Cluster Computing (2020) 23:1073–1083
https://doi.org/10.1007/s10586-019-02978-2

A collaboration of deadline and budget constraints for task scheduling in cloud computing

Mokhtar A. Alworafi[1] · Suresha Mallappa[1]

Received: 12 January 2019 / Revised: 19 May 2019 / Accepted: 23 August 2019 / Published online: 29 August 2019

Abstract

Cloud computing has become the most attractive platform compared to grid computing, that offers several services such as infrastructure as a service, platform as a service, and software as a service, where the users can consume these services on the cloud and pay based on their consumption and on the fulfilment of Quality of Service (QoS) constraints such as deadline and budget. Hence, to schedule the tasks effectively, cost monetary must be considered while optimising implementation time performance under users' defined constraints. In this paper, the Deadline Budget Scheduling (DBS) model is proposed to execute the users' tasks on Virtual Machines (VMs) under the QoS constraints at less execution time. In our proposal, users' tasks will be assigned to appropriate VM which meets either of the two constraints namely (deadline and budget) or one of the constraints based on user satisfaction. Makespan and cost are calculated to evaluate our proposed DBS model with state of the art algorithms. The experimental results illustrate that DBS outperforms other algorithms by minimizing the makespan and cost.

Keywords Cloud computing · Quality of Service · DBS · Makespan · Cost

1 Introduction

During the last few years, cloud computing has altered the form of all computing paradigms by providing a lot of services over the internet [1]. It is referred to services and applications are grouped as Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Software as a Service (SaaS) [2] and can be offered as services and a pricing plan is used for these resources by pay-as-per-use [3]. If the desired performance is not accomplished, the users will hesitate to make payment, therefore, it is necessary to fulfil QoS to users for the given resources [4]. The inherent heterogeneity, highly dynamic nature inevitably, and multidomain characteristic also lead to the difference in resource cost, computing capacity, memory capacity, communication capacity, and storage capacity [5]. The heterogeneous resources can be local or geographically distributed, which are used for execution of computationally intensive applications [6]. Around the world, there are several cloud providers that have variable data centers that facilitate the managing, setting up, and maintaining of the private storage infrastructure like Google Cloud storage, Microsoft Azure, and Amazon S3 [7].

Cloud consumer always negotiates with the Cloud Service Provider (CSP) to sign on Service Level Agreements (SLA) prior to any service use or resource rent [8]. In computing model, a consumer pays only for what he consumes from a service and resources that he needs. A consumer is concerned with two most important factors: cost and time. Thus, the trade-off problem for scheduling the tasks is cost and time [9]. Task scheduling is an important issue in cloud computing where it is responsible for assigning the tasks to the suited resources [10]. User's tasks are submitted to different cloud resources based on the computational cost and time using task constraints (deadline, budget). The running costs can be optimized by a provider, as well as guarantee a high QoS by ensuring equitable access to resources shared by all users [11]. Therefore, the execution of tasks must take into account a

✉ Mokhtar A. Alworafi
mokhtar119@gmail.com

Suresha Mallappa
sureshasuvi@gmail.com

[1] DoS in Computer Science, University of Mysore, Mysore, India

scheduling on the heterogeneous environment in such a way that the user-defined deadline is achieved, and the monetary costs of the tasks executed also do not exceed the user-defined budget. Due to higher leasing costs of resources, the execution costs and utilization of leased resources gain higher importance [12]. In general, task scheduling should fulfil the following requirements of scheduling aims:

1. *QoS/SLA constraint* The SLA clearly states the QoS requirements for scheduling the tasks by specifying the application deadline, task scheduling expenditure budget, service security, and system reliability. Hence the QoS target constraints of task scheduling should be taken into account to achieve QoS requirement and obtain ultimate service revenue or profit to guarantee commercial success.
2. *Service revenue* Cloud computing environment consists of thousands and even millions of servers, causing higher input costs. So, for task scheduling in cloud computing, some economic principles and practices are used, which make the scheduling more efficient in performance and accordingly looks more reasonable. Therefore, for promoting a healthier and more sustainable growth of cloud computing market, meeting and QoS constraints, satisfying resource requests and ways to increase service revenue have become other critical objectives of task scheduling for providers of cloud service.

Under such a background, how can the tasks be scheduled for efficiency by executing the tasks with low payment costs and with less completion time (makespan). Meanwhile, maximizing resource utilization in cloud computing has become a growing concern and is one of the technical difficulties in the academic field [13].

In this paper, the Deadline Budget Scheduling (DBS) model is introduced in which the users will submit their tasks with their budget and deadline to the data centers and then a priority would be defined upon the users' need.

In DBS model, when the task has deadline and budget constraints, it is considered as high priority, while if the task has only deadline constraint it is considered as fair priority; finally with only budget constraint it is considered as less priority for all the users' tasks.

The main contributions of this paper are:

(a) The proposal of DBS model addresses task requirements which are constrained to time and cost.

(b) The DBS model attains less completion time for executing the resources upon two or one of the two constraints along with increasing the service provider revenue, and in addition to minimizing the number of violation of submitted tasks.

(c) Experimental results demonstrate that DBS model is capable of assigning the tasks to suitable resources with high quality in terms of meeting the deadline and budget constraints. The makespan is minimized up to 31.5% and cost up to 31.5% when compared to state-of-the-art algorithms.

The reminder of this paper is organized as follow. Section 2 explains the related work. Section 3 defines the problem. The proposed work of DBS model is presented in Sect. 4. Section 5 presents results of experiments. Performance evaluation is presented in Sect. 6. Section 7 concludes the work.

2 Related work

The scheduling of the tasks over the cloud environment resources becomes a very complicated problem, where there are a lot of metrics such as makespan and utilization of the resources which influence the scheduling. Constraints such as deadline and budget must also be taken into account. So these problems are tackled by scheduling algorithms regarding QoS constraints (deadline, budget, deadline and budget).

2.1 Deadline constraint

Deadline constraint is a major constraint which the schedule algorithms must take into consideration to make their proposed schedulers as efficient as possible.

Pop et al. [14] scheduled a set of periodic tasks by estimating number of resources while taking into account the execution and data transfer costs. They decoupled the task upon coming from created task, execution and scheduling, set of actions as network peer- to- peer relation or client–server over the cloud. The experiments depended on deadline constraint that influences the tasks scheduling.

