CHAPTER 4

WEIGHTED RANK BASED SCHEDULING FOR META BROKERS IN HETEROGENEOUS COMPUTATIONAL GRID ENVIRONMENT

4.1 INTRODUCTION

This chapter deals with an advanced job scheduling algorithm for MB in grid. A Weighted Rank Based Scheduling (WRBS) is developed and implemented for scheduling the jobs and allocating the resources. The WRBS job scheduling strategy is best suited for balancing uneven, heavily loaded and continuous workload under the worst conditions. Here, the worst conditions means that whenever there are continuous heavy workloads, the existing systems may fail to accommodate all the incoming jobs and there may also be a buffer overflow, requiring reallocation of jobs badly degrading the system performance with a tendency to compromise on the QoS.

In order to address such kinds of heavy, complex and continuous workloads, the system is dealt with, using the proposed Weighted Rank Based Scheduling (WRBS) strategy developed for the MB to schedule the incoming jobs. The system introduces and deals with two levels of middleware namely, the MB and multiple RBs for work load distribution, greatly influencing such scenario to bring out a remarkable performance enhancement of the computational grid system.
The concept deals with the grid system, having heterogeneous computational grid environment, consisting of various grid resources. The MB is connected with an AGIS in the first level of the middleware. The second level of the middleware consists of several RBs, each connected with its corresponding GIS. The MB and the RBs consult the AGIS and the corresponding GIS each time, before forwarding the jobs. The proposed WRBS is implemented, which performs its functions in the MB for distributing the workload to the various second level RBs based on their ranks assigned.

This chapter deals with the heterogeneous grid environment, effective resource management, MB and AGIS, as the introductory part, and then, it deals with evaluating the performance of the system with a single RB with that of multiple RBs by proposing WRBS algorithm.

The heterogeneous grid environment is a collection of various similar computational resources, grouped together in the form of clusters, and linked together using a high speed networks, capable enough of sharing devices like processors, hard disks, huge storage devices, printers, RAM and so forth. The computational grid environment contains scattered systems which allow distribution, selection and combination of globally scattered computational resources, based on their accessibility, potential, price and client demands in order to compute a huge volume of user applications. This system serves as a cheaper alternative for supercomputers, for these environment aggregates several computing organizations in order to attain an objective capable enough to handle multiple jobs simultaneously. The resources are made available. Whenever they are required that are not physically seen, as the computing resources are virtually connected with each other.
In a grid environment, the synchronization of the applications is not an easy task, especially, while synchronizing the flow of data among the scattered computational resources and therefore, specialized grid workflow systems are designed for organizing and running a sequence of jobs within the grid environment.

It is the ultimate goal of this technique to minimize the overall time taken for executing all the arriving jobs, and the efficient usage of all the prevailing resources plays a vital role. Distributing a job to a congested resource consumes excess time on waiting. The grid environment is autonomous in nature, and so, exploring the suitable resource remains a tough task to be performed. Huge scientific applications require high speed processors and the best possible scheduling policies with the best resource management.

The grid computing also holds the responsibility of resource management which plays a pivotal role in grid environment, and the fundamental work is to understand the resource requirements of the machines within the grid environment and assigning suitable resources to match the requirements from the resource repository with proper user knowledge and authorization. The ultimate aim of resource management system is to match specific requests to suitable resources, allocating the suitable resources and running the requests by making use of allocated resources. The grid resources comprise processors, disk spaces, memory management, bandwidth over the network and so on controlled by the resource management system. The resource management systems are broadly classified into many types, out of which, a few are used in this work

Service Level Agreement–Supported Resource Management whose purpose is to imitate the business demands of the service provider and to focus on highly independent and dynamic resource design that involves
service contracts, analyzing risks and acquiring proper design information. This helps the system in aggregating the computing resources from various organizations to meet the demands of the users and to quickly process the workload applications without compromising the QoS demands.

