CHAPTER 5

EFFECTIVE JOB SCHEDULING AND LOAD BALANCING IN GRID

5.1 INTRODUCTION

Grid is a collection of distributed resources such as computing, storage, and network resources from various locations and act as a unified processing resource. Grid Networking is utilized in order to achieve cost savings, reduced time consumption, improved collaboration and sharing of resources among jobs. Grid resources are effectively used for processing user jobs with the help of scheduling techniques. Scheduling is the process that helps for choosing which job is to be considered next. Previous chapter provides mobility aware job grouping and scheduling for reducing energy consumption in grids. In addition, dynamic resources are effectively managed using virtualization technology to improve the scheduling performance in grid.

Traditional load balancing method helps in providing better performance on scheduling in computational grids. However, scalability was not improved for the grid users to service their jobs. On the other hand, grid scheduling algorithms were utilized after performing job and machines grouping based on their priorities and configuration respectively. Though, issues related with speed improvement on job processing and job scheduling efficiency remained unaddressed.
In dynamic environment adjusts the selected control parameters like size of population and genetic operation rates by using Self Acclimatized Genetic Algorithm (SAGA) over the general Genetic Algorithm developed by Arindam Sarkar (2014). Acclimatized Genetic Algorithm based Virtual Machines (VM) Resource Migration (AGAVRM) method is proposed for scheduling the jobs obtained from users with improved job scheduling efficiency and scalability. In the proposed AGAVRM method, effective job scheduling is achieved by predicting the resource availability and future load. First, Grid Booster Algorithm is implemented in order to balance the load of grid with reduced idle time of dynamic resources. Then, resource weight value which helps for denoting each resource’s capacity is utilized to make decision on load balancing and allocate Virtual Machines (VM). Finally, Acclimatized Genetic Algorithm is employed in proposed AGAVRM method to achieve optimal solution on VM resource migration. Therefore, higher job scheduling efficiency and scalability are achieved in Grid Networking environment.

5.2 LOAD BALANCING ON HETEROGENEOUS RESOURCES IN GRID COMPUTING

Deepak Kumar Patel and Chitaranjan Tripathy (2014) achieved dynamic-distributed load balancing on computational grids with the help of Improved load balancing on Enhanced GridSim with Deadline Control (IEGDC) technique. A new scheduling mechanism is employed to improve resources utilization and to prevent from resource overloading. Selection method helps in scheduling based on resource bandwidth state and capacity of different resources.

IEGDC involves five major steps that are discussed below. In first step, current load is measured for all resources. If a resource load is current load, then a Gridlet is assigned to that resource with a deadline.
Followed by, new load of that resource is computed. In second step, state of the resource is ensured periodically. There are three states of resources namely under loaded, normally loaded and overloaded which are based on resource and machine thresholds. After ensuring the state of all resources, grid broker inserts it into the respective list. In third step, list of Gridlets is created to transfer them from an overloaded resource to an under loaded resource. Uncompleted Gridlets from overloaded resource and newly arriving Gridlets are inserted into the uncompleted Gridlet list for further scheduling. Figure 5.1 shows the steps involved in IEGDC as shown below.

![Diagram showing steps involved in IEGDC]

**Figure 5.1  Steps Involved in IEGDC**

As shown in Figure 5.1, is utilized in the fourth step a selection method named as Fastest Bandwidth to Highest Capacity (FBHC) where under loaded resources present in the under loaded resource list are arranged. FBHC
selection method performs two arrangements as follows. First arrangement is performed such that, the under loaded resources are arranged based on the state of resource bandwidth i.e., starting from state of under loaded to overloaded. Second arrangement is performed in such a way that, the under loaded resources that have same state of resource bandwidth present in under loaded resource list are rearranged based on the highest capacity of the resource. Bandwidth speed of a resource is fully based on the state of resource bandwidth. Every resource is allocated with a bandwidth. A resource does not exceed the bandwidth’s usage limit to achieve better performance.

In the fifth step, Gridlets are assigned from the uncompleted Gridlet list to the under loaded resources. A novel mechanism is developed for scheduling and improving resource utilization with the prevention of resource overloading. If resource’s load exceeds the upper bound of normally loaded range, then the resource becomes overloaded. The new scheduling mechanism improves number of finished Gridlets by allocating a resource for Gridlets up to the upper bound of normally loaded range. Finally, uncompleted Gridlets are assigned to the under loaded resources have chosen by FBHC selection method and then Gridlets are scheduled according to the same scheduling algorithm.