Shin et al. [15] improved a conservative backfilling algorithm by using the Earliest Deadline First (EDF) algorithm and the Largest Weight First (LWF) algorithm in which, firstly all the jobs arriving at data centers are registered and sorted in ascending order to maintain job's high priority. Then the largest backfill job is selected that maintains the deadline constraint. Their algorithm improved the system performance regarding deadline.

Deldari et al. (2016) proposed to partition the workflow into clusters for minimizing the execution costs upon the users' deadline constraint which considered the main factors in their work by using flexible and an extendable scoring method to select the best cluster groups for meeting their objectives. The proposed algorithm proved to great by minimizing the cost of resources while preserving the users' deadline constraint [11].

Panda et al. [1] introduced three allocation-aware task scheduling algorithms on multi cloud environment based on extended Min–Min and Max–Min algorithms for the multi cloud environment. The proposed algorithms depended mainly on three phases named matching, allocation and scheduling. They evaluated their algorithms by using various benchmark and they measured the makespan, average utilization and throughput and proved the efficiency of their algorithms.

2.2 Budget constraint

Budget is another important QoS constraint. It is the cost threshold that the users want to pay for using cloud resources and services. Each cloud user has a budget for the resources he uses from cloud to execute his tasks. Many researchers addressed this issue and improved the scheduling algorithms regarding budget QoS constraint.

Our previous work (2017) is a Scheduling Cost Approach (SCA) to distribute tasks among available resources based on priority which was determined by the users as well as to improve load balancing. Then the cost of resources like CPU, bandwidth, RAM, and storage is calculated. The experimental results illustrated that SCA outperformed the state of art algorithms [16].

Thanasias et al. [17] discussed task scheduling and resource allocation problem for implementing tasks in IaaS clouds; a novel provisioning and scheduling algorithm is presented to execute tasks under budget constraint while reducing the slowdown. The experimental results illustrated that their proposed algorithms minimized the slowdown in execution time to 70%.

In the execution of scientific applications on the cloud systems, minimizing the schedule length is the main quality of service requirements. Chena et al. [18] proposed minimizing the schedule length using the budget level MSLBL algorithm to determine the cloud processors which satisfy the budget and reduce the application schedule length. The first problem was solved by transferring the tasks with their budget, while the second problem was solved by heuristically scheduling the task by minimizing time complexity.

Rodriguez et al. [19] introduced a scheduling algorithm which optimized the task workflow execution time regarding the budget constraint. The experiments proved that this scheduling algorithm has faster execution time and more effective performance on cloud resources where their proposed algorithm is capable of generating high-quality schedules to meet the budget constraint.

2.3 Budget and deadline constraint

Some researchers have proposed merging the two QoS constraints (deadline, budget) in schedule algorithms for more novelty of the algorithms. All the cloud users need to maintain some QoS constraints as the best and cheapest resource cost to execute their applications as fast as possible.

Saxena et al. [20] classified the tasks with deadline and minimum cost as a constraint. Then priority of fairness, and dynamic optimization is applied. The priority is implemented in round robin algorithm where there are three queue priorities (high, mid, low) as per weights allocated to the tasks. The performance results illustrated that there are benefits to users, cloud providers and also provided fairness at priority level.

Arabnejad et al. [21] tried to minimize the time complexity of the scheduling algorithm. They proposed Deadline-Budget Constrained Scheduling (DBCS) which considered some of the QoS parameters time and cost that are the main issues in the proposed algorithm. Radically, the DBCS algorithm searched for fine schedule mapping which satisfies the users' specified deadline and budget constraints. The implementation of DBCS consumed approximately 4% of time in the worst cases and it achieved low complexity time.

Khorsand et al. [22] proposed Adaptive Two Stage Deadline Constrained Scheduling (ATSDS) algorithm. The first stage depended on run-time circumstances in cloud computing. It is virtual machines number and average available bandwidth that is dynamically fragmented. The second stage is related to assigning the workflow fragments created to be executed into VMs based on VM capacity and deadline constraint.

Arabnejad et al. [23] considered the profit of provider and they proposed Multi-QoS Profit-Aware scheduling algorithm (MQ-PAS) to assign each job with its budget priority. The performance evaluation results showed that the MQ-PAS increased the profit of provider and achieved success rates of completion jobs.

Verma et al. [24] proposed a Hybrid Particle Swarm Optimization (HPSO). The proposed hybrid is a non-dominance sort strategy, which considers two discordant objectives, makespan under deadline and cost under budget. In addition to these conflicting aims the energy consumption of created workflow scheduling is reduced.

3 Problem definition

Tasks scheduling in cloud computing could be more efficient and generate high performance if it is good in assigning appropriate tasks on cloud VMs. In this definition, it is assumed that the cloud environment is hosted in a data center consisting of heterogeneous servers, which in turn hosts several VMs. The VMs configurations may have various processing capacities, memory sizes, and the communication links may have different bandwidths as well as different storage capacities. In this paper, the Deadline Budget Scheduling (DBS) model is proposed, which minimizes execution time and cost while constrained to a user defined deadline and budget. The task scheduling is considered with the following scenarios: the VMs are heterogeneous, which have different efficient performance depending on the resources provisioned to them. In order to formulate DBS model, the Dc as data center consisting of several H hosts is defined, each host contains a set of VMs. The resources of VM is denoted as: R_{VM} = {R_C, R_B, R_M, R_S}, where R_C is the CPU capacity which is represented as million instructions per second (MIPS). R_B is the amount of bandwidth while R_M refers to the memory of VM finally, R_S is storage of VM. A task's processing cost will vary according to the tasks being assigned to different VMs and their costs. On the other hand, the communication cost between two VMs is changing because of bandwidth diversity between two different VMs; also, the cost of memory and storage is different from one VM to another so that a cost of each R_{VM} is denoted as C_R = {C_{RC}, C_{RB}, C_{RM}, C_{RS}} respectively. Each VM implements independent tasks which are denoted as T that has several attributes such as deadline, budget, length, input file size and output file size denoted as T = {TD, TB, Len, Fit, Fot} respectively. The target of this work is to minimize the makespan and execution cost considering user satisfaction.

3.1 Parameters' definition

Variables and parameters used in deadline and budget scheduling model are defined in Table 1.

