CHAPTER 5

DEADLINE BUDGET SCHEDULING FOR VIRTUAL CLOUD ENVIRONMENT

Cloud computing has become the most attractive platform compared to grid computing, that offers several services such as infrastructure, platform, and software as services, where the users can use these services on the cloud and pay according to their consumption and on the fulfillment of quality of service constraints such as deadline and budget. In order to schedule the tasks efficiently, cost monetary metrics must be considered while optimizing execution time performance under users’ defined constraints.

In this chapter, the deadline and budget constraints are presented, and the proposed model aims to achieve the goal of scheduling the tasks to the suitable VM based on the deadline and budget constraint.

5.1. Preamble

In recent years, cloud computing has altered the form of all computing paradigms by providing a lot of services on the internet [7].

---

3Several parts of the research in this chapter have appeared in the following publication:

Chapter 5: Deadline Budget Scheduling for virtual cloud environment

The resources of the cloud such as infrastructure, platform, and software are provided as services and a pricing plan is used for these resources by pay-as-per-use [90].

When the required performance is not attained, the users will hesitate to make payment, so, it is necessary to use efficient scheduling in the cloud computing environment to achieve this purpose [54]. Also, the nature heterogeneity, highly dynamic nature, and multidomain characteristics lead to the differences in the cost of resource, in addition to the capacity of computing, communication capacity, and storage, all of which make it more complicated to manage resources [91]. The heterogeneous resources can be local or distributed geographically, which are used for executing intensive computational applications [92]. Cloud user always negotiates with the cloud service provider to sign on service level agreements before starting any service use [84].

In computing model, a user pays only for what he consumes from a service and resources that he needs. A user is concerned with two most important factors: cost and time. The cost and time factors are difficult challenges for task scheduling [40].

Task scheduling is an important issue in cloud computing where it is responsible for assigning the tasks to the suitable resources [93]. The user’s tasks are submitted to different cloud resources based on the computational time and cost using task constraints (deadline and budget). The running costs can be optimized by a provider, as well as a high QoS is guaranteed by ensuring equitable access to resources shared by all users [94].

Therefore, the execution of tasks must take into account scheduling on the heterogeneous environment to achieve the user-defined deadline, and the monetary
costs of the tasks executed should not exceed the user-defined budget. Because of higher leasing costs of resources, the cost of execution and resource utilization leased has gained considerable importance [5].

In general, task scheduling should fulfil the following requirements of scheduling aims:

i. **Quality of Service / Service Level Agreement Constraint**: The service level agreement should clearly state the quality of service requirements for scheduling the tasks by specifying the deadline, task scheduling expenditure budget, service security, and system reliability. Hence the quality of service target constraints of task scheduling should be taken into account to achieve the quality of service requirement.

ii. **Service Revenue**: Cloud computing environment consists of huge numbers of servers, causing higher input costs. Therefore, in task scheduling in cloud computing several economic principles, and practices are used, which seem as more effective in performance and look more reasonable. So, by meeting quality of service constraints, and satisfying resource requests, how to increase service revenue have become the other critical objective of task scheduling for providers of cloud service.

Thus, how can the tasks be scheduled with low payment costs along with less completion time (makespan), and at the same time, maximizing resource utilization in cloud computing environment has become an increasing concern, also in the academic field it is considered as one of the technical difficulties [36].
5.2. Deadline and Budget Constraints

The deadline and budget values must be negotiated between users and cloud service providers for meaningful scheduling solutions so that the agreed values are reasonable and mutually acceptable [45].

According to the quality of service constraints, the time of execution of the task under the deadline constraint as well as the cost which must not exceed the budget constraint that is specified by the user is ensured [46].

Under the constraints of the deadline and budget, there are some metrics that can work and fall within them such as makespan, number of violations, improvement of makespan ratio, improvement of cost ratio, total gain cost, provider profit, and resource utilization as shown in Figure 5.1.
Figure 5.1: Metrics under deadline and budget constraints

5.3. **Task Scheduling Based on Deadline Budget Model**

This contribution highlights deadline and budget constraints in task scheduling to improve the QoS in association with two basic metrics 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). Therefore, we proposed the Deadline Budget Scheduling (DBS) model to execute the users' tasks on VMs under the QoS constraints at less execution time. The methodology of the DBS model assumes that the cloud environment is hosted on 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. The DBS model seeks to minimize 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 performance efficiencies depending on the resources that are provided to them. In order to formulate the DBS model, we define \( D_c \) as data center consisting of several \( H \) hosts, each host contains a set of VMs. We denote resources of VM 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 varies depending on where the tasks are assigned to different cost VMs. 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 and are denoted as \( T = \{ TD, TB, Len, Fit, Fot \} \) respectively. Our target is to minimize the makespan and execution cost considering user satisfaction.
5.3.1. Task Constraint Type

Each task has a constraint type based on user satisfaction. In our proposed DBS model, there are three types of constraint type assigned as in Equation 5.1:

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

(5.1)

5.3.2. Clustering of Resources

The available resources are clustered based on the user’s satisfaction in the following way:

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

As shown in Figure 5.2, the scheduling DBS model consists of cloud users who send their tasks including the constraints (deadline, budget). Each task is checked for detecting its constraint type based on user satisfaction. If the task constraint type is deadline and budget it will be implemented in cluster one; if the constraint type is 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. Other attributes of the task (length, file size) is also included in the selection.
5.3.3. Scheduling Strategy

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 Figure 5.3.

When the constraint type is D & B, the deadline and budget constraints should be met. Our proposed DBS model will obtain the appropriate VM in the first cluster for implementing the task. Next, in level two, length of the task (Len) and the input file size (Fit) are compared as mentioned in Equation 5.2.
Chapter 5: Deadline Budget Scheduling for virtual cloud environment

\[ \text{D&B Constraint Type} = \begin{cases} 
\text{completion time} & \text{if } \text{Len} > \text{Fit} \\
\text{data transfer time} & \text{if } \text{Len} \leq \text{Fit} 
\end{cases} \] (5.2)

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.

