduce (PRISM) for scheduling in the MapReduce model. MapReduce has been utilized for its efficiency in reducing the running time of the data-intensive jobs but most of the MapReduce schedulers are designed on the basis of task-level solutions that provide suboptimal job performance. Moreover, the task-level schedulers face difficulties in reducing the job execution time. Hence the PRISM was developed which divides tasks into phases. Each phase with a constant resource usage profile performs scheduling at the phase level. Thus the overall job execution time can be reduced significantly but the problem of meeting job deadlines in the phase level scheduling is a serious concern that requires specified attention.

Zhu et al. [9] presented real-time task oriented energy aware (EA) scheduling called EARH for the virtualized clouds. The proposed approach is based on rolling-horizon (RH) optimization and the procedures are developed for creation, migration, and cancellation of VMs to dynamically adjust the scale of cloud to achieve real time deadlines and reduce energy. The EARH approach has the drawback of the number of cycles assigned to the VMs that cannot be updated dynamically.

Table 1: Drawbacks of scheduling schemes in literature.

Author	Scheduling scheme	Drawbacks
Sahni and Vidyarthi [4]	Cost-effective deadline constraint dynamic scheduling algorithm	VM failures increase the workload of other VMs and affect the execution time
Tsai et al. [5]	Hyper-heuristic scheduling algorithm	High overhead of connection
Zhu et al. [6]	Agent-based scheduling algorithm in virtualized clouds (ANGEL)	Nonconsideration of communication and dispatching time reducing performance
Zhu et al. [7]	Evolutionary multiobjective (EMO) workflow scheduling	Nonconsideration of monetary costs and time overhead does not improve performance
Zhang et al. [8]	Phase and resource information-aware scheduler for MapReduce (PRISM)	Deadlines are not specified
Zhu et al. [9]	Energy aware rolling-horizon (EARH) optimization based scheduling	Lack of updation in number of VM cycles
Maguluri and Srikant [10]	Throughput-optimal scheduling & load-balancing algorithm	Utilizing queue lengths in weights is based on assumption
Zuo et al. [11]	Self-adaptive learning particle swarm optimization- (SLPSO-) based scheduling	Lack of priority to deadline constraint tasks results in task failures
Su et al. [12]	Cost efficient task scheduling	Does not consider the completion time and cost (computation cost and communication cost)

Maguluri and Srikant [10] suggested a scheduling method for job scheduling with unknown duration in the cloud environment. The job sizes are assumed to be unknown not only at arrival, but also at the beginning of service. Hence the throughput-optimal scheduling and load-balancing algorithm for a cloud data center is introduced, when the job sizes are unknown. This algorithm is based on using queue lengths for weights in max-weight schedule instead of the workload.

Zuo et al. [11] presented self-adaptive learning particle swarm optimization- (SLPSO-) based scheduling approach for deadline constraint task scheduling in hybrid IaaS clouds. The approach solves the problem of meeting the peak demand for preserving the quality-of-service constraints by using the PSO optimization technique. The approach provides better scheduling of the tasks with maximizing the profit of IaaS provider while guaranteeing QoS. The problem with this approach is the lack of priority determination which results in failure of deadline tasks.

Scheduling tasks in a cloud computing environment is a challenging process. In [12] Su et al. presented a cost efficient task scheduling method that can be utilized for processing large size programs. But the performance of the approach is not sufficient as it did not consider the completion time and cost for scheduling.

From the literature it is found that the major issues in the above described methods are high cost consumption especially for communication and computation of data from cloud data centers. The inability to meet up the deadlines, due to the inappropriate data path allocation while task scheduling, is another area of concern. The analysis of various scheduling schemes is listed as below.

2.1. Analysis of Scheduling Schemes. Generally, the efficient task scheduling concepts of the clusters and the grid are not effective in the cloud environment. The main reason is that in cloud computing the resource provision is on-demand and the resources are provided on the basis of pay-per-use. Hence the scheduling approach has to make use of the features of the cloud in order to efficiently schedule the tasks without time delay. While processing a task in a VM, the data are needed to be obtained from the distant data centers located at multiple locations. As the tasks are deadline constraint, the data are needed to be obtained within the particular time using effective scheduling approaches. However, the solution for scheduling deadline constraint tasks in the cloud leads to a new problem in the form of cost. The computation and the storage resources are the basic resources in the cloud environment that forms the cost models.