4 Proposed work

This work highlights deadline and budget constraints in task scheduling to improve the QoS in association with two basic parameters i.e., minimize execution time and cost of executed tasks, and finally enhance the revenue of services as well as resource utilization of the VMs and the host(s).

4.1 Task Constraint Type

In this proposal each task has a constraint type based on user satisfaction. In the proposed DBS model, there are three types of constraint type assigned as in Eq. 1:

$$\text{Task Constraint Type} = \begin{cases} D \text{ and } B & \text{if the constraints are Deadline and Budget} \\ D & \text{if the constraint is Deadline} \\ B & \text{if the constraint is Budget} \end{cases} \tag{1}$$

4.2 Clustering of resources

The available cloud resources can be clustered based on user's satisfaction in the following way:

1. First cluster consists of a set of VMs which meet the budget and deadline constraints.

Table 1 Abbreviations used in the proposed DBS model

Symbol	Explanation	Symbol	Explanation
Dc	Data center	U	Resource utilization
H	Host	n	Total number of tasks
VM	Virtual machine	EX	Execution time
T	Task	C_{RC}	Cost of CPU
Len	Length of task	C_{RB}	Cost of resource bandwidth
TB	Task budget	C_{RM}	Cost of resource memory
TD	Task deadline	C_{RS}	Cost of resource storage
Fit	File input size or the bit amount of task	GC	Gain cost
Fot	File output size or the bit amount of task	EDT	Expected data transfer
R_{VM}	Resource of virtual machine	ECT	Expected complete time
R_C	Resource of CPU	ECG	Expected CPU gain
R_B	Resource of bandwidth	EDG	Expected data gain
R_M	Resource of memory	m	Number of virtual machines
R_S	Resource of storage	DT	Data transfer
P	Number of previous tasks in specific VMj		

2. Second cluster consists of a set of VMs which meet deadline constraint.
3. Third cluster consists of a set of VMs which meet only the budget constraint.

When looked at Fig. 1, the scheduling DBS model appears consisting of resources and cloud users who send their tasks including the constraints (deadline, budget). Each task is checked for detecting its constraint type based on user satisfaction and it will be sent to a cluster of resources that is capable of fulfilling the task constraint. If the constraint is both deadline and budget then the task will be implemented in cluster one; if the constraint is a deadline the task will be implemented in cluster two, or else if the task's constraint type is budget it will be implemented in cluster three. After that, at the same cluster, other attributes of the task (length, file size) is also included in the selection for achieving minimum makespan and minimum cost as illustrated in the next section.

4.3 Scheduling strategy

In this paper, the DBS task scheduling depends on two levels for binding the tasks to suitable resources, so the determination of task constraint type is known as level one, while at level two, the DBS model determines the task's requirements based on the task attributes as illustrated in Fig. 2.

In level one, each task will be checked to detect its constraint type as illustrated in Eq. 1, and based on the constraint each task will be sent to the appropriate cluster as mentioned earlier. Next, in level two, the attributes of the same task will be compared. This step of comparison is for finding the best resource in one cluster, therefore, the length of the task (Len) and the input file size (Fit) are compared.

If the constraint type is D & B, the deadline and budget constraints should be met. The proposed DBS model will obtain the appropriate VM in the first cluster for implementing the task as mentioned in Eq. 2.