![Organogram of the DBS model](image)

Figure 5.3: Organogram of the DBS model

In the first cluster, the VM which achieves less expected completion time will be selected. The Expected Completion Time (ECT) is calculated based on the Equation 5.3.

\[ \text{ECT} = \sum_{i=1}^{p} \text{EX}_i + \text{EX} \text{ of current task} \] (5.3)

Where \( p \) is number of previous tasks in specific VM, while \( \text{EX} \) is the execution time of the task, which is calculated based on the Equation 5.4 [85].

\[ \text{EX} = \frac{\text{Len}}{R_C} \] (5.4)

In cases where the length of the task is lesser than the input file size, the task will be assigned to a VM that has less data transfer time. Data transfer time is calculated as mentioned in the Equation 5.5.
Then, the expected data transfer time of the task in each VM is calculated based on the Equation 5.6.

\[ \text{EDT} = \sum_{i=1}^{P} DT_i + DT \text{ of current task} \]  \hspace{1cm} (5.6)

In our proposed DBS model, the second constraint type is D which focuses on the deadline constraint only, the completion time of task is calculated in the second cluster and the task will be sent to VM that has lesser expected completion time. The expected completion time is mentioned in the Equation 5.3.

The last constraint type of our proposed DBS model is B. The task requirement is obtaining the cost of VMs in the third cluster as mentioned in the Equation 5.7 which compares between the length of the task and the input file size as below:

\[ \text{B Constraint Type} = \begin{cases} \text{CPU Gain} & \text{if } \text{Len} > \text{Fit} \\ \text{Data Transfer Gain} & \text{if } \text{Len} \leq \text{Fit} \end{cases} \]  \hspace{1cm} (5.7)

When the length of the task is the largest, assuming that faster machine is used by selecting VM that has less Expected CPU Gain (ECG) it is calculated based on the Equation 5.8.

\[ \text{ECG} = \frac{\text{Len}_i}{R_C} \times C_{RC} \]  \hspace{1cm} (5.8)

On the other hand, if the file size is the largest, the data transfer cost is kept high. The task will be mapped to faster VM, and the Expected Data Gain (EDG) is calculated as shown in the Equation 5.9.
\[ \text{EDG} = \frac{\text{Fit}_i + \text{Fot}_i}{R_B} \ast C_{RB} \] (5.9)

Finally, the Gain Cost (GC) of the task will be calculated based on the Equation 5.10 [81].

\[ \text{GC} = \left( \frac{\text{Len}_i}{R_C} \ast C_{RC} + \frac{\text{Fit}_i + \text{Fot}_i}{R_B} \ast C_{RB} + \frac{\text{Fit}_i + \text{Fot}_i}{R_M} \ast C_{RM} + \frac{\text{Fit}_i + \text{Fot}_i}{R_S} \ast C_{RS} \right) \] (5.10)

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

We take an example to show the mechanism of our proposed DBS model, for which, we consider Table 5.1 and 5.2 to describe the task attributes and VMs configurations respectively:

Table 5.1: Task attributes

<table>
<thead>
<tr>
<th>Task_ID</th>
<th>Length</th>
<th>Input file size</th>
<th>Output file size</th>
<th>TB</th>
<th>TD</th>
<th>Constraint type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1000</td>
<td>1200</td>
<td>300</td>
<td>3</td>
<td>1</td>
<td>D &amp; B</td>
</tr>
<tr>
<td>2</td>
<td>2000</td>
<td>2750</td>
<td>100</td>
<td>3</td>
<td>-</td>
<td>B</td>
</tr>
<tr>
<td>3</td>
<td>5000</td>
<td>1500</td>
<td>200</td>
<td>-</td>
<td>10</td>
<td>D</td>
</tr>
<tr>
<td>4</td>
<td>4000</td>
<td>2000</td>
<td>500</td>
<td>4</td>
<td>-</td>
<td>B</td>
</tr>
</tbody>
</table>

Table 5.2: VM configurations

<table>
<thead>
<tr>
<th>VM_Id</th>
<th>VM_MIPS</th>
<th>VM_BW</th>
<th>VM_RM</th>
<th>VM_ST</th>
<th>Cost-MIPS</th>
<th>Cost-BW</th>
<th>Cost-RM</th>
<th>Cost-ST</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>500</td>
<td>60</td>
<td>113</td>
<td>250</td>
<td>0.01</td>
<td>0.001</td>
<td>0.2</td>
<td>0.02</td>
</tr>
<tr>
<td>2</td>
<td>2000</td>
<td>80</td>
<td>210</td>
<td>250</td>
<td>0.04</td>
<td>0.001</td>
<td>0.3</td>
<td>0.02</td>
</tr>
<tr>
<td>3</td>
<td>1000</td>
<td>40</td>
<td>170</td>
<td>350</td>
<td>0.02</td>
<td>0.001</td>
<td>0.1</td>
<td>0.04</td>
</tr>
</tbody>
</table>

The constraint type of each task is detected based on the three kinds of constraint type used in our proposed DBS model as mentioned in the Equation 5.1. Based on the detected constraint type, the tasks are distributed among the available VMs; when the constraint type is D & B, the expected deadline and the expected gain cost are calculated to check which of the VMs fulfill the two constraints: deadline and budget. If the constraint type is D, the expected deadline is checked to determine if each VM has met the deadline constraint, otherwise, the expected gain cost is calculated to obtain the VM that has met the budget constraint; and then each VM is classified into cluster one, cluster two, cluster three as described in the Table 5.3.
Table 5.3. Expected deadline and gain cost of each task into each VM

<table>
<thead>
<tr>
<th>Task _ID</th>
<th>Expected deadline</th>
<th>Gain Cost (GC)</th>
<th>Cluster of VM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>VM1</td>
<td>VM2</td>
<td>VM3</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0.5</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>2.5</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
</tbody>
</table>

In tasks 2 and 4 the expected deadline is not calculated because the user has already determined the constraint as budget, besides, in task 3 the gain cost is not calculated because the user determined the constraint as the deadline.