Table 1 shows the various scheduling schemes described in literature and their drawbacks.

The high cost problems can be reduced by effectively selecting the minimum cost paths based on availability of the data paths. The problem is that not all the tasks take the same execution time which means some tasks require data quicker than the other tasks. But when using only the minimum cost path, the data would have to wait in queue or might be lost due to queue overflow. So the cost paths are needed to be selected adaptively for deadline constraint tasks. These two problems are the major focus of this research.

3. Adaptive Cost-Based Task Scheduling

The proposed adaptive cost-based task scheduling (ACTS) is discussed in this section. The scheduling of the tasks to the VMs can be performed effectively using the proposed scheduling method. This work takes inspiration from the work of Su et al. In their work, cost efficient task scheduling is used which considers the overall execution time and total monetary costs for scheduling. Though the execution time and monetary cost are considered this scheme cannot be considered as efficient due to the reason that these two factors are collaborative factors. The execution time is the

time for task completion. This means the execution time includes the time from which the tasks are assigned to a VM until the output of the tasks is obtained. However, the time consumed for each process in task execution varies and not all of them can be minimized. In this sense, the time taken for obtaining the data from the data centers for task execution is considerably higher than all other process in task execution. Similarly, the monetary cost is the combined cost of resources for computation, communication, storage, data transfer, and so forth; in these processes, the costs for computation and communication are normally higher than other costs. But Su et al. considered only the combined factors for scheduling. Hence in the proposed ACTS we focused on specifically considering the individual processes as factors for scheduling. The major factors are *data access completion time*, *computation cost,* and *communication cost*.

The data that are required to be processed in the VM or the PM are stored in the distant data centers. These data are needed to be fetched to the processing VMs from the data centers through the cost-effective paths. The data access of each VM follows an independent Poisson distribution associated with the average rate of the arrival rate of the network I/O requests. The data access to the driver domain (PM) is processed on the basis of providing the access to the first come users while the other users wait in the queue. The service time of a data access in the driver domain is represented in an arbitrary distribution.

The data access completion time is considered to be the determination point in the selection of the data paths. The completion time for the data access is calculated by utilizing the parameters of the network input/output requests in the physical machines. The mean of the service time network I/O requests in the PMs is given by μ and the variance of the service time network I/O requests in the PMs is given by σ. The arrival rate of the network I/O requests to the PMs is given by λ. Then the completion time t of a data access can be estimated using the formula

$$t = \frac{2\mu - \lambda + \lambda\mu^2\sigma^2}{2\mu^2 - 2\lambda\mu}. \quad (1)$$

The arrival rate of the network I/O requests to the PMs can be calculated by

$$\lambda = \sum \lambda^{(e)} \cdot r^{(e)} + \sum \lambda^{(n)} \cdot r^{(n)}, \quad (2)$$

where $r^{(e)}$ and $r^{(n)}$ are the ratio of the CPU time allocated to the existing and new VMs. $\lambda^{(e)}$ and $\lambda^{(n)}$ are the arrival rate of the network I/O requests of the existing and new VMs to the PMs.

The tasks are performed in the virtual machine (VM) which obtains the data from cloud centers through the data access paths. Each data access path contains resources for processing the requests and accessing the data and also requires storage resources for storing the accessed data. Each of the resources carries certain costs for utilizing the resources. The computation cost includes the cost of resources for execution of the I/O requests for the data access and the cost for reaccessing the same data again. It also includes the cost for regenerating the datasets. The communication cost is the total cost of the resources utilized for the processing of the I/O requests. It can be expressed as the product of the data set size and the network traffic price.

The cost of the possible data access paths is analyzed in order to determine the minimum cost path. The cost of each path can be estimated by

$$\text{Cost} = \text{Computation cost} + \text{Communication cost}. \quad (3)$$

The computation cost and communication cost are vital in the determination of the cost-effective paths as these resources handle the I/O requests of the VMs. When the VM executes a task, for accessing the data from the data centers, the VM sends request for the access. The data centers receive the I/O requests and then provide access for the data.

The proposed ACTS considers both the cost and the completion time of data access for efficiently scheduling the tasks. ACTS assigns priority to the tasks based on the completion time. Time T is chosen as a fixed time and the completion time is compared with T to determine the priority. The low priority tasks are those that have more completion time and hence the path is selected as minimum cost path to reduce the overall cost. The reason for this approach to low priority tasks is because these tasks can be executed in a normal time without much urgency. Similarly, the high priority tasks are those that require data within the less completion time and hence the paths that provide quicker data access are selected without waiting for the minimum cost path. This may increase the cost but the main aim is to obtain the requested data within the time and hence the small variation in the overall cost can be negligible. After the execution of the tasks, the CPU utilization and the bandwidth utilization are estimated.