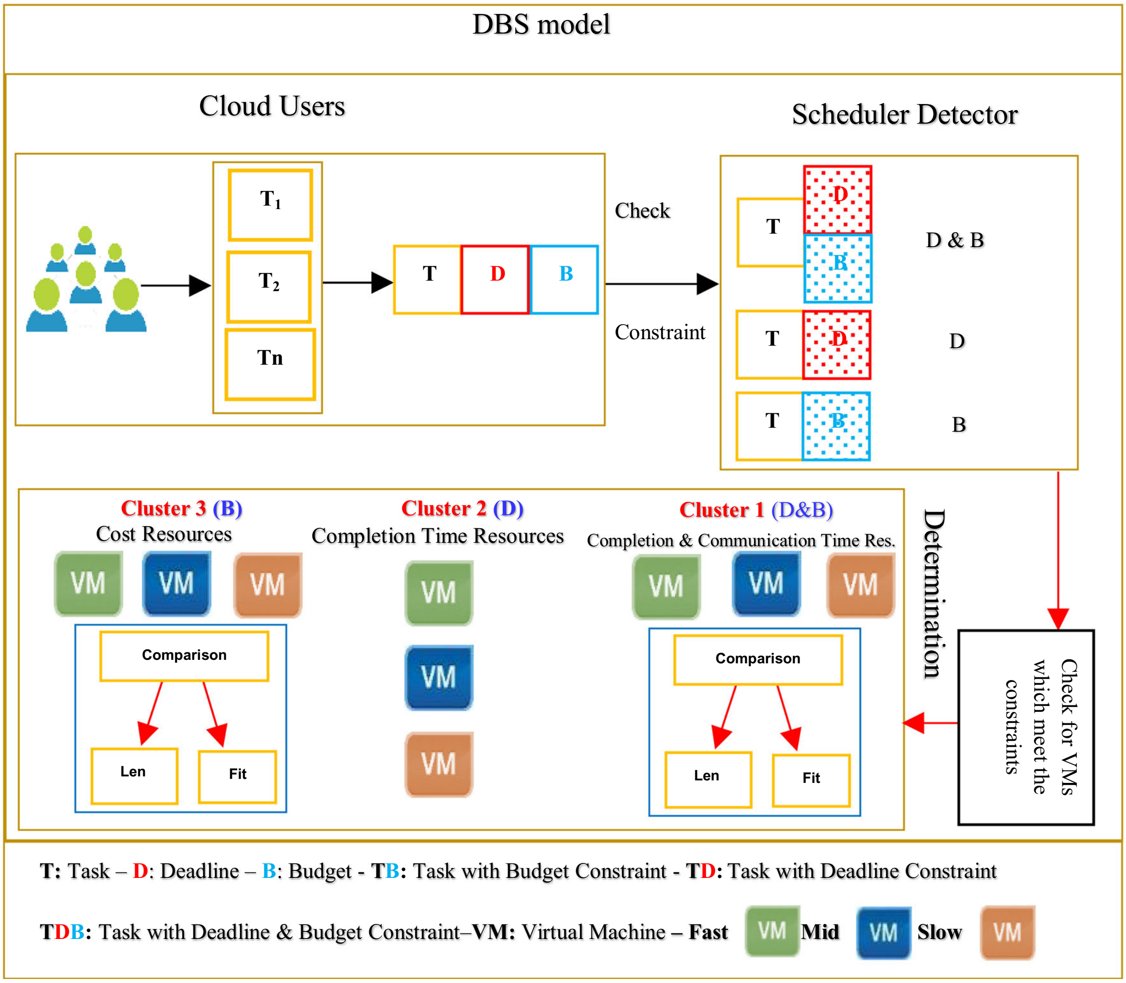

DBS model
Cloud Users
Scheduler Detector
T1
T2
Tn
T
D
B
Check
Constraint
T
D
B
D & B
T
D
D
T
B
B
Cluster 3 (B)
Cost Resources
VM
VM
VM
Comparison
Len
Fit
Cluster 2 (D)
Completion Time Resources
VM
VM
VM
Cluster 1 (D&B)
Completion & Communication Time Res.
VM
VM
VM
Comparison
Len
Fit
Determination
Check for VMs which meet the constraints
T: Task – D: Deadline – B: Budget - TB: Task with Budget Constraint - TD: Task with Deadline Constraint
TDB: Task with Deadline & Budget Constraint–VM: Virtual Machine – Fast VM Mid VM Slow VM

Fig. 1 Proposed deadline budget scheduling model

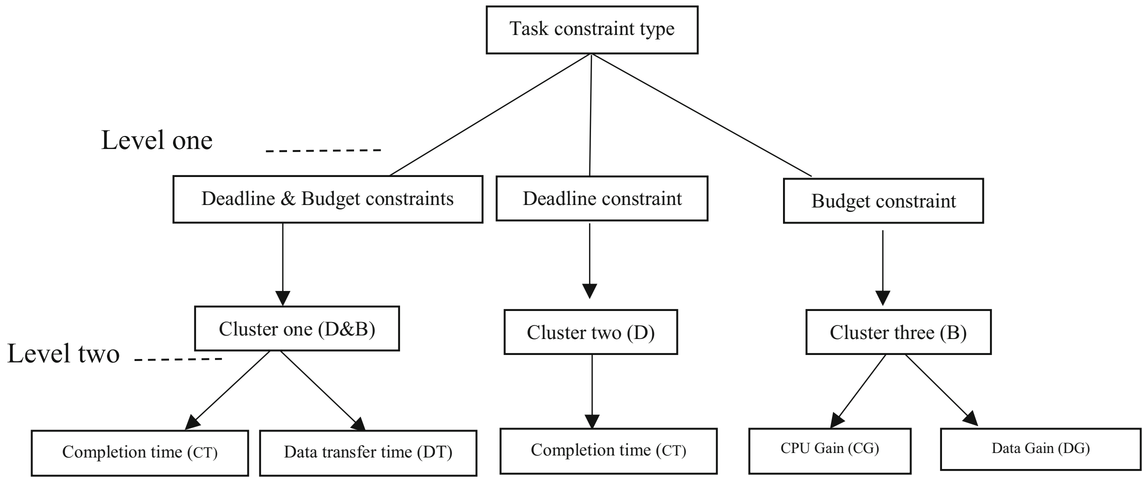


Fig. 2 Organogram of the DBS model

$$\text{D\&B Constraint Type} = \begin{cases} \text{completion time if Len} > \text{Fit} \\ \text{data transfer time if Len} < \text{Fit} \end{cases} \tag{2}$$

When the length of the task is larger than the input file size, the completion time of task must be considered, else the data transfer time must be considered.

In the first cluster, the VM which achieves less expected completion time will be selected. The expected completion time is calculated based on Eq. 3.

$$\text{ECT} = \sum_{i=1}^{p} \text{EX}_\text{i} + \text{EX of current task} \tag{3}$$

where p is the number of previously assigned tasks in specific VM_j, EX is the execution time of task, which is calculated based on Eq. 4 [25].

$$\text{EX} = \frac{Len}{R_C} \tag{4}$$

In cases when the length of the task is lesser than its input file size, the task will be assigned to a VM that return less data transfer time among all VMs in a cluster. Data transfer time is calculated as mentioned in Eq. 5.

$$\text{DT} = \frac{Fit + Fot}{R_B} \tag{5}$$

The expected data transfer time of task in each VM is calculated based on Eq. 6.

$$\text{EDT} = \sum_{i=1}^{p} \text{DT}_\text{i} + \text{DT of current task} \tag{6}$$

So the VM which has less EDT will be selected to implement the task.

In the proposed DBS model, the second constraint type is D, which focuses on the deadline constraint only. The completion time of task is calculated for the second cluster and the task will be sent to VM that has lesser completion time. The expected completion time is as mentioned in Eq. 3.

The last constraint type of the proposed DBS model is B. The task requirements is obtaining the cost of execution in third cluster as mentioned in Eq. 7 which compares between the length of task and input file size as below:

$$\text{B Constraint Type} = \begin{cases} \text{CPU Gain if Len} > Fit \\ \text{Data Transfer Gain if Len} < Fit \end{cases} \tag{7}$$

When the length of the task is the largest, assuming that faster machine is used by selecting VM that has less expected CPU cost based on Eq. 8.

$$\text{ECG} = \frac{Len_i}{R_C} \times C_{RC} \tag{8}$$

On the other hand, if the file size is largest, the data transfer cost is kept high. The task will be mapped to faster VM as allotted in Eq. 9.

$$\text{EDG} = \frac{Fit_i + Fot_i}{R_B} \times C_{RB} \tag{9}$$

Finally, the gain cost of task will be calculated based on Eq. 10 [16].

$$\text{Gain Cost (GC)} = \left(\frac{Len_i}{R_C} \times C_{RC} + \frac{\text{Fit} + Fot_i}{R_B} \times C_{RB} + \frac{Fit_i + Fot_i}{R_M} \times C_{RM} + \frac{Fit_i + Fot_i}{R_S} \times C_{RS} \right) \tag{10}$$

The status of all VMs is updated, and the scheduling model computes the optimal solution to select the appropriate scheduling decision.

4.3.1 Case study

As mentioned before, the tasks come with different constraints types such as D & B, D, B. The first type is tackled as a simple case study to clarify the concept of DBS model. Assume there are three VMs, two tasks having D&B constraints with attributes as mentioned in Table 2.