Agent based Grid Resource Management is an approach that offers techniques to provide adjustable, error free and expandable resource management services. The agent matches the requests along with the available resources and allocates the suitable resources for fulfilling the business demands. Every agent has abilities to assign different scheduling methods and broadcast their potential to identify services. The fundamental idea is to execute a requested resource with the available resources from the resource repositories that helps the system in such a way that the grid agents may have the ability to quickly interact with the various levels of middleware and to find the resource availability which matches the exact application to be processed. This helps the system to quickly process the jobs.

Failure Aware Grid Resource Management is a technique that suits any environment, where the scattered resources are subjected to controls and failures. The technique suitably adjusts the real load conditions and decides the re-matching intervals for reducing the chances of miscalculating the failure times, for a task is re-matched before it gets ended. As the failure of the resources are inevitable in grid systems, this technique brings awareness of the failure of resources, so that necessary alternate options can be used for tolerating the faults or to avoid assigning jobs to faulty resources.

Performance based Grid Resource Management system estimates the performance of grids, defines three important roles namely, the users, resources and the system for controlling the resources. The requests for the resources are given to the system rather directly to the resources and the users suitably select their resource through the system for performing their jobs.
where every independent resource is controlled by a single resource management system. In other words, the resources are totally controlled by the middleware component where the status of the resources are frequently updated in GIS, as well as AGIS, helping the optimum utilization of resources.

An MB is a kind of middleware that helps in performing an efficient allocation of tasks to improve the QoS by managing the resources in a better manner. The role of the meta scheduler is to have a track on the RBs assigned with ranks and the incoming jobs are received by the MB initially and then, routed to the RBs by analyzing the size of the jobs and the ranks assigned to the RBs. This helps the system fairly distribute the workload among the RBs, based on their ranks which in turn helps to improve the performance of the system. The MB is highly applicable in circumstances, where the system receives uneven and heavily loaded jobs to be processed. Linked with the corresponding AGIS, the MB consults the AGIS each time, before allocating the jobs to the corresponding RBs.

The AGIS is one of the active components of the grid system designed in such a way that it holds the status of the RBs. The information present in the AGIS is updated frequently based on a certain threshold value. Once the MB receives the jobs from the user, it refers to the AGIS and identifies to which particular RB the jobs need to be forwarded for processing, helping the system improve the performance, when the system receives heavy and uneven workloads, providing information to the meta schedulers to schedule the jobs.
4.2 SYSTEM PERFORMANCE WITH A SINGLE RESOURCE BROKER

The conventional grid system holds only one RB for forwarding the arriving jobs to particular resources, prevailing within the grid environment. The GIS is consulted by the RB for obtaining all the possible information about the resources that links the jobs in accordance with the allocation policies. The ultimate objective is to allocate the arriving jobs and stabilizing the loads among the prevailing resources in order to attain an efficient organization of the grid resources and the grid framework with a single RB is depicted in Figure 4.1.

![Figure 4.1 Single resource broker framework](image)

The client submits the job to the grid system for processing and the RB receives the jobs and consults the GIS for resource availability. The RB is associated with a set of grid resources always available for it, termed as Processing Elements (PEs). The PEs denotes the processing nodes or the CPUs available with the resource pool, and the status of the PEs is maintained...
in the GIS and frequently updated. The RB consults the GIS each time, when it receives the jobs from the users, which are split into equal sized tasks called gridlets and assigned to the suitable, available free PEs for processing, returning the processed gridlets to the clients.

In this system, the efficient usage of the prevailing resources consistently bothers the QoS necessities. The large scientific applications require more computational services and hence a suitable linking of resources is needed in order to achieve optimal services. The suitable linking of the resources provides an efficient consumption of the prevailing resources by stabilizing the loads within the resources present in grids, depending on the policies employed for allocation which make use of several algorithms and frequent update the status of the resources.