Figure 1 shows the proposed ACTS procedure. This work focuses on scheduling the tasks to the VMs with minimum cost paths to reduce the complexity in the data accessing from the cloud data center. The tasks are allocated to the underloaded VMs based on the normal load conditions. The tasks allocated to VMs access the data from the distant cloud data centers. The cost that recurred for I/O processing is computed and the completion time for data access is estimated. Then the CPU utilization and bandwidth utilization are calculated and updated for successive task executions.

For example, let us consider V tasks of simple mathematical programs with flexible properties of bandwidth, random access memory (RAM), and million instructions per second (mips). These parameters of the cloud tasks are user defined and can be flexibly chosen. Moreover, the simulations are made in the real-time simulation environment (CloudSim) which provides user friendly behavior. The tasks are nonpreemptive dependent tasks.

The VMs are initiated from the cloud environment with existing VMs denoted as E and the newly initiated VMs are placed under N. This is because when there is large load, the new VMs are introduced. The tasks execute the simple mathematical programs with the length differing based on the initiated codes. The addition program of $(a + b)$ is executed once for a task with 4 bits while it is repeated to achieve the prescribed length in the chosen tasks.

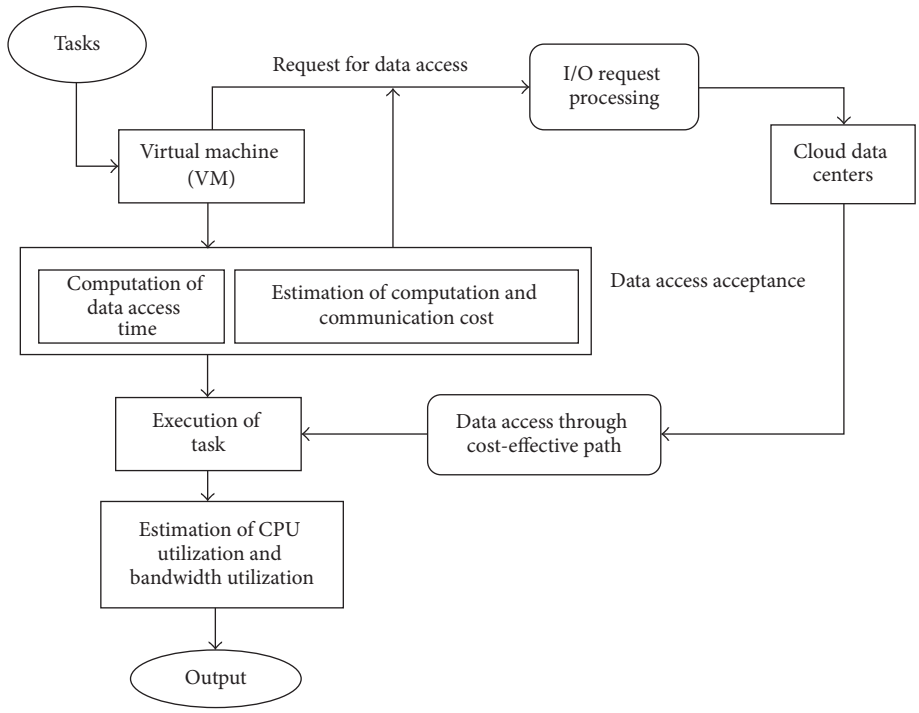

Figure 1: Adaptive cost-based task scheduling.

Now let us take task v with m resources available. Initially the tasks are checked for possible execution. All the VMs are running in parallel and are unrelated and each VM runs on its own resources. There is no sharing of its own resources by other VMs. We schedule nonpreemptive dependent tasks to the VMs. For each task v, the arrival rate λ_j and $T(v_i, m_j)$ are calculated. Then the costs C_{comp} and $C_{\text{communication}}$ are computed for each data path d using (2), (7), and (9). The computation cost in equation (7) is estimated as the sum of all costs incurred for running a task v on a VM m of a provider p (8) while the communication cost (9) is the product of cost for data required and the inbound network traffic prices. Based on the completion time, the tasks priority is assigned. Then based on $T(v_i, m_j)$ and cost, the paths are sorted. Then the paths are allocated to each task and then the underloaded VMs are loaded with the tasks which access the data from the cloud data center at the deadline time. Then the CPU utilization (11) and bandwidth utilization (12) are calculated for determining the efficiency of the system. This scheduling procedure is sorted in the following algorithm.