5.3 PRIORITY BASED PARALLEL MULTI-SCHEDULER FOR GRIDS

GoodheadTomvie Abraham et al. (2015) developed Grid scheduling algorithms with the grouping of jobs and Grid machines. Job grouping is performed with job priorities and Grid machines are grouped according to their configuration. Jobs and Grid machines groupings are achieved prior to the implementation of respective scheduling algorithm. Grouping of job is performed by considering four groups with the help of
Priority method where machines are also grouped into four groups before applying MinMin Grid scheduling algorithm. Scheduling algorithms are applied concurrently on the paired groups in order to enhance the parallelism and throughput.

Grid machine acquires the attributes such as Worker Id to identify the machine, Number of CPUs to measure the potential of a machine and CPU Speed in terms of GHz. Two methods are employed to perform machines grouping that include Similar Together method and Evenly Distributed method. Similar Together method groups the machines together based on their similar configuration with number of CPUs and speed of the CPU. Evenly Distributed method helps in ensuring that the machines of all configurations are equally divided and distributed into the groups according to their configuration. Hence, every group has formed with same or almost same number of machine configurations. Finally, parallelism is achieved by utilizing Priority-based Parallel Multi-scheduling method with Grid scheduling algorithms.

5.4 PROPOSED ACCLIMATIZED GENETIC ALGORITHM BASED VM RESOURCE MIGRATION (AGAVRM) METHOD

The issues associated with improving scalability and job scheduling efficiency are addressed with the help of proposed Acclimatized Genetic Algorithm based VM Resource Migration (AGAVRM) method by scheduling the jobs in effective manner as follows. Proposed AGAVRM method performs effective job scheduling based on the resource availability where future load is also predicted. Grid Booster Algorithm is utilized in order to balance the load (jobs) of dynamic resources in grid. Here, load balancing decision is made and Virtual Machine allocation is achieved by considering
the resource weights. Resource weight value is obtained based on the capacity of the respective resource. Then, Acclimatized Genetic Algorithm is implemented in proposed the AGAVRM method to perform Virtual Machine resource migration in Grid Networking environment. Therefore, job scheduling efficiency gets improved in a significant manner.

Physical resource utilization and respective loads of individual VMs are considered periodically in order to achieve effective Virtual Machine resource management in proposed AGAVRM method. Physical resources include CPU cycles and memory that are utilized by VMs. VM resource utilization and management are carried out with the help of resource mapping across virtual machines to improve the job scheduling efficiency. Figure 5.2 illustrates the architecture of the proposed AGAVRM method.

![Architecture of the Proposed AGAVRM Method](image)

**Figure 5.2 Architecture of the Proposed AGAVRM Method**

Let a set of all physical machines in the grid system be assumed as \( P = \{P_1, P_2, P_3 \ldots P_N\} \), where \( N \) represents the total number of physical
machines. Individual physical machine is referred as Pi, where i provides the physical machine number which is ranged as (1 <= i <= N). Similarly, a set of VMs on every physical machine Pi is represented as Vi = {Vi1, Vi2, Vim}, where, m represents number of VMs on the physical machine i. The proposed AGAVRM method achieves the implementation of VM, V on the present system by utilizing a solution set denoted by S = {S1, S2, S3 ....SN}. This solution set provides the mapping solution after VM, V is allocated to each of the physical machines. Here, Si represents the mapping structure when V is arranged with a physical machine Pi.

5.4.1 Job Allocation with Grid Booster Algorithm

Load balancing in a grid is important for the running machines in case of any load imbalance which are caused by resource variation. Heterogeneous environment requires various processing capabilities and dynamic load balancing methods to provide better performance on job processing for the grid users. Proposed AGAVRM method provides load balancing decision according to the node weights. These weights are varied for each machine because of the distributed load on the nodes at run time. With the aim of achieving effective scheduling and load balancing among heterogeneous execution resources in grid, problems of VM resource migration is taken into account. VM resources are allocated to every physical node based on the load in a grid. Figure 5.3 shows the Grid Controller and Job Allocation in proposed AGAVRM method.
As shown in Figure 5.3, first a user submits the job requests to the Grid Controller where the jobs are distributed along with VMs. After that, these Jobs are executed at various Physical Machines (PMs). During VM allocation, proposed AGAVRM method makes use of the outcome of future predictor. This future predictor is utilized for predicting the required future resource of VMs by considering the prior statistics. With the help of prediction result, the given jobs are distributed among Grid resources. Followed by the job allocation, jobs are executed independently in the respective PMs. The results of successfully completed jobs are forwarded to Grid controller.