The following is an explanation of how to obtain the values in Table 5.3:

In task 1, the constraint type is deadline and budget. The expected deadline and gain cost for task 1 is calculated as follow:

**Task 1 in VM1:**

**Expected deadline of task 1** = length of task / MIPS of VM1 = 1000 / 500 = 2

**Gain CPU Cost of task 1** = length of task / MIPS of VM1 * cost of CPU = 1000 / 500 * 0.01 = 0.02

**Gain Bandwidth Cost of task 1** = Fit + Fot / Bandwidth * cost of Bandwidth = 1200 + 300 / 60 * 0.001 = 0.025

**Gain Memory Cost of task 1** = Fit + Fot / Memory * cost of Memory = 1200 + 300 / 113 * 0.2 = 2.65

**Gain Storage Cost of task 1** = Fit + Fot / Storage * cost of Storage = 1200 + 300 / 250 * 0.02 = 0.12

**EGC of task 1 in VM1** = 0.02 + 0.025 + 2.65 + 0.12 = **2.815**

**Expected deadline** of task 1 in VM1 > TD of task 1 → 2 > 1.

**Expected GC of task 1 in VM1** < TB of task 1 → 2.815 < 3.

So the Expected deadline = **2** (exceed the constraint defined), and Expected GC=**2.815** → means VM1 is labeled as 0. VM1 not meets the constraints.
Task 1 in VM2:

**Expected deadline of task 1** = length of task / MIPS of VM2 = 1000/ 2000 = 0.5

**Gain CPU Cost of task 1** = length of task / MIPS of VM2 * cost of CPU = 1000 / 2000 * 0.04 = 0.02

**Gain Bandwidth Cost of task 1** = Fit + Fot / Bandwidth * cost of Bandwidth = 1200 + 300 / 80 * 0.001 = 0.018

**Gain Memory Cost of task 1** = Fit + Fot / Memory * cost of Memory = 1200 + 300 / 210 * 0.3 = 2.14

**Gain Storage Cost of task 1** = Fit + Fot / Storage * cost of Storage = 1200 + 300 / 250 * 0.02 = 0.12

**EGC of task 1 in VM2** = 0.02 + 0.018 + 2.14 + 0.12 = **2.298**

**Expected deadline** of task 1 in VM2 < TD of task 1 → 0.5 < 1.

**Expected GC** of task 1 in VM2 < TB of task 1 → 2.298 < 3.

So the Expected deadline = 0.5, and Expected GC = **2.298** → means VM2 is labeled as 1. VM2 meets both the constraints.

Task 1 in VM3:

**Expected deadline of task 1** = length of task / MIPS of VM3 = 1000/ 1000 = 1

**Expected GC of task 1** = length of task / MIPS of VM3 * cost of CPU = 1000 / 1000 * 0.02 = 0.02

**Gain Bandwidth Cost of task 1** = Fit + Fot / Bandwidth * cost of Bandwidth = 1200 + 300 / 40 * 0.001 = 0.037

**Gain Memory Cost of task 1** = Fit + Fot / Memory * cost of Memory = 1200 + 300 / 170 * 0.1 = 0.88

**Gain Storage Cost of task 1** = Fit + Fot / Storage * cost of Storage = 1200 + 300 / 350 * 0.04 = 0.17

**EGC of task 1 in VM3** = 0.02 + 0.037 + 0.88 + 0.17 = **1.107**
**Expected deadline** of task 1 in VM₃ = TD of task 1 $\rightarrow$ 1 = 1.

**Expected GC** of task 1 in VM₃ < TB of task 1 $\rightarrow$ 1.107 < 3.

So the Expected deadline = 1, and Expected GC = 1.107 means VM₃ is labeled as 1. VM₃ meets both the constraints.

In task 2, the constraint type is budget so only need to calculate the gain cost for task 2 as follow:

**Task 2 in VM₁:**

**Gain CPU Cost of task 2** = length of task / MIPS of VM₁ * cost of CPU = 2000 / 500 * 0.01 = 0.04

**Gain Bandwidth Cost of task 2** = Fit + Fot / Bandwidth * cost of Bandwidth = 2750 + 100 / 60 * 0.001 = 0.047

**Gain Memory Cost of task 2** = Fit + Fot / Memory * cost of Memory = 2750 + 100 / 113 * 0.2 = 5.04

**Gain Storage Cost of task 2** = Fit + Fot / Storage * cost of Storage = 2750 + 100 / 250 * 0.02 = 0.228

**EGC of task 2 in VM₁** = 0.04 + 0.047 + 5.04 + 0.228 = 5.355

**Expected GC** of task 2 in VM₁ > TB of task 2 $\rightarrow$ 5.355 > 3 means VM₁ is labeled = 0.

**Task 2 in VM₂:**

**Gain CPU Cost of task 2** = length of task / MIPS of VM₂ * cost of CPU = 2000 / 2000 * 0.04 = 0.04

**Gain Bandwidth Cost of task 2** = Fit + Fot / Bandwidth * cost of Bandwidth = 2750 + 100 / 80 * 0.001 = 0.035

**Gain Memory Cost of task 2** = Fit + Fot / Memory * cost of Memory = 2750 + 100 / 210 * 0.3 = 4.07
**Gain Storage Cost of task 2** = \( \text{Fit} + \text{Fot} / \text{Storage} \times \text{cost of Storage} = 2750 + 100 / 250 \times 0.02 = 0.228 \)

**EGC of task 2 in VM_2** = 0.04 + 0.035 + 4.07 + 0.228 = **4.373**

**Expected GC of task 2 in VM_2** > TB of task 2 \( \rightarrow 4.373 > 3 \) means VM_2 is labeled =0.

**Task 2 in VM_3:**

**Gain CPU Cost of task 2** = \( \text{length of task} / \text{MIPS of VM_3} \times \text{cost of CPU} = 2000 / 1000 \times 0.02 = 0.04 \)

**Gain Bandwidth Cost of task 2** = \( \text{Fit} + \text{Fot} / \text{Bandwidth} \times \text{cost of Bandwidth} = 2750 + 100 / 40 \times 0.001 = 0.071 \)