Based on the above assumption, the completion time of task 1 must be considered as mentioned in Eq. 2 and is calculated based on Eqs. 3 and 4.

$$\text{ECT in VM1} = \text{length of task1}/\text{MIPS of VM1} = \frac{2000}{1000} + 0 = 2$$

$$\text{ECT in VM2} = \text{length of task1}/\text{MIPS of VM1} = \frac{2000}{250} + 0 = 8$$

$$\text{ECT in VM3} = \text{length of task1}/\text{MIPS of VM1} = \frac{2000}{600} + 0 = 3.33$$

With compared these results, the task 1 will be assigning to VM1 which return less ECT.

But looking at task 2 attributes, the lesser EDT is considered, because, the length of task 2 is less than the file input data for the same task, and is calculated based on Eqs. 5 and 6:

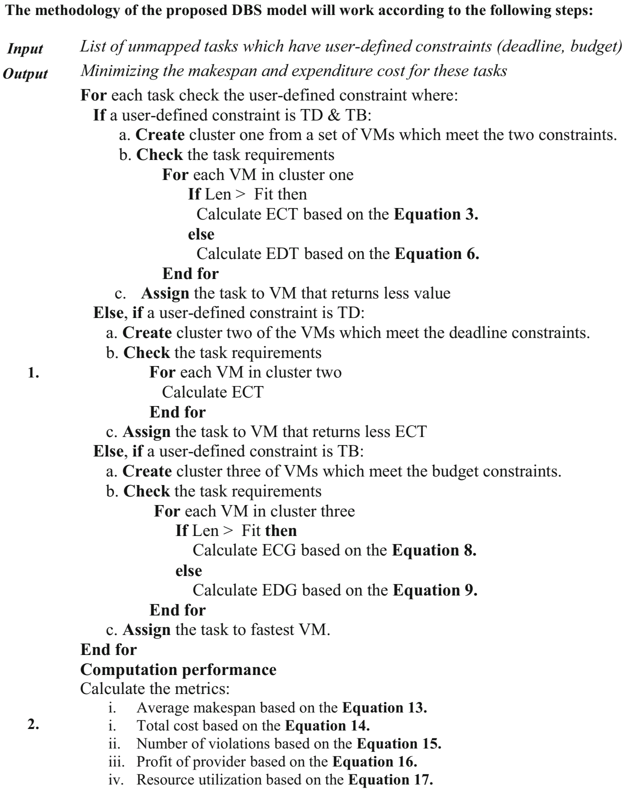
$$\begin{aligned}\text{ECT in VM1} &= \frac{Fit + Fot}{R_B}\text{of current task} + \frac{Fit + Fot}{R_B}\text{ of pervious task}\\ &= \frac{3000 + 200}{300} + \frac{400 + 250}{300} = 12.83\\ \text{ECT in VM2} &= \frac{Fit + Fot}{R_B}\text{of current task} + \frac{Fit + Fot}{R_B}\text{ of pervious task}\\ &= \frac{3000 + 200}{200} + 0 = 16\\ \text{ECT in VM3} &= \frac{Fit + Fot}{R_B}\text{of current task} + \frac{Fit + Fot}{R_B}\text{ of pervious task}\\ &= \frac{3000 + 200}{100} + 0 = 32\end{aligned}$$

Task 2 will be assigned to VM1 taking into account that task 1 was also sent to the same VM in the previous scheduling, where VM1 also returns less EDT among all VMs.

Other constraint types will be handled the same way by taking into account the constraint type and task attributes.

4.4 The proposed model methodology

The methodology of the proposed DBS model will work according to the following steps:



The methodology of the proposed DBS model will work according to the following steps:

```
Input    List of unmapped tasks which have user-defined constraints (deadline, budget)
Output   Minimizing the makespan and expenditure cost for these tasks
         For each task check the user-defined constraint where:
           If a user-defined constraint is TD & TB:
               a. Create cluster one from a set of VMs which meet the two constraints.
               b. Check the task requirements
                     For each VM in cluster one
                        If Len >  Fit then
                          Calculate ECT based on the Equation 3.
                        else
                          Calculate EDT based on the Equation 6.
                     End for
               c.  Assign the task to VM that returns less value
           Else, if a user-defined constraint is TD:
             a. Create cluster two of the VMs which meet the deadline constraints.
             b. Check the task requirements
1.                 For each VM in cluster two
                     Calculate ECT
                   End for
             c. Assign the task to VM that returns less ECT
           Else, if a user-defined constraint is TB:
             a. Create cluster three of VMs which meet the budget constraints.
             b. Check the task requirements
                    For each VM in cluster three
                       If Len >  Fit then
                          Calculate ECG based on the Equation 8.
                       else
                          Calculate EDG based on the Equation 9.
                   End for
             c. Assign the task to fastest VM.
         End for
         Computation performance
         Calculate the metrics:
               i.   Average makespan based on the Equation 13.
2.             i.   Total cost based on the Equation 14.
               ii.  Number of violations based on the Equation 15.
               iii. Profit of provider based on the Equation 16.
               iv.  Resource utilization based on the Equation 17.
```

5 Results of experiments

The most appropriate toolkit which can simulate the cloud environment efficiently is CloudSim toolkit. CloudSim is an extensible model and simulation framework. It is a model for experiments simulating the cloud computing infrastructure and services. In order to evaluate the performance of the proposed DBS model, many experiments are implemented based on the CloudSim toolkit which has