The transformation related to the service quality and price for designing the system are involved. Various mechanisms like test summit and redundancy are employed to handle the fault within the system and to improve the service provided by the system. The detection of the faulty resources is not an easy task during job execution. The system gains merits like reduced complexity, QoS attainment and cost reduction, however it faces major challenges during load stabilization, increased QoS, efficient resource consumption, tolerance towards fault and large computational resources for handling complex and scientific applications.

The performance of the system with single RB is tested for varying loads and are shown in Table 4.1.
<table>
<thead>
<tr>
<th>No. of Gridlets</th>
<th>Execution Time using Single Resource Broker (Milliseconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>55.120</td>
</tr>
<tr>
<td>6</td>
<td>55.130</td>
</tr>
<tr>
<td>12</td>
<td>55.170</td>
</tr>
<tr>
<td>24</td>
<td>55.290</td>
</tr>
<tr>
<td>48</td>
<td>75.160</td>
</tr>
<tr>
<td>96</td>
<td>118.050</td>
</tr>
<tr>
<td>192</td>
<td>198.050</td>
</tr>
<tr>
<td>384</td>
<td>369.061</td>
</tr>
<tr>
<td>768</td>
<td>661.015</td>
</tr>
<tr>
<td>1536</td>
<td>1249.119</td>
</tr>
<tr>
<td>3072</td>
<td>2530.130</td>
</tr>
</tbody>
</table>

The system is experimented with varying number of gridlets ranging from 3 gridlets to 3072 gridlets and the performance of the system is experimented. The corresponding execution time of those numbers of gridlets are also observed and represented in Figure 4.2. The performance of the system with single RB is analysed for varying number of loads and the corresponding execution time of gridlets is noted.
The scheduling mechanism with a single RB yields satisfying results working well when the system load is less and the arrival rate of jobs are moderate, however it becomes a challenging task, when there are continuous and uneven heavy loads. So, the system is further extended for addressing the above categories of workload by introducing multiple resource brokering strategy with a MB discussed below.

4.3 WEIGHTED RANK BASED SCHEDULING ALGORITHM

This technique allows the grid environment to utilize several RBs for work load distribution. Each resource groups is governed by different RBs and subjected to a limited processing speed further linking RBs to the GIS. The resources within a group enroll themselves to that particular GIS in order to assist job forwarding.
The technique employs a MB which is responsible for gathering the equal sized jobs for frequent forwarding of them to the subsequent level of RBs based on their weights. All the information related to the second level of brokers is maintained within AGIS. This technique allows minimizing the job execution time for the jobs submitted by the user. The MB utilizes weight based rank schemes for linking the arrived jobs to the suitable resources within the pool of resources. The allocation scheme utilizes two main factors for selecting the resource groups namely the load factor and rank assignment allotted to different resource groups.

The load factor is defined as the number of active PEs out of the total number of available PEs. The loads on the resources are estimated for every group and it serves as a load factor for allotting ranks to the resource groups. The resources are grouped based on the speed of processing where already defined weights are allotted to every resource group which is proportional to the speed of processing.

The rank assignment for a resource group is defined as the load factor of the resource group divided by the weight of the same resource group where the weight depends upon the speed of the processing element.

Depending on the ranking of the resource groups, the optimal resources are selected for running the arriving gridlets, and the framework of the proposed mechanism is shown in Figure 4.3.
Figure 4.3 Multiple resource broker architecture

The system proposes several RBs within the grids, containing a MB at the initial level and four RBs at the second level. An AGIS is maintained by the MB and all the RBs at the second level contain an independent GIS.

Once the user submits the job, the MB receives them and quickly consults the AGIS, after which it is forwarded to the second level of RBs for further consultation with their own GIS for obtaining the status of their own resource groups. Each resource group varies in its speed of processing and the processing capacity of group ‘a’ is in the range of (0, 750) MIPS. The processing capacity of group ‘b’ varies between (750, 1500) MIPS, the
processing capability of group ‘c’ is in the range of (1500, 3000) MIPS and the processing capacity of group ‘d’ is in the range of (3000, ∞) MIPS.