**Gain Memory Cost of task 2** = \( \text{Fit} + \text{Fot} / \text{Memory} \times \text{cost of Memory} = 2750 + 100 / 170 \times 0.1 = 1.6 \)

**Gain Storage Cost of task 2** = \( \text{Fit} + \text{Fot} / \text{Storage} \times \text{cost of Storage} = 2750 + 100 / 350 \times 0.04 = 0.325 \)

**EGC of task 2 in VM_3** = 0.04 + 0.071 + 1.6 + 0.325 = **2.036**

**Expected GC of task 2 in VM_3** < TB of task 2 \( \rightarrow 2.036 < 3 \) means VM_3 is labeled =1.

In task 3, the constraint type is deadline so need to calculate the expected deadline for task 3 as follow:

**Task 3 in VM_1:**

**Expected deadline of task 3** = \( \text{length of task} / \text{MIPS of VM_1} = 5000 / 500 = 10 \)

**Expected deadline** of task 3 in VM_1 = TD of task 3 \( \rightarrow 10 = 10 \) means VM_1 is labeled =1.
Task 3 in VM2:

**Expected deadline of task 3** = length of task / MIPS of VM1 = 5000 / 2000 = 2.5

**Expected deadline** of task 3 in VM2 < TD of task 3 → 2.5 < 10 means VM2 is labeled =1.

Task 3 in VM3:

**Expected deadline of task 3** = length of task / MIPS of VM1 = 5000 / 1000 = 5

**Expected deadline** of task 3 in VM3 < TD of task 3 → 5 < 10 means VM3 is labeled =1.

In task 4, the constraint type is budget so need to calculate the gain cost for task 4 as follow:

**Task 4 in VM1:**

Gain CPU Cost of task 4 = length of task / MIPS of VM1 * cost of CPU = 4000 / 500 * 0.01 = 0.08

Gain Bandwidth Cost of task 4 = Fit + Fot / Bandwidth * cost of Bandwidth = 2000+500 / 60 * 0.001 = 0.041

Gain Memory Cost of task 4 = Fit + Fot / Memory * cost of Memory = 2000 + 500 / 113 * 0.2 = 4.42

Gain Storage Cost of task 4 = Fit + Fot / Storage * cost of Storage = 2000 + 500 / 250 * 0.02 = 0.2

**EGC of task 4 in VM1 = 0.08 + 0.041 + 4.42 + 0.2 = 4.741**

Expected GC of task 4 in VM1 > TB of task 4 → 4.741 > 4 means VM1 is labeled =0.

**Task 4 in VM2:**

Gain CPU Cost of task 4 = length of task / MIPS of VM2 * cost of CPU = 4000/ 2000 * 0.04 = 0.08
Gain Bandwidth Cost of task 4 = \( \text{Fit} + \text{Fot} / \text{Bandwidth} \times \text{cost of Bandwidth} = 2000 + 500 / 80 \times 0.001 = 0.031 \)

Gain Memory Cost of task 4 = \( \text{Fit} + \text{Fot} / \text{Memory} \times \text{cost of Memory} = 2000 + 500 / 210 \times 0.3 = 3.57 \)

Gain Storage Cost of task 4 = \( \text{Fit} + \text{Fot} / \text{Storage} \times \text{cost of Storage} = 2000 + 500 / 250 \times 0.2 = 0.2 \)

EGC of task 4 in VM\(_2\) = 0.08 + 0.031 + 3.57 + 0.2 = 3.881

Expected GC of task 4 in VM\(_2\) < TB of task 4 \(\rightarrow 3.881 < 4\) means VM\(_2\) is labeled =1.

Task 4 in VM\(_3\):

Gain CPU Cost of task 4 = \( \text{length of task} / \text{MIPS of VM}_3 \times \text{cost of CPU} = 4000 / 1000 \times 0.02 = 0.08 \)

Gain Bandwidth Cost of task 4 = \( \text{Fit} + \text{Fot} / \text{Bandwidth} \times \text{cost of Bandwidth} = 2000 + 500 / 40 \times 0.001 = 0.062 \)

Gain Memory Cost of task 4 = \( \text{Fit} + \text{Fot} / \text{Memory} \times \text{cost of Memory} = 2000 + 500 / 170 \times 0.1 = 1.47 \)

Gain Storage Cost of task 4 = \( \text{Fit} + \text{Fot} / \text{Storage} \times \text{cost of Storage} = 2000 + 500 / 350 \times 0.04 = 0.285 \)

EGC of task 4 in VM\(_3\) = 0.08 + 0.062 + 1.47 + 0.285 = 1.897

Expected GC of task 4 in VM\(_3\) < TB of task 4 \(\rightarrow 1.897 < 4\) means VM\(_3\) is labeled =1.

After that, our proposed DBS model will check and compare the length and file size for each task as shown in the Table 5.4:

<table>
<thead>
<tr>
<th>Task_ ID</th>
<th>Constraint type</th>
<th>Len &gt; Fit</th>
<th>Len &lt; Fit</th>
<th>Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>D &amp; B</td>
<td>√</td>
<td></td>
<td>EDT</td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>√</td>
<td></td>
<td>EDG</td>
</tr>
<tr>
<td>3</td>
<td>D</td>
<td>√</td>
<td></td>
<td>ECT</td>
</tr>
<tr>
<td>4</td>
<td>B</td>
<td>√</td>
<td></td>
<td>ECG</td>
</tr>
</tbody>
</table>
The following is an explanation of how to obtain the table values in Table 5.4:

**Task 1 constraint type is time and cost:**
- Len of task 1 < input file of task 1 \( \rightarrow 1000 < 1200 \) so calculated the EDT.

**Task 2 constraint type is cost:**
- Len of task 2 < input file of task 2 \( \rightarrow 2000 < 2750 \) so calculated the EDG.

**Task 3 constraint type is time:**
- Len of task 3 > input file of task 3 \( \rightarrow 5000 > 1500 \) so calculated the ECT.