Table 2 Task and VMs attributes

VMs	Attribute	
	R_C	R_B
VM 1	1000	300
VM 2	250	200
VM 3	600	100

Tasks	Attribute		
	Length	fit	Fot
Task 1	2000	400	250
Task 2	2500	3000	200

several entities. The simulator consists of data center entity which represents cloud environment and is partitioned into two hosts. Also the entities: Datacenter Broker, VM and cloudlet entities are used. A central piece of DBS model is the Data center Broker which simulates a provisioning policies and scheduling algorithms. The VM entity is extended to add the cost of each resource. It is assumed that VMs have a single core and executes the tasks in parallel. The tasks are represented using Cloudlet entity which is submitted by a user to the cloud. This entity is extended also to consider the deadline and budget attributes. The Data center Broker administrates the VMs provisioning and the scheduling of tasks. When the tasks are received in cloudlets form, the broker takes a decision to obtain or terminate VMs based on the task scheduling. The Data center Broker reads the list of submitted cloudlets and binds each cloudlet to appropriate VM based on a pre-specified policy. The list of cloudlets is assigned in an iterative way, until all cloudlets are submitted and executed. Each cloudlet submitted must be removed from cloudlets list. The broker also monitors the cloudlets during the runtime to give a report whether cloudlets are executed or not. The simulation software environment is configured from one data center that has two physical machines (Host). Each host has 16 GB RAM, 1 TB storage, 100 GB/s Bandwidth and Time–shared VM scheduling algorithm. First physical Machine is (Quad-core Machine), while second physical Machine is (Dual –Core Machine), with X86 Architecture, Linux operating system, Xen Virtual Machine Monitor (VMM), which supports a speed of 10.000 MIPS. Virtual machines are created, each VM with 10 GB image size, 0.5 GB memory, 1 GB/s bandwidth, one processing element, and 500, 1000, 2000, 3000 MIPS. Time-Shared Scheduler and Xen VMM were used. Tasks are created in different lengths and input file sizes, also with different numbers as 250, 500, 750, and 1000 tasks.

The implementation hardware is on a laptop with these configurations: 2.5 GHz, Corei5 CPU, 4 GB Memory, and 512 GB Hard Disk.

Table 3 Summarizes all the measured factors in DBS model (average makespan, total gain cost, No. of violation, provider profit, and average resource utilization) when executing the users' tasks on Virtual Machines (VMs) under the QoS constraints.

6 Performance evaluation

A primary objective of this work is to assess the performance of the proposed DBS model. For that purpose, key performance metrics are considered to evaluate the efficiency of the model performance from both perspectives, i.e., users and service providers. The makespan is considered as the time required to complete the execution of all tasks as well as the cost of the execution of these tasks. Remaining budget and deadline metrics indicate whether the proposed DBS model can execute the workload without surpassing their constraints.

6.1 Makespan

With respect to the user, the time which is consumed to complete executing the tasks must be reduced. The makespan is the completion time of all tasks which are executed by specific VM, as defined in Eq. 11.

$$\text{Makespan} = \sum_{i=1}^{n} (ECT_i * A\,[i,j]) \tag{11}$$

where n is the total number of tasks, and $1 \leq j \leq m$, $A[i, j]$, a Boolean variable can be defined as follows in Eq. 12 [1]:

$$A[i,j] = \begin{cases} 1 & \text{if assign to } VM_j \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

Then for all VMs, the average makespan is calculated as mentioned in Eq. 13 [26].

$$Avg.Makespan = \frac{\sum_{j=1}^{m} Makespan_j}{m} \tag{13}$$

Through the set of experiments, it is proved that the proposed DBS model is capable of running the tasks according to user satisfaction in heterogeneous environment and minimizes the average makespan in all experiments as compared with other algorithms: GA, Max–Min, Round Robin and SJF, which is clear from Fig. 3.

Table 3 Experiments conducted for tasks with different VMs of DBS model

Exps.	No. of VMs	No. of tasks	Average makespan	Total gain cost	No. of violations	Provider profit	Average resource utilization
1	5	250	1883	576,158	10	1,686,740	1827
2	7	500	2879	1,200,682	18	3,337,891	2927
3	9	750	3246	1,770,457	45	4,975,104	3174
4	11	1000	3659	2,359,776	45	6,646,103	3719

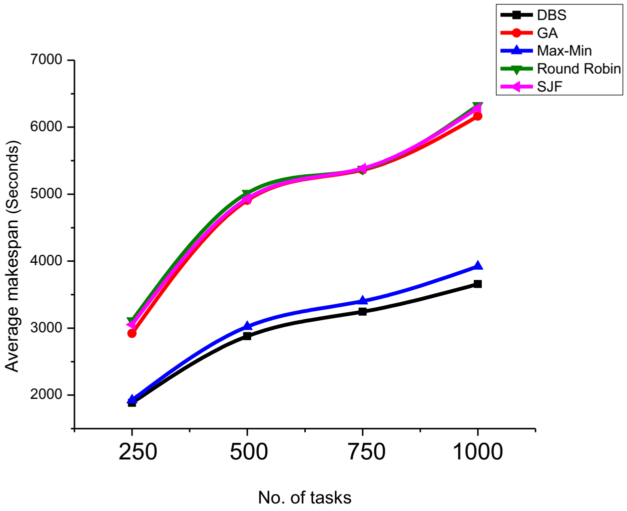


Fig. 3 Comparison of average makespan

Conducting the tasks (250, 500, 750, 1000) on VMs (5, 7, 9, 11), the largest average makespan occurs in SJF algorithm, next Round Robin algorithm, and then GA, followed by Max–Min. Finally, the least makespan is represented in DBS model, which illustrates that the tasks' time execution upon deadline and budget constraints is a less as possible, which in turn maximizes the DBS model performance level.

6.2 Total gain cost

Another performance metric which is used to evaluate the proposed DBS model is the cost which is determined by an algorithm's ability to execute tasks under specified budget constraint, and this is evaluated by using the task's cost to its budget. Total gain cost is calculated as in Eq. 14 [16].

$$\text{Total Gain Cost} = \sum_{i=1}^{n} GC_i \tag{14}$$

From Fig. 4, the lowest gain cost is for executing tasks with DBS model while the highest cost is for tasks executed with SJF algorithm.

6.3 Number of violations (NoV)

The number of violation is the total number of tasks that violate their deadline. The NoV is defined in Eq. 15 [27] (Fig. 5).

$$\text{Number of Violations} = \sum_{i=1}^{n} TvD \tag{15}$$

where *TvD* is the Task violated Deadline.