Algorithm 1 (adaptive cost-based task scheduling).

Input: number of tasks, VMs

Output: task scheduling

Begin

Deploy the set of physical machines.

E = set of existing VMs present in the cloud computing system.

N = set of new VMs to be created.

Set of tasks $V = \{v_1, v_2, \ldots, v_i\}$.

Set of resources $M = \{m_1, m_2, \ldots, m_n\}$.

For each task v_i,

Arrival rate λ_j to PM_j using (1)

$$\lambda_j = \sum_{i \in E} \lambda_i^{(e)} \cdot r_i^{(e)} + \sum_{i \in N} \lambda_i^{(n)} \cdot r_i^{(n)}, \tag{4}$$

//where $r_i^{(e)}$ *and* $r_i^{(n)}$ *are the ratio of the CPU time allocated to the existing and new VMs.* $\lambda_i^{(e)}$ *and* $\lambda_i^{(n)}$ *are the arrival rate of the network I/O requests of the existing and new VMs to the PMs.*

Compute completion time of data access $T(v_i, m_j)$ using (2)

$$T\left(v_i, m_j\right) = \frac{2\mu_j - \lambda_j + \lambda_j \mu_j^2 \sigma_j^2}{2\mu_j^2 - 2\lambda_j \mu_j}, \tag{5}$$

//where μ_j *is the mean service time of network I/O requests in* m_j, σ_j *is the variance of the service time distribution, and* λ_j *is the arrival rate of network I/O requests to* m_j

End for

Compute cost of each possible data path d using (3)

$$\begin{aligned} &\text{Cost} = \text{Computation cost} + \text{Communication cost}, \\ &C_d = C_{\text{comp}} + C_{\text{communication}}. \end{aligned} \tag{6}$$

Computation cost

$$C_{\text{comp}} = \sum_{v_i}^{V} \min_{m_j}^{M} \left(C_{\text{task}} \left(v_i, p, m_j \right) \right), \tag{7}$$

where the cost of running a task v_i on provider p with VM m_j is defined as

$$C_{\text{task}} \left(v_i, p, m_j \right) = \begin{cases} \text{RT}_{v_i}^{m_j,p} \cdot C_{m_j}^{p}, & \text{RT}_{v_i}^{m_j,p} \leq \text{DL}_a \\ \infty, & \text{RT}_{v_i}^{m_j,p} > \text{DL}_a \\ \infty, & m_j \notin M, \end{cases} \tag{8}$$

//where set of tasks is given by V and *p is the service provider.* DL_a *is the time to deadline of* v_i. $RT_{v_i}^{m_j,p}$ *is the runtime of a task* v_i. $C_{m_j}^{p}$ *is the cost of running an VM on p for one time unit.*

Communication cost can be computed as

$$C_{\text{communication}} = D_a \cdot \text{NW}_p^{\text{in}}, \tag{9}$$

//where D_a *is the GB required for task* v_i *and* NW_p^{in} *is the inbound network traffic prices per GB of the provider p.*

Select minimum cost path $C_{d\min}$.

Assign priority to tasks v_i.

If (*Priority of* v_i = *low* && $T(v_i, m_j) \geq T$)

$$\text{Data path} = C_{d\min}. \tag{10}$$

Else if (*Priority of* v_i = *high* && $T(v_i, m_j) < T$)

// T is a fixed time with which the data access completion time of the tasks is compared to determine the priority;

analyze data paths C_d which satisfies the time to deadline DL_a for tasks v_i;

data path = $C_{dt}[C_{dt} \neq C_{d\min}]$;

// path has faster data access to satisfy time to deadline even without minimum cost

End if

Assign tasks to VMs.