**Task 4 constraint type is cost:**
- Len of task 4 > input file of task 4 \( \rightarrow 4000 > 2000 \) so calculated the ECG.

Each task should be scheduled to appropriate VM that achieves efficient performance, then, we calculate the EDT, EDG, ECT, and ECG based on the Equations 5.6, 5.9, 5.3, 5.8 respectively as described in Table 5.5:

<table>
<thead>
<tr>
<th>Task_ID</th>
<th>Procedure</th>
<th>VM1</th>
<th>VM2</th>
<th>VM3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>EDT</td>
<td>Not meet TD &amp; TB</td>
<td>18.75</td>
<td>37.5</td>
</tr>
<tr>
<td>2</td>
<td>EDG</td>
<td>Not meet TB</td>
<td>Not meet TB</td>
<td>0.07</td>
</tr>
<tr>
<td>3</td>
<td>ECT</td>
<td>10</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>ECG</td>
<td>Not meet TB</td>
<td>0.08</td>
<td>0.08</td>
</tr>
</tbody>
</table>

Based on the results obtained from Table 5.4, we calculated all the procedures for all the tasks on the VMs. Following are the computations of the obtained results of Table 5.5:
Task 1:
The task 1 in first virtual machine did not meet the task deadline and budget constraints, so we do not calculate the expected deadline and expected gain cost in VM\textsubscript{1} based on the results from Table 5.3.

- Task 1 EDT VM\textsubscript{2} = DT of pervious task + DT of task = 0 + 1500 / 80 = \textbf{18.75}
- Task 1 EDT VM\textsubscript{3} = DT of pervious task + DT of task = 0 + 1500 / 40 = 37.5

Task 2:
The task 2 in first and second virtual machines did not meet the task budget, so we do not calculate the expected gain cost in VM\textsubscript{1} and VM\textsubscript{2} based on the results from Table 5.3.

- Task 2 EDG VM\textsubscript{3} = Fit + Fot / Bandwidth * cost \text{ Bandwidth=2850/40} \text{ *0.001=0.07}

Task 3:

- Task 3 ECT VM\textsubscript{1} = Ex of pervious task + EX of current task = 5000 / 500 = 10
- Task 3 ECT VM\textsubscript{2} = Ex of pervious task (task 1) + EX of current task = (1000/2000) + (5000/2000) = 0.5 + 2.5 = \textbf{3}
- Task 3 ECT VM\textsubscript{3} = Ex of pervious task (task 2) + EX of current task = (2000/1000) + (5000 / 1000) = 2+5 = \textbf{7}

Task 4:
The task 4 in first virtual machine did not meet the task budget, so we do not calculate the expected gain cost in VM\textsubscript{1} based on the results from Table 5.3.
Task 4 ECG VM$_2$ = length task / CPU * cost CPU = 4000 / 2000 * 0.04 = 0.08

Task 4 ECG VM$_3$ = length task / CPU * cost CPU = 4000 / 1000 * 0.02 = 0.08

Based on Table 5.5:

- Task 1 will be mapped to VM$_2$ that returns less EDT.
- Task 2 will be mapped to VM$_3$ that meets tasks constraints.
- Task 3 will be mapped to VM$_2$ that returns less ECT.
- Task 4 will be mapped to VM$_2$ that returns less ECG as first.
5.3.5. Steps of Deadline Budget Scheduling Model

The steps of DBS model are as follows:

<table>
<thead>
<tr>
<th>Input</th>
<th>List of unmapped tasks which have user-defined constraints (deadline, budget)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output</td>
<td>Minimizing the makespan and expenditure cost for these tasks</td>
</tr>
</tbody>
</table>

1. 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

   else

   Calculate EDT

   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

   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

   else

   Calculate EDG

   End for

   c. **Assign** the task to fastest VM.

End for

2. **Computation performance**

Calculate the metrics:

i. Average makespan based on the Equation 5.13.


iii. Number of violations based on the Equation 5.15.

iv. Profit of provider based on the Equation 5.16.


vi. Improvement of makespan ratio based on the Equation 5.18.

vii. Improvement of cost ratio based on the Equation 5.19.
5.4. Performance Metrics

In this section, we present the performance metrics used to evaluate the proposed DBS model. These metrics are used in task scheduling to measure the efficiency of the algorithms as follows:

5.4.1. 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 the Equation 5.11.

\[ \text{Makespan} = \sum_{j=1}^{n} (ECT_i * A[i, j]) \]  \hspace{1cm} (5.11)

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 the Equation 5.12 [7]:

\[ A[i, j] = \begin{cases} 1 & \text{if } T \text{ is assigned to VM}_j \\ 0 & \text{Otherwise} \end{cases} \]  \hspace{1cm} (5.12)

Then for all VMs, the average makespan is calculated as mentioned in the Equation 5.13 [32]

\[ \text{Avg. Makespan} = \frac{\sum_{j=1}^{m} \text{Makespan}_j}{m} \]  \hspace{1cm} (5.13)

5.4.2. Total Gain Cost: Another performance metric which is used to evaluate our 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 the Equation 5.14 [81].
Total Gain Cost = $\sum_{i=1}^{n} GC_i$  \hspace{1cm} (5.14)

5.4.3. **Number of Violations (NoV):** The number of violation is the total number of tasks that violate their deadline. The NoV is defined in the Equation 5.15 [30].

\[
\text{Number of Violations} = \sum_{i=1}^{n} TVD
\]  \hspace{1cm} (5.15)

5.4.4. **Provider Profit:** For service providers, the pivotal metric is profit which means total revenues that are summed up from the amounts charged to the successful tasks of users [47] 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 the Equation 5.16.

\[
\text{Provider Profit} = \sum_{i=1}^{n} (TB_i - GC_i)
\]  \hspace{1cm} (5.16)

5.4.5. **Average Resource Utilization:** Another critical metric which concerns the service provider is resource utilization. Our proposed DBS model was evaluated by efficient utilization of the resources by achieving the users’ constraints compared to other algorithms where the resource utilization should be maximized. The average resource utilization is defined in the Equation 5.17 [63].