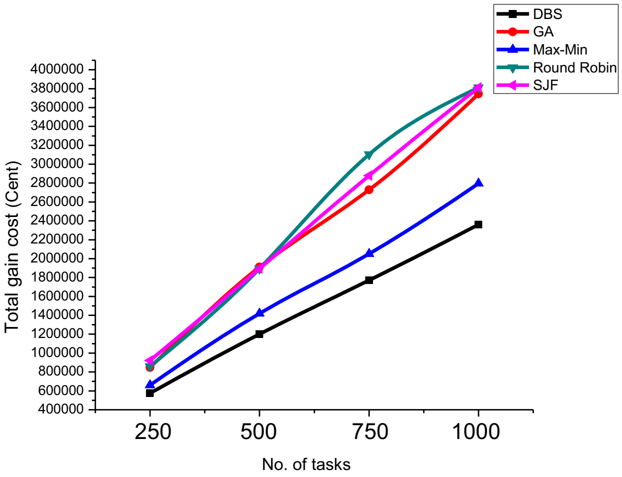


Fig. 4 Total gain cost

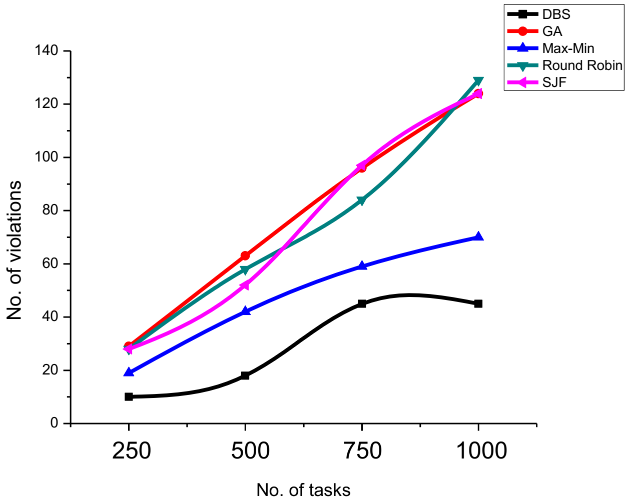


Fig. 5 Number of violation

6.4 Provider profit

For service providers, the pivotal parameter is profit which means total revenues that are summed from the amounts charged to the successful tasks of users [23] which is calculated by subtracting the actual implemented cost of the task from the budget of the successful task. The profit will be calculated based on Eq. 16.

$$\text{Provider Profit} = \sum_{i=1}^{n} (TB_i - GC_i) \tag{16}$$

Figure 6 shows all the differences between the state-of-art scheduling algorithms and the proposed DBS model graphically. It is obvious that the improvement in provider profit achieved when executing tasks on VMs is better under DBS compared to other algorithms.

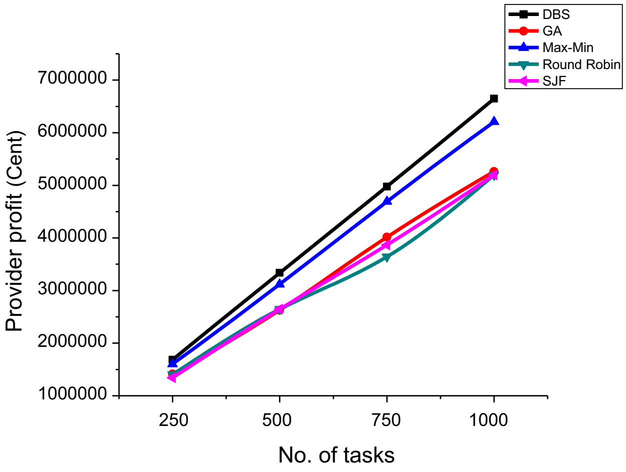


Fig. 6 Provider profit

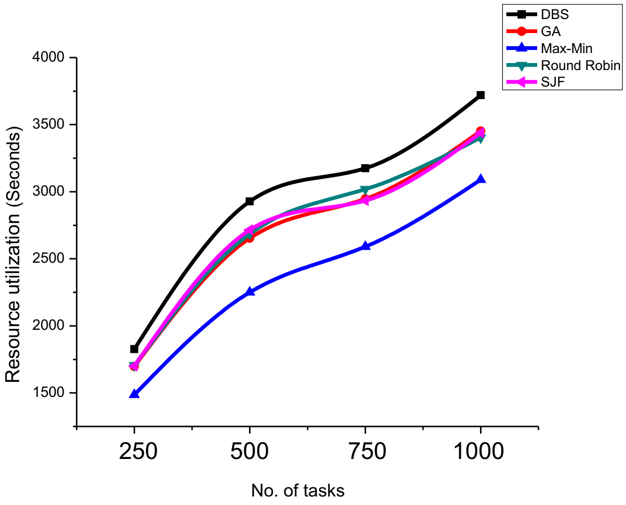


Fig. 7 Comparison of resource utilization

6.5 Average resource utilization

Another critical parameter which concerns the provider is resource utilization. The proposed DBS model is evaluated by efficient utilization of the resources in achieving users' constraints compared with other algorithms where the resource utilization should be maximized. The average resource utilization is defined in Eq. 17 [28].

$$U = \frac{\sum_{j=1}^{n} number\ of\ secceful\ tasks}{\sum_{k=1}^{m} \mathrm{R}_C} \tag{17}$$

Figure 7 Illustrates that the proposed DBS model has the highest resource utilization for all the tasks upon user satisfaction compared to GA, Max–Min, Round Robin, and SJF algorithms.

7 Conclusions

In this paper, the Deadline Budget Scheduling model is proposed as capable of scheduling tasks in the heterogeneous cloud environment with two QoS requirements: time and cost while meeting user satisfaction. The most significant factors of the proposed DBS model are to minimize the makespan under user-defined deadline and reduce monetary costs while not surpassing the user-defined budget. The simulation experiments prove that the proposed DBS model obtained better performance in minimizing the makespan and cost when compared to state-of-art-algorithms in several different configurations, such as low resources or high resources ability, different number of tasks and VMs. For meeting user constraints, the number of violation is minimized in the proposed DBS model while increasing revenue of provider and resources utilization. So the DBS model is considered better compared to the state-of-art algorithms: GA, Max–Min, Round Robin, and SJF.