Grid Booster algorithm first computes the weight of a node in a parallel processing system. After that, job proportion for each parallel machine is obtained by utilizing the weight metrics as follows.
\[ W_{CPU} = \text{Available CPU} \times \text{Weight Constant for CPU} \quad (5.1) \]
\[ W_{Mem} = \text{Available Memory} \times \text{Weight Constant for Memory} \quad (5.2) \]
\[ W_{load} = \text{Current load} \times \text{Weight Constant for Load} \quad (5.3) \]

\[ \text{Weight of Machine} = W_{CPU} + W_{Mem} - W_{load} \quad (5.4) \]

From Equation (5.4), ‘\( W_{CPU} \)’ represents the CPU Weight, ‘\( W_{Mem} \)’ refers to Memory Weight, and ‘\( W_{load} \)’ stands for the Weight for current load. Based on these Weight values, Weight of the Machine is computed. Then, For a Node X, required job Percentage is measured as shown below.

\[ W_X = \frac{w_X}{\sum w_i} \quad (5.5) \]

In Equation (5.5), ‘\( \sum w_i \)’ refers to the total weight of all nodes where ‘\( W_X \)’ represents the weight of Node X. The proposed AGAVRM method implements Grid Booster algorithm to achieve Job Allocation and reduce the idle time of a machine as shown in the following Figure.
### Input:
CPU Weight $W_{CPU}$, Memory Weight $W_{Mem}$, Weight for current load $W_{load}$

### Output:
Reduced idle time

<table>
<thead>
<tr>
<th>Step 1: Begin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step 2: Collect resource information from each parallel node</td>
</tr>
<tr>
<td>Step 3: Calculate Weight for each resource separately</td>
</tr>
</tbody>
</table>

\[
W_{CPU}, W_{Mem} \text{ and } W_{load}
\]

| Step 4: Determine the weight of a node using equation (5.4) |
| Step 5: Calculate required job Percentage for a machine using equation (5.5) |
| Step 6: Divide the job based on job proportion |
| Step 7: Distribute the job among parallel nodes |
| Step 8: End |

---

**Figure 5.4 Grid Booster Algorithm**

As shown in Figure 5.4, Jobs submitted by the user are distributed among parallel nodes in grid to provide better outcome for the users. Grid Booster algorithm performs the weight calculation of a node followed by required job proportion measurement based on weight metrics. Grid Booster algorithm effectively reduces the idle time to improve grid performance using the proposed AGAVRM method.
5.4.2 Resource Provisioning with Acclimatized Genetic Algorithm (AGA)

The proposed AGAVRM method implements Acclimatized Genetic Algorithm based VM resource migration to effectively schedule the jobs with load balancing in grid. This algorithm measures the impact on the system in advance after adding a new VM resource in grid with the help of past data and current state of the system. Acclimatize Genetic Algorithm obtains the solution such that least effect on the system is selected. Hence, better load balancing is achieved with minimum number of dynamic VM migrations. Acclimatized Genetic Algorithm is implemented in proposed AGAVRM method by performing selection, crossover, and mutation to schedule the jobs. Here Rank R is obtained by using the following equation.

\[
R = \sum_{i=1}^{N_j} \frac{W_i}{\max(W_i)} + \sum_{i=1}^{N_j} \frac{\max(P_i) - P_i}{\max(P_i)} + \sum_{i=1}^{N_{vm}} \frac{T_i}{N_{vm}} + \sum_{i=1}^{N_j} \frac{\min(C_i) - C_i}{N_j} \tag{5.6}
\]

From Equation (5.6), ‘N_j’ represents the Number of jobs, ‘N_vm’ refers to Number of Virtual Machines, ‘W_i’ stands for Weight of i\(^{th}\) Job, ‘P_i’ represents Communication Cost of i\(^{th}\) Job, ‘T_i’ refers to the Idle time of i\(^{th}\) VM, and ‘C_i’ represents Migration cost of i\(^{th}\) Job.