4.4 ALLOCATION STRATEGY

Once the user submits the jobs, the meta scheduler receives them, the algorithm developed for the meta scheduler collects them and instantly consults the AGIS for forwarding them to the second level brokers. The decisions are purely depending on the weights based on which ranks are assigned during the allocation. The allocation technique estimates the load factor for every resource groups by using the Equation (4.1).

\[
\text{Load Factor}_i (LF) = \left( \frac{\text{No. of active PEs}_i}{\text{Total PEs}_i} \right)
\]  

(4.1)

where, ‘i’ represents the group of Processing Elements (PEs).

Each of the four groups is allocated weight in accordance with the speed of the processing. Group ‘a’ is allocated a weight of ‘1’, group ‘b’ is allocated a weight of ‘2’, group ‘c’ is allocated a weight of ‘4’ and group ‘d’ is allocated a weight of ‘8’.

The relation between the weight and the speed for processing is represented using the following relation as shown in Equation (4.2).

\[
\text{Weight}_i (W_i) = \frac{\text{Speed}_i \text{ of CPU}}{750}
\]  

(4.2)

where, ‘i’ represents the group of PEs.

The rank is estimated by using the load factor (LF) and the weight (W_i) of the particular group as represented below in Equation (4.3).

\[
\text{Rank}_i (R_i) = \frac{\text{Load Factor}_i (LF)}{\text{Weight}_i (W_i)}
\]  

(4.3)
The ranks are set in rising order as \( R_a, R_b, R_c \) and \( R_d \) related to various groups \( a, b, c \) and \( d \) respectively with diverse occurrences. A random probability allocation function is used for selecting a suitable resource group in order to process the gridlets.

The organization of resources and the allocation of applications within a huge scattered system are very tedious. In order to demonstrate the efficiency of the RBs and related allocation algorithms, it is necessary to estimate their performance under various situations, like differing workloads with differing resources and clients with diverse demands. It is difficult in a grid environment to estimate the performance of a scheduler in a well-organized and frequent manner as resources and the clients are scattered among several organizations owing to their own strategies.

The GridSim tool is used to generate a simulation environment, which simulate various groups of diverse resources, users, applications, RBs and allocators. The RBs play a vital role in choosing the apt resources which literally means that every client owns a RB intended to serve the best resources and to satisfy the user demands, while the job of the allocator is to organize the resources like groups owing full power over the strategies used for resource scheduling. It is clear that all the jobs are to be surrendered to the central allocator which aims to perform a high consumption of available resources, meeting user demands based on the resource scheduling strategies.
Algorithm: Weighted Rank based Scheduling for job allocation using Meta Broker and Resource Brokers

**Input:** Jobs

**Output:** Execution time of gridlets

IF MB receives an equal sized chunks THEN

(a) AGIS is consulted; Let the jobs be routed to subsequent levels of RB for consulting GIS to gather information related to the resource status.

(b) IF resources are available, the jobs are scheduled to those resources which are ranked and prioritized based on their processing speed.

(c) GOTO Step 3.

ENDIF

IF MB accepts normal jobs or heavy jobs THEN

(a) AGIS is consulted: Let the jobs be directed to the subsequent levels of the RB for consulting GIS to know the current resource status.

(b) The resources are unavailable since the resources are not ready to spend large time for a normal job.

(d) Resource unavailable is reported and re-scheduling is allowed.

(e) GOTO STEP 3.

ENDIF

IF MB accepts unequal abnormal job THEN

(a) AGIS is consulted

(b) Let them be directed to subsequent levels of the RB; LCDA is performed by consulting GIS

(c) IF resources available, the jobs be scheduled ELSE let jobs be rescheduled

ENDIF

STEP 3: EXIT
Description

The MB consults the AGIS after receiving the jobs and those jobs are forwarded to the subsequent levels of RBs for further consultation with GIS for obtaining information related to the status of the prevailing resources.