\[
RU = \frac{\sum_{j=1}^{n} \text{number of successful tasks}}{\sum_{k=1}^{m} RC}
\]  \hspace{1cm} (5.17)
5.4.6. **Improvement of Makespan Ratio**: The vital metric that is taken into account for the performance of our model is makespan, so, we need to measure the ratio of improvement of this metric. The improvement of makespan ratio is calculated by Equation the 5.18 [57].

\[
\text{Makespan ratio} = \left( 1 - \frac{\sum_{i=1}^{e} \text{Avg Makespan}_{\text{DBS}}}{\sum_{i=1}^{e} \text{Avg Makespan of other algorithm}} \right) \times 100 \quad (5.18)
\]

where \( e \) is the number of experiments

5.4.7. **Improvement of Cost Ratio**: The cost ratio is an important metric to measure the performance of our DBS model. Therefore, we calculated the improvement of cost ratio by the Equation 5.19 [57].

\[
\text{Cost ratio} = \left( 1 - \frac{\sum_{i=1}^{e} \text{Cost}_{\text{DBS}}}{\sum_{i=1}^{e} \text{Cost of other algorithm}} \right) \times 100 \quad (5.19)
\]
5.5. Results of Experiments and Analysis

5.5.1. Implementation Environment

We used the CloudSim toolkit simulator for implementing the experiments of the DBS model. The CloudSim is characterized by modeling behavior for components of cloud system like data center, processing elements, virtual machine, etc. [95]. Researchers and industry-based developers can look at specific and important issues in system design without worrying about the details of infrastructures and cloud-based services [96].

The CloudSim toolkit is a commonly used simulator due to its flexibility and simplicity, and is executed using java language [97].

The advantages of the CloudSim are that it allows the establishment of policies to bind tasks with VMs, allocating VMs for hosting them in data centers, scheduling of VMs, and for energy consumption [98].
5.5.2. Experiments Configuration

The Table 5.6 clarifies the configuration for the experiments:

<table>
<thead>
<tr>
<th>Configurations</th>
<th>Data center</th>
<th>Server</th>
<th>Virtual machine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Numbers</td>
<td>2</td>
<td>5, 7, 9, and 11</td>
<td></td>
</tr>
<tr>
<td>Core</td>
<td>Quad-Core and Dual-Core</td>
<td>1 processing element</td>
<td></td>
</tr>
<tr>
<td>Memory (RAM)</td>
<td>16 GB</td>
<td>0.5 GB</td>
<td></td>
</tr>
<tr>
<td>Storage</td>
<td>1 TB</td>
<td>10 GB image size</td>
<td></td>
</tr>
<tr>
<td>Bandwidth</td>
<td>100 GB/s</td>
<td>1 GB/s</td>
<td></td>
</tr>
<tr>
<td>VM scheduling algorithm</td>
<td>Time-Shared</td>
<td>Time-Shared</td>
<td></td>
</tr>
<tr>
<td>Architecture</td>
<td>X86 Architecture</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Operating system</td>
<td>Linux</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Virtual machine monitor</td>
<td>Xen</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Speed</td>
<td>10,000 MIPS</td>
<td>500, 1000, 2000 and 3000 MIPS</td>
<td></td>
</tr>
</tbody>
</table>

5.5.3. Dataset

The CloudSim allows for modeling the task through a programming structure which is called a cloudlet [99]. So, in this part of research work, the dataset is a cloudlet, the cloudlet represents a task in the CloudSim simulator. The cloudlet contains the number of instructions that need to be implemented [100].

The characteristics of a cloudlet are:

1. Length.
2. Million Instructions (MI).
3. Input file size.
4. Output file size [89].
And there is provision to extend and add other attributes as deadline and budget constraints. For analyzing the results of our DBS model, tasks were generated and distributed as 250, 500, 750, and 1000 tasks.

We have calculated the measured metrics (average makespan, total gain cost, number of violations, provider profit, and average resource utilization) as shown in the Table 5.7:

Table 5.7: Experiments conducted for tasks with different VMs of the DBS model

<table>
<thead>
<tr>
<th>Expts.</th>
<th>No. of VMs</th>
<th>No. of tasks</th>
<th>Average makespan</th>
<th>Total gain cost</th>
<th>Number of violations</th>
<th>Provider profit</th>
<th>Average resource utilization</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
<td>250</td>
<td>1883</td>
<td>576158</td>
<td>10</td>
<td>1686740</td>
<td>1827</td>
</tr>
<tr>
<td>2</td>
<td>7</td>
<td>500</td>
<td>2879</td>
<td>1200682</td>
<td>18</td>
<td>3337891</td>
<td>2927</td>
</tr>
<tr>
<td>3</td>
<td>9</td>
<td>750</td>
<td>3246</td>
<td>1770457</td>
<td>45</td>
<td>4975104</td>
<td>3174</td>
</tr>
<tr>
<td>4</td>
<td>11</td>
<td>1000</td>
<td>3659</td>
<td>2359776</td>
<td>45</td>
<td>6646103</td>
<td>3719</td>
</tr>
</tbody>
</table>
5.6. Performance Evaluation

To evaluate the performance and efficiency of the proposed DBS model we conducted four experiments, where we used varying numbers of VMs to host varying numbers of tasks. The experiments were conducted as follows:

i. 250 tasks with five VMs.

ii. 500 tasks with seven VMs.

iii. 750 tasks with nine VMs.

iv. 1000 tasks with eleven VMs.

The efficiency level of our proposed DBS model can be shown by comparing it with the state of the art scheduling algorithms such as Genetic (GA), Max-Min, Round Robin, and SJF algorithms. Each one of these algorithms has a different strategy to assign the tasks on the available resources.

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

Table 5.8 summarizes the experiments done on different number of tasks hosted on various numbers of VMs measuring the average makespan for our proposed DBS model in comparison to the other scheduling algorithms. As we see from the results,
the average makespan for the DBS model for each task executed on VM has less value than the average makespan of the tasks' executed in the other scheduling algorithms.