References

1. Panda, S.K., Gupta, I., Jana, P.K.: Task scheduling algorithms for multi-cloud systems: allocation-aware approach. Springer Science+Business Media, New York (2017)
2. Karthiban, K., Smys, S. Privacy preserving approaches in cloud computing. In: 2018 2nd International Conference on Inventive Systems and Control (ICISC). IEEE (2018)
3. Gill, S.S., et al.: BULLET: particle swarm optimization based scheduling technique for provisioned cloud resources. J. Netw. Syst. Manag. **26**, 1–40 (2018)
4. Kalra, M., Singh, S.: A review of metaheuristic scheduling techniques in cloud computing. Egypt. Inf. J. **16**(3), 275–295 (2015)
5. Tang, X., Li, X., Fu, Z.: Budget-constraint stochastic task scheduling on heterogeneous cloud systems. Concurr. Comput. Pract. Exp. **29**(19), e4210 (2017)
6. Arabnejad, H., Barbosa, J.G.: List scheduling algorithm for heterogeneous systems by an optimistic cost table. IEEE Trans. Parallel Distrib. Syst. **25**(3), 682–694 (2014)
7. Praveena, A., Smys, S. Ensuring data security in cloud based social networks. In: 2017 International Conference of Electronics, Communication and Aerospace Technology (ICECA). Vol. 2. IEEE (2017)
8. Panda, S.K., Jana, P.K.: SLA-based task scheduling algorithms for heterogeneous multi-cloud environment. J. Supercomput. **73**(6), 2730–2762 (2017)
9. Arabnejad, H., Barbosa, J.G.: A budget constrained scheduling algorithm for workflow applications. J. Grid Comput. **12**(4), 665–679 (2014)
10. Mao, Y., Zhong, H., Li, X. Hierarchical model-based associate tasks scheduling with the deadline constraints in the cloud. In: Proceeding of the 2015 IEEE International Conference on Information and Automation, Lijiang, China (2015)
11. Arabnejad, H., Barbosa, J.G. Budget constrained scheduling strategies for on-line workflow applications. In: International Conference on Computational Science and Its Applications. Springer, Cham (2014)

12. Deldari, A., Naghibzadeh, M., Abrishami, S.: CCA: a deadline-constrained workflow scheduling algorithm for multicore resources on the cloud. J. Supercomput. **73**(2), 756–781 (2017)
13. Peng, Z., et al. A reinforcement learning-based mixed job scheduler scheme for cloud computing under SLA constraint. In: IEEE 3rd International Conference on Cyber Security and Cloud Computing (CSCloud), 2016. IEEE (2016)
14. Pop, F., Dobre, C., Cristea, V., Bessis, N., Xhafa, F., Barolli, L.: Deadline scheduling for aperiodic tasks in inter-cloud environments: a new approach to resource management. J. Supercomput **71**, 1754–1765 (2015). https://doi.org/10.1007/s11227-014-1285-82015
15. Shin, S.M., Kim, Y., Lee, S.K. Deadline-guaranteed scheduling algorithm with improved resource utilization for cloud computing. In: 12th Annual IEEE Consumer Communications and Networking Conference (CCNC) (2015)
16. Alworafi, M.A., et al.: Cost-aware task scheduling in cloud computing environment. Int. J. Comput. Netw. Inf. Secur. **9**(5), 52 (2017)
17. Thanasias, V., Lee, C., Hanif, M., Kim, E., Helal, S. VM capacity-aware scheduling within budget constraints in IaaS clouds. PLOS ONE https://doi.org/10.1371/journal.pone.0160456 (2016)
18. Chen, W., Xie, G., Li, R., Bai, Y., Fana, C., Li, K.: Efficient task scheduling for budget constrained parallel applications on heterogeneous cloud computing systems. Future Gener. Comput. Syst. **74**, 1–11 (2017)
19. Rodriguez, M.A., Buyya, R. Budget-driven scheduling of scientific workflows in IaaS clouds with fine-grained billing periods. ACM Trans. Auton. Adapt. Syst. **12**(2). doi: http://dx.doi.org/10.1145/3041036 (2017)
20. Saxena, D., Dr, R.K., Chauhan, D., Kait, R.: Dynamic fair priority optimization task scheduling algorithm in cloud computing: concepts and implementations. I J Comput. Netw. Inf. Secur. **2**, 41–48 (2016)
21. Arabnejad, H., Barbosa, J.G., Prodan, R.: Low-time complexity budget–deadline constrained workflow scheduling on heterogeneous resources. Future Gener. Comput. Syst. **55**, 29–40 (2016)
22. Khorsand, R., Safi-Esfahani, F., Nematbakhsh, N., Mohsenzade, M.: ATSDS: adaptive two-stage deadline-constrained workflow scheduling considering run-time circumstances in cloud computing environments. J. Supercomput. **73**(6), 2430–2455 (2016)
23. Arabnejad, H., Barbosa, J.G.: Multi-QoS constrained and Profit-aware scheduling approach for concurrent workflows on heterogeneous systems. Future Gener. Comput. Syst. **68**, 211–221 (2017)
24. Verma, A., Kaushal, S.: A hybrid multi-objective particle swarm optimization for scientific workflow scheduling. Parallel Comput. **62**, 1–19 (2017)
25. Alworafi, M.A., et al. An improved SJF scheduling algorithm in cloud computing environment. In: International Conference on Electrical, Electronics, Communication, Computer and Optimization Techniques (ICEECCOT), IEEE (2016)
26. Dhari, A., Arif, K.I.: An efficient load balancing scheme for cloud computing. Indian J. Sci. Technol. **10**(11), 1–8 (2017)
27. Alworafi, M.A., Mallappa, S.: An enhanced task scheduling in cloud computing based on deadline-aware model. Int. J. Grid High Perform. Comput. **10**(1), 31–53 (2018)
28. Nawaz, S. Real Time Tasks Scheduling in Cloud Computing Environment. Diss. National Institute of Technology Rourkela (2015)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Mokhtar A. Alworafi Ph.D Researcher Scholar, DoS in Computer Science, University of Mysore, Mysore, India. He has earned BSc in Computer Science from Al-Technologia University in Iraq and MSc in CS from Kuvempu University, India. He works as a teaching assistant in The Center of Computer and Information Technology, Ibb University, Yemen. He has published 7 research papers in reputed international Journals and Conferences. His area of research interest includes Cloud Computing, Parallel Computing, Computer Networks, Internet of Things, and Big Data.

Dr. Suresha is currently working as a Professor, in the DoS in Computer Science, University of Mysore, Mysore. He has 26 years of teaching experience in computer science at postgraduate level in various universities. He has obtained MSc. from University of Mysore, M.Phil. from DAVV, M.Tech. from IIT-Kharagpur, and Ph.D. from IISc-Bangalore. He has published research papers in reputed International and national conferences. His area of research includes Dynamic Web Caching, Database Systems, Image Search Engines, E-governance, Opinion Mining, and Cloud Computing. He has also taught many courses in foreign university as part of teaching assignments.