In the following equation to find the rank value, Percentage of waiting time ‘PW’, Percentage of idle time ‘PI’, percentage of communication cost ‘PC’ and percentage of replication cost ‘PR’ are measured in the proposed AGAVRM method as shown below.

\[
P_W = \frac{\text{Sum (waiting time for each job)}}{\text{highest waiting time} \times \text{number of jobs}} \tag{5.7}
\]
Based on the measurements of $PW, PI, PC$ and $PR$ that are obtained from Equations (5.7) to (5.10), rank is calculated by adding the above values (equation 5.11). Acclimatized Genetic Algorithm in proposed AGAVRM method selects the sequence with minimum rank. Hence, rank is reduced for generating new sequence. By utilizing Acclimatized Genetic Algorithm, $PW, PI, PC$ and $PR$ factors are effectively reduced. Figure 5.5 shows the steps involved in Acclimatized Genetic Algorithm as follows.

\[
PI = \frac{\text{Sum (idle time for each node)}}{\text{highest idle} \times \text{number of jobs}} \tag{5.8}
\]

\[
PC = \frac{\text{Sum (communication cost of each node)}}{\text{Maximum communication cost} \times \text{number of jobs}} \tag{5.9}
\]

\[
PR = \frac{\text{Sum (replication cost of each node)}}{\text{Maximum replication cost} \times \text{number of jobs}} \tag{5.10}
\]

\[
\text{Rank} = PW + PI + PC + PR \tag{5.11}
\]
**Input:** Number of jobs, resources weight, load prediction  
**Output:** Improved job scheduling efficiency

<table>
<thead>
<tr>
<th>Step 1: Begin</th>
</tr>
</thead>
</table>

**Step 2:** Collect the information about jobs  
**Step 3:** Randomly generate initial sequences where each sequence may contain job as well as node  

**Step 4:** *Selection process* - generate 8 sequences  
**Step 5:** First copy 4 sequences from initial sequences  
**Step 6:** Generate 4 remaining sequences as randomly select sequences from initial sequences with best fitness value  

**Step 7:** *Cross over process* - Generate 8 sequences  
**Step 8:** First copy 4 sequences from selection sequences  
**Step 9:** Remaining 4 sequences are generated as combining selection sequences  

**Step 10:** *Mutation process* - Generate 8 sequences  
**Step 11:** First copy 4 sequences from Cross over sequences  
**Step 12:** Remaining 4 sequences are generated as randomly select a sequences from above  

**Step 13:** Perform some interchange in the sequence  
**Step 14:** Sort these sequences on rank base, select first n sequences with minimum weight using equation 5.11  
**Step 15:** End

*Figure 5.5  Acclimatized Genetic Algorithm*
As shown in Figure 5.5, Acclimatized Genetic Algorithm performs effective job scheduling with improved job scheduling efficiency with the help of Selection, Cross over, and Mutation operations. Better sequences are obtained based on minimum weight in proposed AGAVRM method.

5.5 EXPERIMENTAL EVALUATION

Proposed AGAVRM method utilizes the Amazon Simple Storage Service (Amazon S3) dataset for easily schedules the job between the cloud users. Amazon S3 is experimented using JAVA coding. Amazon S3 is the warehouse of data containing images, files and other type of valuable information. Amazon S3 stores the data objects for many services and permits the concurrent read and write access.

Therefore, the proposed AGAVRM method is designed for job scheduling in grid computing. Number of users and user requests are considered from 5 to 50 to conduct the experiments with 10 iterations. AGAVRM method performs the experimental evaluation and compared the results against with the following parameters. Performance of proposed AGAVRM method is analyzed in terms of the parameters such as job scheduling efficiency, idle time and scalability as follows.

5.6 RESULTS AND DISCUSSION

Proposed Acclimatized Genetic Algorithm based VM Resource Migration (AGAVRM) method is compared with the existing methods such as Improved load balancing on Enhanced GridSim with Deadline Control (IEGDC) developed by Deepak Kumar Patel and Chitaranjan Tripathy (2014)
and Priority-grouping method (Priority) developed by GoodheadTomvie Abraham et al. (2015).