Figure 4.4 Flow diagram

The RB’s are already ranked. The jobs meeting those criteria are sent to those particular groups for execution. In case of encountering normal jobs they are
sent back for re-scheduling. If the incoming jobs are abnormal and unequal then AGIS is consulted at once the MB receives the jobs. The job is routed to the subsequent RB based on the rank. RB further consults the GIS to gather information about the resource status and performs LCDA so as to assign jobs to the available resources. The flow of allocating the jobs to resources in this strategy is depicted in Figure 4.4.

4.5 EXPERIMENTAL STUDY

The performance analysis is conducted with varying number of gridlets as input to the system, and the corresponding execution time of the gridlets is shown in Table 4.2. The number of PEs in each group is set as 1000 for simulation purpose, and the corresponding graph representing the execution time of jobs is shown in Figure 4.5.

![Graph showing system performance using multiple resource brokers](image-url)

Figure 4.5 System performance using multiple resource brokers
Table 4.2 Comparison between execution time based on gridlets

<table>
<thead>
<tr>
<th>No. of Gridlets</th>
<th>Execution Time (milliseconds)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Single Resource Broker</td>
<td>Multiple Resource Brokers</td>
</tr>
<tr>
<td>3</td>
<td>55.120</td>
<td>54.120</td>
</tr>
<tr>
<td>6</td>
<td>55.130</td>
<td>55.030</td>
</tr>
<tr>
<td>12</td>
<td>55.170</td>
<td>55.100</td>
</tr>
<tr>
<td>24</td>
<td>55.290</td>
<td>54.785</td>
</tr>
<tr>
<td>48</td>
<td>75.160</td>
<td>56.965</td>
</tr>
<tr>
<td>96</td>
<td>118.050</td>
<td>57.275</td>
</tr>
<tr>
<td>192</td>
<td>198.050</td>
<td>62.834</td>
</tr>
<tr>
<td>384</td>
<td>369.061</td>
<td>85.919</td>
</tr>
<tr>
<td>768</td>
<td>661.015</td>
<td>151.21</td>
</tr>
<tr>
<td>1536</td>
<td>1249.119</td>
<td>291.014</td>
</tr>
<tr>
<td>3072</td>
<td>2530.130</td>
<td>538.300</td>
</tr>
</tbody>
</table>

Table 4.2 shows the comparison between single RB and multiple RBs with WRBS algorithm with differing gridlets. The system performance is analysed and it is clearly evident that the system performance is comparatively better in terms of execution time for lesser number of gridlets. When the number of gridlets increases, the system with multiple RBs shows a significant improvement and the same is plotted using a graph.
Figure 4.6 Comparison analysis of system performance with single resource broker and multiple resource brokers

Figure 4.6 shows the performance analysis of the system with a single RB and multiple RBs. The system is tested for varying number of gridlets which ranges between 3 and 3072. The system performance is almost the same when tested with lesser workload. The performance seems to be outstanding, while the gridlets are gradually increased. It is clearly evident that the system with multiple RBs with WRBS is best suited for higher grid loads.

4.6 CONCLUSION

The chapter describes assigning ranks to RBs which are assigned with the incoming loads. The incoming job characteristics are used here for assigning ranks to the resource groups. After encountering a job by the MB, they are forwarded to the subsequent levels of RBs for further consultation with GIS to know the status of the prevailing resources. Based on these results, the jobs are allocated to those resources for execution based on the ranks assigned. This algorithm has proven to be better, when the number of
incoming gridlets is lesser and shows a outstanding performance when there is an increase in the number of jobs. It means that the proposed algorithm is highly efficient, when the system is heavily loaded with numerous gridlets.

The system is tested with the simulation environment to evaluate the performance of the system and the comparison analysis is also carried out with varying loads for both scheduling with a single RB and scheduling with multiple RBs by implementing WRBS. The system yields better results for both the strategies with lesser workload allotments, but a significant performance has been noted with higher workloads, while WRBS uses the multiple RB strategy.