Estimation of CPU utilization

$$\overline{\text{CPU}} = \frac{\text{cl}_{\text{MIPS}} \cdot \text{CPU}_{\text{MIPS}}}{1000 \cdot \text{cl}_{\text{ms}}}, \tag{11}$$

// where CPU is the CPU utilization; cl_{MIPS} *is the calculated cloudlet's MIPS length;* CPU_{MIPS} *is the MIPS ration of the CPU;* cl_{ms} *is the cloudlet's duration in milliseconds when executed on a CPU with a MIPS rating of* CPU_{MIPS}

Estimation of bandwidth utilization

$$\text{BW}_u = \frac{\tau_v \times 100}{\text{BW}_v \times \psi_v}, \tag{12}$$

// where BW_u *is the bandwidth utilization;* BW_v *is the allotted bandwidth quota;* τ_v *is the amount of data transferred during the life of VM;* ψ_v *is the duration which is the VM lifetime and it is equal to the VM release time to the VM creation time.*

Update VM characteristics for next iteration.

End.

3.1. Description. The tasks V, the number of VMs, and VM resources m are initialized.. The set of existing VMs E and the set of newly created VMs N are assigned. For each task, the data access completion time is calculated as $T(v_i, m_j)$. Similarly the computation cost and communication cost are also calculated in order to estimate the cost of each data path. Using the completion time and computation cost, and communication cost of each path, the scheduling is performed. The tasks are assigned priorities based on the completion time. The high priority tasks which have less completion time are allocated fast data access paths C_{dt} that satisfy the time to deadline without prioritizing the cost. But for the low priority tasks which have high completion time the minimum cost paths $C_{d\min}$ are allocated. Then the tasks are executed and the utilization of CPU and bandwidth are calculated.

4. Experimental Results

The experiments are conducted to evaluate the performance of the proposed adaptive cost-based task scheduling and the results are tabulated. The cost efficient task scheduling is presented in [12] utilized in this work for performance comparison without considering the cost and completion time of the data access and compared with the proposed ACTS considering the cost and data access completion time. The experiments are carried out using the CloudSim [15] tool. The classes of the CloudSim simulator have been extended (overridden) to utilize the newly written algorithm. The simulator CloudSim opens the possibility of evaluating the hypothesis prior to software development in an environment which can reproduce tests. Specifically, in case of cloud computing where the access to the infrastructure incurs payments in real currency, a simulation-based approach allows cloud customers to test their services in repeatable and controllable environment. Additionally it allows tuning the performance bottlenecks before the deployment on real clouds. The efficiency of the approaches is compared in terms of computation cost, communication cost, execution time, CPU utilization, and bandwidth.

The numbers of tasks and VMs considered are flexible to user requirements which mean the user provides memory, mips, and bandwidth values which are randomly utilized in the VM. The appropriate determination of the characteristics of the VM and the tasks is highly recommended for obtaining

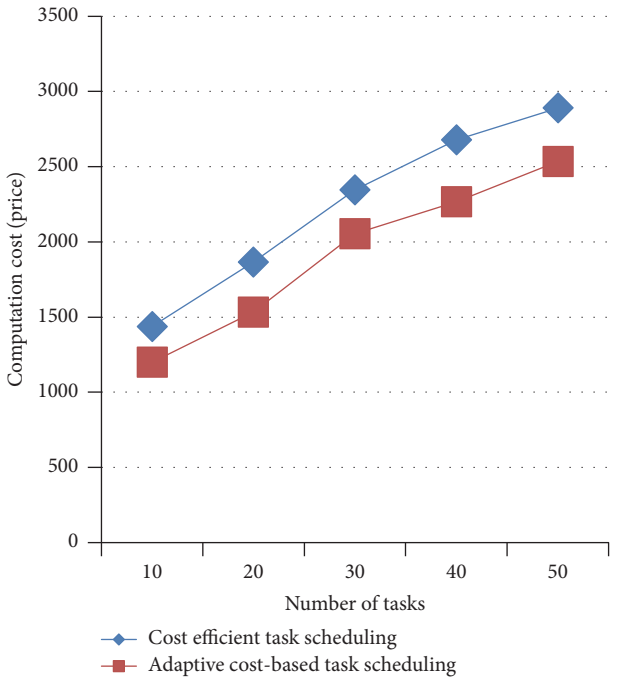

Figure 2: Comparison of computation cost.