5.6.1 Impact of JobSchedulingEfficiency

Job scheduling efficiency is computed as the ratio of number of successfully scheduled jobs according to the total number of user request for accessing the grid resources. It is measured for analyzing the performance of proposed AGAVRM method which performs jobs scheduling. Job scheduling efficiency is mathematically described as follows.

\[
JSE = \frac{\text{no. of successfully scheduled jobs}}{\text{total no. of user requests}} \times 100 \quad (5.12)
\]

From equation (5.12), Job scheduling efficiency ‘\(JSE\)’ is measured in terms of percentage (%). Higher job scheduling efficiency ensures the better performance of a method.

**Table 5.1 Tabulation of Job Scheduling Efficiency**

<table>
<thead>
<tr>
<th>Number of user requests</th>
<th>Existing IEGDC</th>
<th>Existing Priority</th>
<th>Proposed AGAVRM</th>
<th>AGAVRM VsIEGDC</th>
<th>AGAVRM VsPriority</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>70.1</td>
<td>66.3</td>
<td>81.3</td>
<td>16.0</td>
<td>22.6</td>
</tr>
<tr>
<td>10</td>
<td>72.3</td>
<td>68.1</td>
<td>84.6</td>
<td>17.0</td>
<td>24.2</td>
</tr>
<tr>
<td>15</td>
<td>74.5</td>
<td>69.5</td>
<td>86.9</td>
<td>16.6</td>
<td>25.0</td>
</tr>
<tr>
<td>20</td>
<td>76.2</td>
<td>71.7</td>
<td>89.1</td>
<td>16.9</td>
<td>24.3</td>
</tr>
<tr>
<td>25</td>
<td>77.5</td>
<td>72.8</td>
<td>90.3</td>
<td>16.5</td>
<td>24.0</td>
</tr>
<tr>
<td>Numbe r of user request s</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------------</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td></td>
<td>Existing IEGDC</td>
<td>ExistingPriority</td>
<td>Proposed AGAVR M</td>
<td>AGAVR M VsIEGDC</td>
<td>AGAVR M VsPriority</td>
</tr>
<tr>
<td>30</td>
<td>79.1</td>
<td>74.5</td>
<td>91.5</td>
<td>15.7</td>
<td>22.8</td>
</tr>
<tr>
<td>35</td>
<td>80.5</td>
<td>76.4</td>
<td>92.7</td>
<td>15.2</td>
<td>21.3</td>
</tr>
<tr>
<td>40</td>
<td>81.6</td>
<td>77.3</td>
<td>94.1</td>
<td>15.3</td>
<td>21.7</td>
</tr>
<tr>
<td>45</td>
<td>82.7</td>
<td>78.9</td>
<td>94.8</td>
<td>14.6</td>
<td>20.2</td>
</tr>
<tr>
<td>50</td>
<td>84.1</td>
<td>80.3</td>
<td>95.6</td>
<td>13.7</td>
<td>19.1</td>
</tr>
<tr>
<td>Average value</td>
<td>77.86</td>
<td>73.58</td>
<td>90.09</td>
<td>15.8</td>
<td>22.5</td>
</tr>
</tbody>
</table>

**Sample calculation formula**

\[
= \frac{\text{Proposed} - \text{Existing}}{\text{Existing}} \times 100
\]

Consider the job scheduling efficiency of proposed method is 81.3 and existing₁ is 70.1 and then existing₂ is 66.3 from Table 5.1.

Calculation formula = \(\frac{\text{Proposed} - \text{existing}_1}{\text{existing}_1} \times 100\)

\[
= \frac{81.3 - 70.1}{70.1} \times 100
\]

\[
= 15.97
\]

By using calculation formula = \(\frac{\text{Proposed} - \text{existing}_2}{\text{existing}_2} \times 100\)

\[
= \frac{81.3 - 66.3}{66.3} \times 100
\]

\[
= 22.62
\]
Similarly all values are calculated using this formula and finally average of these values is calculated and noted in fifth and sixth column of Table 5.1.