Table 5.8: Average of makespan of DBS, GA, Max-Min, Round Robin and SJF algorithms

<table>
<thead>
<tr>
<th>Experiments</th>
<th>Average makespan</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DBS</td>
</tr>
<tr>
<td>No. of tasks</td>
<td>No. of VMs</td>
</tr>
<tr>
<td>250</td>
<td>5</td>
</tr>
<tr>
<td>500</td>
<td>7</td>
</tr>
<tr>
<td>750</td>
<td>9</td>
</tr>
<tr>
<td>1000</td>
<td>11</td>
</tr>
</tbody>
</table>

Through the set of experiments, it has been proved that our 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 to other algorithms: GA, Max-Min, Round Robin, and SJF, as it is clear from the Figure 5.4. By conducting the tasks (250, 500, 750, 1000) on VMs (5, 7, 9, 11), we see that the largest average makespan occurs in the SJF algorithm, next in the Round Robin algorithm, and then in the GA, followed by the Max-Min. Finally, the least makespan is represented in the DBS model, which illustrates that the tasks’ execution time upon the deadline and budget constraints is as less as possible, which in turn maximizes the DBS model performance level.
Chapter 5: Deadline Budget Scheduling for virtual cloud environment

Figure 5.4: Comparison of average makespan

The Table 5.9 shows us how the proposed DBS model attains its objective to enhance the total gain costs by using the resources to execute its tasks where the costs are less than the state of the art scheduling algorithms.

Table 5.9: Total gain cost of DBS, GA, Max-Min, Round Robin and SJF algorithms

<table>
<thead>
<tr>
<th>Experiments</th>
<th>Total gain cost</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DBS</td>
</tr>
<tr>
<td>No. of tasks</td>
<td>No. of VMs</td>
</tr>
<tr>
<td>250</td>
<td>5</td>
</tr>
<tr>
<td>500</td>
<td>7</td>
</tr>
<tr>
<td>750</td>
<td>9</td>
</tr>
<tr>
<td>1000</td>
<td>11</td>
</tr>
</tbody>
</table>
From Figure 5.5, we can notice that the lowest gain cost was for executing tasks with the DBS model while the highest cost was for tasks executed with the SJF algorithm.

![Figure 5.5: Total gain cost](image)

The number of violations is another metric to evaluate the efficiency of the algorithm. When the algorithm achieves less number of violations, it indicates the efficiency of the algorithm. This is achieved in our proposed DBS model, which attains lesser number of violated tasks compared to the other existing algorithms illustrated in the Table 5.10.
Table 5.10: Number of violations of DBS, GA, Max-Min, Round Robin and SJF algorithms

<table>
<thead>
<tr>
<th>Experiments</th>
<th>Number of violations</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of tasks</td>
<td>No. of VMs</td>
</tr>
<tr>
<td>250</td>
<td>5</td>
</tr>
<tr>
<td>500</td>
<td>7</td>
</tr>
<tr>
<td>750</td>
<td>9</td>
</tr>
<tr>
<td>1000</td>
<td>11</td>
</tr>
</tbody>
</table>

Our proposed DBS model with number of violations is as illustrated in the Figure 5.6. The number of tasks violations in the DBS model, based on deadline and budget constraints is lesser when compared to GA, Max-Min, Round Robin, and SJF algorithms.

Figure 5.6: Number of violations

The Table 5.11 summarizes the experiments done on varying number of tasks hosted on varying number of VMs when measuring the provider profit for our proposed DBS model compared to other scheduling algorithms. As we note from the results, the
provider profit in the DBS model is always higher than the provider profit for the same tasks executed in other scheduling algorithms.

Table 5.11: Provider profit of DBS, GA, Max-Min, Round Robin and SJF algorithms

<table>
<thead>
<tr>
<th>Experiments</th>
<th>No. of tasks</th>
<th>No. of VMs</th>
<th>DBS</th>
<th>GA</th>
<th>Max-Min</th>
<th>Round Robin</th>
<th>SJF</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>250</td>
<td>5</td>
<td><strong>1686740</strong></td>
<td>1414874</td>
<td>1602697</td>
<td>1403249</td>
<td>1341766</td>
</tr>
<tr>
<td></td>
<td>500</td>
<td>7</td>
<td><strong>3337891</strong></td>
<td>2628381</td>
<td>3119848</td>
<td>2644751</td>
<td>2644497</td>
</tr>
<tr>
<td></td>
<td>750</td>
<td>9</td>
<td><strong>4975104</strong></td>
<td>4017073</td>
<td>4694679</td>
<td>3641805</td>
<td>3866607</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>11</td>
<td><strong>6646103</strong></td>
<td>5261244</td>
<td>6208620</td>
<td>5193870</td>
<td>5194580</td>
</tr>
</tbody>
</table>

Figure 5.7 shows all the differences between the state of the art scheduling algorithms and the proposed DBS model graphically, in which it is obvious that the improvement in provider profit is achieved while executing tasks on VMs under the DBS model compared to other algorithms.

![Figure 5.7: Provider profit](image-url)
Chapter 5: Deadline Budget Scheduling for virtual cloud environment

To enhance the performance of cloud computing system, the task scheduling strategy should maximize the resource utilization. The results of the proposed DBS model in Table 5.12 proves that our model is capable of maximizing the utilization of resources based on deadline and budget constraints when compared to other algorithms.

Table 5.12: Resource utilization of DBS, GA, Max-Min, Round Robin and SJF algorithms

<table>
<thead>
<tr>
<th>Experiments</th>
<th>Resource utilization</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of tasks</td>
<td>No. of VMs</td>
</tr>
<tr>
<td>250</td>
<td>5</td>
</tr>
<tr>
<td>500</td>
<td>7</td>
</tr>
<tr>
<td>750</td>
<td>9</td>
</tr>
<tr>
<td>1000</td>
<td>11</td>
</tr>
</tbody>
</table>

Figure 5.8 Illustrates that the proposed DBS model has the highest resource utilization for all the tasks upon user satisfaction compared with GA, Max-Min, Round Robin, and SJF algorithms.