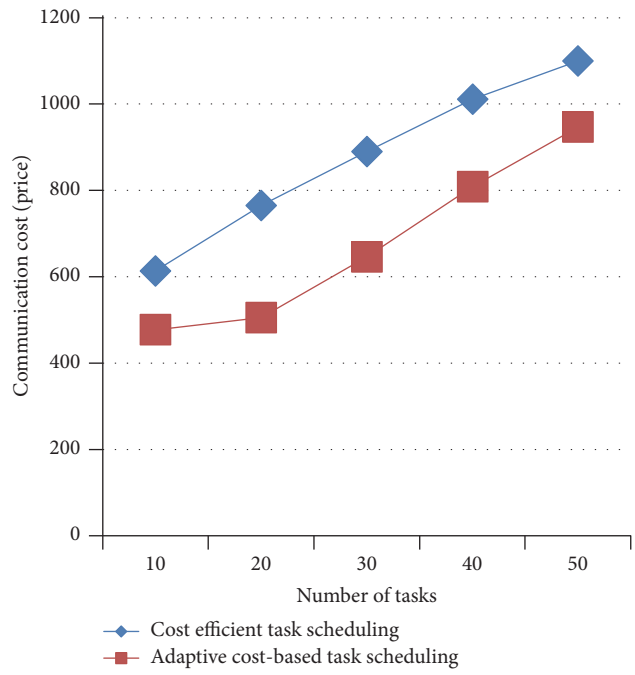

Figure 3: Comparison of communication cost.

desired performance evaluation results. The VM characteristics are as follows: ram (256, 312, 712, and 856) bytes; mips (330, 370, and 400); bandwidth (700, 750, 800, and 900) bits per second (bps). Likewise, the *I/O intensive tasks* are taken as follows: length (4, 8, 11, 5, 3, 9, and 10); memory (256, 312, 378, 280, 436, 553, and 375) bytes. An I/O intensive task performs the function of reading the input/output data and writes them onto the files. These values are user provided values and suppose if the number of VMs is 10 then the combination of ram, mips, and bandwidth is chosen randomly. For example, in case of the ram for 10 VMs, the one possible set of values would be 256, 312, 712, 856, 256, 312, 712, 856, 256, and 312, respectively.

4.1. Computation Cost. Computation cost is the cost that is required for utilizing the resources for computation of the I/O requests for the data access. It can be computed using (7).

Figure 2 shows the comparison of the existing cost efficient task scheduling without considering the completion time and the cost with the proposed adaptive cost-based task scheduling (ACTS) with considering the completion time and the cost in terms of the computation cost. In the x-axis, the number of tasks is taken while along the y-axis the computation cost (price) is taken. When the number of tasks is 50, the cost efficient task scheduling has computation cost of 2890 but the proposed ACTS has 2534.8. Thus the proposed ACTS provides better scheduling with minimal computation cost.

4.2. Communication Cost. Communication cost is the cost that is required for utilizing the resources for I/O requests and responses between the data center and the VM for the data access. It can be calculated using (9).

Figure 3 shows the comparison of the existing cost efficient task scheduling without considering the completion time and the cost with the proposed adaptive cost-based task scheduling (ACTS) with considering the completion time and the cost in terms of the communication cost. In the x-axis, the tasks are taken while along the y-axis the communication cost (price) is taken. When the number of tasks is 50, the existing cost efficient task scheduling has communication cost of 1100 but the proposed adaptive cost-based task scheduling has 946.6. This shows that the proposed ACTS consumes less cost than the existing scheme.

4.3. Execution Time. The execution time is the time required to process a task in a VM. The execution time is estimated as the product of number of cycles for executing per instruction, time per cycle, and the number of instructions.

Figure 4 shows the comparison of the existing cost efficient task scheduling without considering the completion time and the cost with the proposed Adaptive cost-based task scheduling (ACTS) with considering the completion time and the cost in terms of the execution time. In the x-axis, the tasks are taken while along the y-axis the execution time in milliseconds (ms) is taken. When the number of tasks is 50, the existing cost efficient task scheduling has execution time of 4.978 ms but the proposed ACTS has 2.56 ms. This shows that the proposed ACTS reduces the time taken for the overall process.

4.4. CPU Utilization. CPU utilization refers to the usage of processing resources or the amount of work handled by a CPU. CPU utilization varies depending on the amount and type of managed computing tasks. It is estimated using (11).