Table 5.1 shows the tabulation of job scheduling efficiency based on number of user requests for the proposed AGAVRM and different existing methods. Number of user requests for accessing grid resources is taken as input which is varied from the range of 5 to 50 for experimental purpose. As shown in Table 5.1, job scheduling efficiency gets improved gradually for the corresponding increase in number of user requests for all the methods. Though, maximum job scheduling efficiency is achieved in the proposed AGAVRM method when compared to other existing methods.

![Figure 5.6 Measurement of Job Scheduling Efficiency](image)

Figure 5.6 demonstrates the measurement of job scheduling efficiency using different methods such as proposed AGAVRM method, existing IEGDC method developed by Deepak Kumar Patel and Chitaranjan Tripathy (2014) and existing Priority method developed by
Goodhead Tomvie Abraham et al. (2015). Figure 5.6 clearly shows that, the proposed AGAVRM method significantly enhances job scheduling efficiency when compared to the existing methods. This higher job scheduling efficiency is provided by the proposed AGAVRM method with the implementation of Acclimatized Genetic Algorithm. In addition, load balancing on Grids is achieved with the help of Grid Booster Algorithm. Therefore, proposed AGAVRM method improves job scheduling efficiency by around 16% when compared to existing IEGDC method and around 23% when compared to existing Priority method respectively.

5.6.2 Impact of Idle Time

Idle time is measured for grid resources as the amount of time a particular resource was not busy with respect to number of grid users. Idle time is mathematically formulated as shown in the following Equation.

\[ IT = \text{number of users} \times \text{Time (resource was not busy)} \quad (5.13) \]

From Equation (5.13), Idle Time ‘\( IT \)’ is measured in terms of milliseconds (ms). A method is said to be more efficient when the idle time is low. Hence, jobs given by the user are processed effectively by utilizing the grid resources.
Table 5.2  Tabulation of Idle Time

<table>
<thead>
<tr>
<th>Number of User Requests</th>
<th>Existing IEGDC</th>
<th>Existing Priority</th>
<th>Proposed AGAVRM</th>
<th>AGAVRM VsIEGDC</th>
<th>AGAVRM VsPriority</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>2.7</td>
<td>4.1</td>
<td>1.9</td>
<td>-29.6</td>
<td>-53.7</td>
</tr>
<tr>
<td>10</td>
<td>3.4</td>
<td>4.9</td>
<td>2.5</td>
<td>-26.5</td>
<td>-49.0</td>
</tr>
<tr>
<td>15</td>
<td>4.6</td>
<td>5.8</td>
<td>3.6</td>
<td>-21.7</td>
<td>-37.9</td>
</tr>
<tr>
<td>20</td>
<td>5.1</td>
<td>6.3</td>
<td>3.9</td>
<td>-23.5</td>
<td>-38.1</td>
</tr>
<tr>
<td>25</td>
<td>5.8</td>
<td>7.4</td>
<td>4.2</td>
<td>-27.6</td>
<td>-43.2</td>
</tr>
<tr>
<td>30</td>
<td>6.2</td>
<td>7.9</td>
<td>4.8</td>
<td>-22.6</td>
<td>-39.2</td>
</tr>
<tr>
<td>35</td>
<td>6.9</td>
<td>8.6</td>
<td>5.4</td>
<td>-21.7</td>
<td>-37.2</td>
</tr>
<tr>
<td>40</td>
<td>7.4</td>
<td>9.4</td>
<td>6.1</td>
<td>-17.6</td>
<td>-35.1</td>
</tr>
<tr>
<td>45</td>
<td>8.3</td>
<td>10.1</td>
<td>7.3</td>
<td>-12.0</td>
<td>-27.7</td>
</tr>
<tr>
<td>50</td>
<td>9.7</td>
<td>10.8</td>
<td>8.5</td>
<td>-12.4</td>
<td>-21.3</td>
</tr>
<tr>
<td>Average Value</td>
<td>6.01</td>
<td>7.53</td>
<td>4.82</td>
<td>-21.5</td>
<td>-38.2</td>
</tr>
</tbody>
</table>

Table 5.2 presents the tabulation of idle time in terms of number of users on grid using proposed AGAVRM and different existing methods. Number of users is taken from the range of 5 to 50 for conducting experiments. From Table 5.2, it is clear that for the respective increase in number of user requests, idle time is also improved for all methods. Among these methods, proposed AGAVRM method effectively reduces the idle time on grid resources when compared to other existing methods.
Figure 5.7 Measurement of Idle Time