Figure 5.8: Comparison of resource utilization
Chapter 5: Deadline Budget Scheduling for virtual cloud environment

The improvement in makespan ratio is observed in the Table 5.13 for our proposed DBS model over GA, Max-Min, Round Robin, and SJF algorithms by 39%, 5%, 41%, and 41% respectively.

Table 5.13: Improvement in makespan ratio for DBS compared with GA, Max-Min, Round Robin, and SJF algorithms

<table>
<thead>
<tr>
<th>Experiments</th>
<th>DBS</th>
<th>GA</th>
<th>Max-Min</th>
<th>Round Robin</th>
<th>SJF</th>
</tr>
</thead>
<tbody>
<tr>
<td>1883</td>
<td>2922</td>
<td>1926</td>
<td>3114</td>
<td>3052</td>
<td></td>
</tr>
<tr>
<td>2879</td>
<td>4909</td>
<td>3020</td>
<td>5015</td>
<td>4933</td>
<td></td>
</tr>
<tr>
<td>3246</td>
<td>5362</td>
<td>3402</td>
<td>5374</td>
<td>5383</td>
<td></td>
</tr>
<tr>
<td>3659</td>
<td>6166</td>
<td>3922</td>
<td>6325</td>
<td>6284</td>
<td></td>
</tr>
<tr>
<td>Sum of Expts.</td>
<td>11667</td>
<td>19359</td>
<td>12270</td>
<td>19828</td>
<td>19652</td>
</tr>
<tr>
<td>Improvement DBS vs algorithm</td>
<td>39%</td>
<td>5%</td>
<td>41%</td>
<td>41%</td>
<td></td>
</tr>
<tr>
<td>Sum of all makespan ratio</td>
<td>126</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average</td>
<td>31.5 %</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The improvement of cost ratio is observed in the Table 5.14 for our proposed DBS model over GA, Max-Min, Round Robin, and SJF algorithms by 36%, 14%, 38%, and 38% respectively.

Table 5.14: Improvement of cost ratio for DBS compared with GA, Max-Min, Round Robin, and SJF algorithms

<table>
<thead>
<tr>
<th>Experiments</th>
<th>DBS</th>
<th>GA</th>
<th>Max-Min</th>
<th>Round Robin</th>
<th>SJF</th>
</tr>
</thead>
<tbody>
<tr>
<td>576158</td>
<td>848033</td>
<td>660214</td>
<td>859654</td>
<td>921143</td>
<td></td>
</tr>
<tr>
<td>1200682</td>
<td>1910206</td>
<td>1418749</td>
<td>1893837</td>
<td>1894094</td>
<td></td>
</tr>
<tr>
<td>1770457</td>
<td>2728512</td>
<td>2050902</td>
<td>3103795</td>
<td>2878976</td>
<td></td>
</tr>
<tr>
<td>2359776</td>
<td>3744648</td>
<td>2797295</td>
<td>3811976</td>
<td>3811306</td>
<td></td>
</tr>
<tr>
<td>Sum of Expts.</td>
<td>5907073</td>
<td>9231399</td>
<td>6927160</td>
<td>9669262</td>
<td>9505519</td>
</tr>
<tr>
<td>Improvement DBS vs other algorithm</td>
<td>36%</td>
<td>14%</td>
<td>38%</td>
<td>38%</td>
<td></td>
</tr>
<tr>
<td>Sum of all cost ratio</td>
<td>126</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average</td>
<td>31.5 %</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
We used T-test to make statistical analysis of our proposed DBS model. T-test is an efficient function that calculates the mean of DBS model which demonstrates the application of normality. Table 5.15 tabulates the T-test results of DBS model compared to other algorithms such as GA, Max-Min, Round Robin, and SJF.

Table 5.15: T-test of DBS model compared to GA, Max-Min, Round Robin, and SJF algorithms

<table>
<thead>
<tr>
<th>Metrics</th>
<th>DBS model vs other algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>GA</td>
</tr>
<tr>
<td>Average makespan</td>
<td>0.02</td>
</tr>
<tr>
<td>Total gain cost</td>
<td>0.1</td>
</tr>
<tr>
<td>Number of violations</td>
<td>0.03</td>
</tr>
<tr>
<td>Provider profit</td>
<td>0.2</td>
</tr>
<tr>
<td>Resource utilization</td>
<td>0.3</td>
</tr>
</tbody>
</table>

5.7. Chapter Contribution

We have proposed a novel model, where the user defines two constraints - deadline and budget as follows:

i. Users’ tasks will be assigned to appropriate VM based on user-defined constraints.

ii. Achieving user satisfaction through minimizing the completion time and cost of this process by implementing the tasks onto VMs and meeting the QoS constraints.

iii. Provider satisfaction is achieved through maximizing profit revenue and resource utilization.
5.8. Chapter Summary

In summary, we can say this contribution will answer the third question: How can we design task scheduling model to execute tasks while achieving user-defined constraints (deadline and budget). We proposed a Deadline Budget Scheduling (DBS) model capable of scheduling the tasks in the heterogeneous cloud environment with two conflicting QoS requirements: time and cost while meeting user satisfaction. The most significant metrics 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 obtains better performance in minimizing the makespan and cost when compared to the state of the art algorithms in several different configurations, such as low resources or high resources ability, different number of tasks and VMs. For meeting the user constraints, the number of violations is minimized in DBS model while increasing the profit of the provider and resources utilization. So our model is considered better compared to the state of the art algorithms: GA, Max-Min, Round Robin, and SJF. The improvement of makespan ratio for our proposed DBS model over GA, Max-Min, Round Robin, and SJF algorithms is by 39%, 5%, 41%, and 41% respectively, in addition to the improvement of cost ratio for our proposed DBS model over GA, Max-Min, Round Robin, and SJF algorithms by 36%, 14%, 38%, and 38% respectively.

In addition, the T-test was calculated to make a statistical analysis of our proposed DBS model to other algorithms such as GA, Max-Min, Round Robin, and SJF.