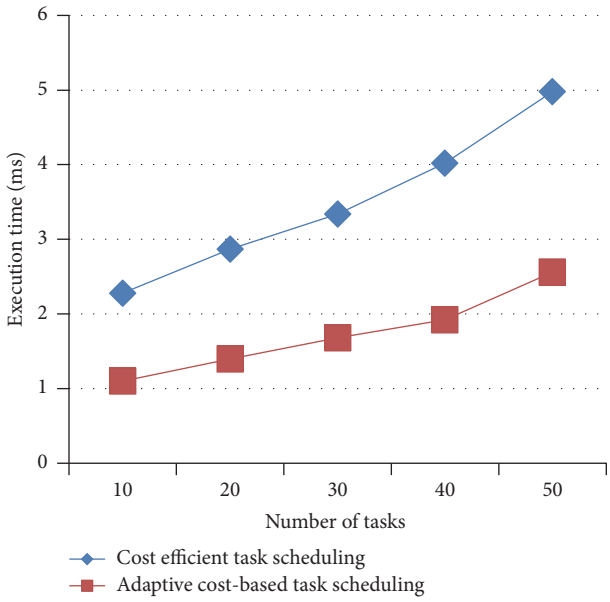

Figure 4: Comparison of execution time.

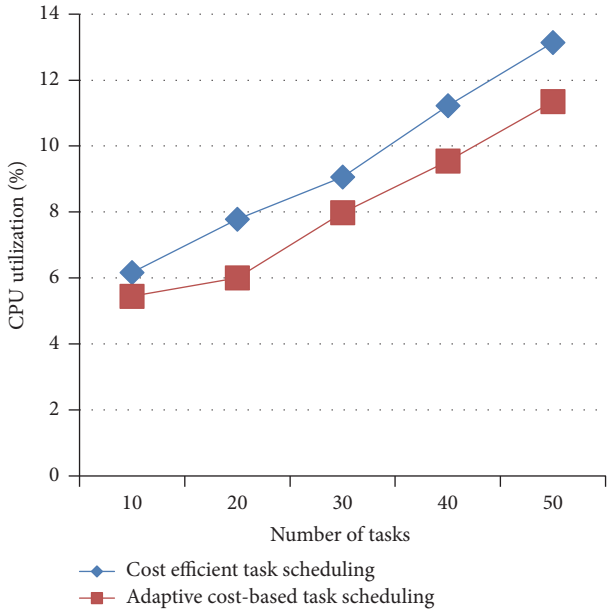

Figure 5: Comparison of CPU utilization.

Figure 5 shows the comparison of the existing cost efficient task scheduling with the proposed adaptive cost-based task scheduling (ACTS) in terms of the CPU utilization. In the x-axis, the number of tasks is taken while along the y-axis the CPU utilization in % is taken. When the number of tasks is 50, the existing cost efficient task scheduling has CPU utilization of 13.14% but the proposed ACTS has 11.345%. This shows that the proposed ACTS has less CPU utilization.

4.5. Bandwidth Utilization. Bandwidth is the amount of data that can be transmitted in a fixed amount of time. It is given in bits per second (bps). It is estimated using (12).

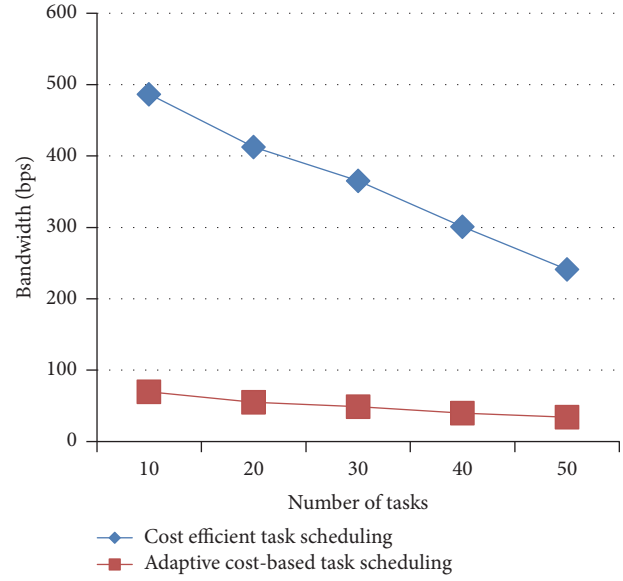

Figure 6: Comparison of bandwidth utilization.

Figure 6 shows the comparison of the existing cost efficient task scheduling with the proposed Adaptive cost-based task scheduling (ACTS) in terms of the bandwidth. In the x-axis, the number of tasks is taken while along the y-axis the bandwidth in bps is taken. When the number of tasks is 50, the existing cost efficient task scheduling has bandwidth of 240.98 bps but the proposed ACTS has 34.123 bps.