Figure 5.7 illustrates the measurement of idle time for proposed AGAVRM method which is compared with the existing methods namely IEGDC developed by Deepak Kumar Patel and Chitaranjan Tripathy (2014) and Priority method developed by Goodhead Tomvie Abraham et al. (2015). As shown in Figure 5.7, idle time is effectively minimized in the proposed AGAVRM method when compared to other existing methods. This efficient reduction of idle time is achieved in proposed AGAVRM method by utilizing Grid Booster Algorithm where weight of each resource is measured separately. Then, job Percentage for a machine is obtained based on which the jobs are distributed among parallel nodes. Hence, proposed AGAVRM method reduces the idle time for grid resources by around 22% when compared to existing IEGDC method and around 38% when compared to existing Priority method respectively.
5.6.3 Impact of Scalability

Scalability is measured from the ratio between number of users addressed for their request on accessing grid resources through scheduling and total number of grid users. Scalability is mathematically formulated as shown below.

\[
\text{Scalability} = \frac{\text{number of users addressed for their request}}{\text{total number of users}} \times 100
\]  

From Equation (5.14), Scalability is measured in terms of percentage (%). If scalability is high, then the method is said to be more effective.

Table 5.3 shows the scalability measurement tabulation based on number of users for the proposed AGAVRM and different existing methods. Number of users is taken as input which is considered from the range of 5 to 50 for experimental purpose.
Table 5.3  Tabulation of Scalability

<table>
<thead>
<tr>
<th>Numbe r of user request s</th>
<th>Scalability (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Existing IEGDC</td>
</tr>
<tr>
<td>5</td>
<td>68</td>
</tr>
<tr>
<td>10</td>
<td>69</td>
</tr>
<tr>
<td>15</td>
<td>70</td>
</tr>
<tr>
<td>20</td>
<td>71</td>
</tr>
<tr>
<td>25</td>
<td>73</td>
</tr>
<tr>
<td>30</td>
<td>74</td>
</tr>
<tr>
<td>35</td>
<td>76</td>
</tr>
<tr>
<td>40</td>
<td>77</td>
</tr>
<tr>
<td>45</td>
<td>79</td>
</tr>
<tr>
<td>50</td>
<td>81</td>
</tr>
<tr>
<td>Average value</td>
<td>73.8</td>
</tr>
</tbody>
</table>

From Table 5.3, it is clear that scalability is increased for the respective increase in number of users for all methods. However, the proposed AGAVRM method provides better performance in terms of improving scalability when compared to other existing methods.
Figure 5.8 Measurement of Scalability

Figure 5.8 shows the measurement of scalability for the proposed AGAVRM method which is compared with the existing methods such as IEGDC developed by Deepak Kumar Patel and Chitaranjan Tripathy (2014) and Priority method developed by Goodhead Tomvie Abraham et al. (2015). As shown in Figure 5.8, scalability is comparatively improved in the proposed AGAVRM method when compared to other methods. Maximum scalability is achieved in the proposed AGAVRM method with the help of Acclimatized Genetic Algorithm where scheduling priorities are utilized. Hence, scalability is enhanced in the proposed AGAVRM method by 14% when compared to existing IEGDC method and around 19% when compared to existing Priority method respectively.
Acclimatized Genetic Algorithm based VM Resource Migration (AGAVRM) method was developed in Grid Networking environment with the aim of scheduling the user jobs with maximum job scheduling efficiency and scalability. Proposed AGAVRM method achieve effective job scheduling with the help of resource availability and future load prediction. Initially, Grid Booster Algorithm was employed for balancing the load in grid with minimum idle time of resources. After that, resource weight value was obtained based on each resource’s capacity to make decision on load balancing and allocate VMs. Finally, Acclimatized Genetic Algorithm was implemented in proposed AGAVRM method for providing the optimal solution on reducing VM resource migration and improving job scheduling efficiency. Experimental results showed that, the proposed AGAVRM method reduce the idle time by around 30% (Average of 21.5% and 38.2%), improved job scheduling efficiency and scalability by approximately 20% (Average of 15.8% and 22.5%) and approximately 17% (Average of 14% and 19.3%) when compared to existing methods. In the next chapter the analysis of proposed job scheduling methods with health records maintenance is discussed.