Thus from the experimental results it is clear that the proposed Adaptive cost-based task scheduling (ACTS) which considers the completion time and computation cost and communication cost is efficient compared to the existing cost efficient task scheduling.

5. Conclusion

Scheduling tasks in cloud computing with reduced delay and effective cost management are a challenging task. Hence in this paper, adaptive cost-based task scheduling (ACTS) is proposed considering the data access completion time and the cost for data access. By considering these two factors, the data can be fetched from the data centers effectively and the scheduling performance can be improved. The approach focuses on providing data access for executing each task with maintained costs. Experimental results also show that the proposed adaptive cost-based task scheduling provides better performance in terms of execution time, computation cost, communication cost, and bandwidth and CPU utilization when compared with existing cost-efficient task scheduling approach.

In this paper, the task scheduling is performed for the already determined task demands and it is quite challenging to schedule tasks with undetermined demands. This could be performed by utilizing efficient resource provisioning techniques in the future. The cost for regeneration of datasets is not computed in ACTS but it is not efficient for exception cases which should be considered in the future researches.

Moreover, the load-balancing problems are also needed to be resolved for providing efficient cloud computing services which would be our future scope of research.

Competing Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] P. Mell and T. Grance, "The NIST definition of cloud computing," *National Institute of Standards and Technology*, vol. 53, no. 6, p. 50, 2009.

[2] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: state-of-the-art and research challenges," *Journal of Internet Services and Applications*, vol. 1, no. 1, pp. 7–18, 2010.

[3] K. Nanath and R. Pillai, "A model for cost-benefit analysis of cloud computing," *Journal of International Technology and Information Management*, vol. 22, no. 3, article 6, 2013.

[4] J. Sahni and D. Vidyarthi, "A cost-effective deadline-constrained dynamic scheduling algorithm for scientific workflows in a cloud environment," *IEEE Transactions on Cloud Computing*, 2015.

[5] C. W. Tsai, W. C. Huang, M. H. Chiang, M. C. Chiang, and C. S. Yang, "A hyper-heuristic scheduling algorithm for cloud," *IEEE Transactions on Cloud Computing*, vol. 2, no. 2, pp. 236–250, 2014.

[6] X. Zhu, C. Chen, L. T. Yang, and Y. Xiang, "ANGEL: agent-based scheduling for real-time tasks in virtualized clouds," *IEEE Transactions on Computers*, vol. 64, no. 12, pp. 3389–3403, 2015.

[7] Z. Zhu, G. Zhang, M. Li, and X. Liu, "Evolutionary multi-objective workflow scheduling in cloud," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 5, pp. 1344–1357, 2016.

[8] Q. Zhang, M. F. Zhani, Y. Yang, R. Boutaba, and B. Wong, "PRISM: fine-grained resource-aware scheduling for MapReduce," *IEEE Transactions on Cloud Computing*, vol. 3, no. 2, pp. 182–194, 2015.

[9] X. Zhu, L. T. Yang, H. Chen, J. Wang, S. Yin, and X. Liu, "Real-time tasks oriented energy-aware scheduling in virtualized clouds," *IEEE Transactions on Cloud Computing*, vol. 2, no. 2, pp. 168–180, 2014.

[10] S. T. Maguluri and R. Srikant, "Scheduling jobs with unknown duration in clouds," *IEEE/ACM Transactions on Networking*, vol. 22, no. 6, pp. 1938–1951, 2014.

[11] X. Zuo, G. Zhang, and W. Tan, "Self-adaptive learning pso-based deadline constrained task scheduling for hybrid iaas cloud," *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 2, pp. 564–573, 2014.

[12] S. Su, J. Li, Q. Huang, X. Huang, K. Shuang, and J. Wang, "Cost-efficient task scheduling for executing large programs in the cloud," *Parallel Computing*, vol. 39, no. 4-5, pp. 177–188, 2013.

[13] J.-W. Lin, C.-H. Chen, and C.-Y. Lin, "Integrating QoS awareness with virtualization in cloud computing systems for delay-sensitive applications," *Future Generation Computer Systems*, vol. 37, pp. 478–487, 2014.

[14] D. Yuan, Y. Yang, X. Liu et al., "A highly practical approach toward achieving minimum data sets storage cost in the cloud," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 6, pp. 1234–1244, 2013.

[15] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software—Practice and Experience*, vol. 41, no. 1, pp. 23–50, 2011.