CHAPTER 5

CONCLUSION

5.1 INTRODUCTION

The research, presented in this thesis, addresses the major issues of heterogeneous computational grid environment such as, scheduling the grid jobs, balancing the workload, resource fault tolerance through optimum utilization of resources and improving the overall performance of computational grid. The significant contributions and their impact in terms of performance are consolidated and described in this chapter.

The ultimate aim is to perform the job scheduling efficiently in diverse computational grid environments and to provide an effective job scheduling techniques along with fault tolerance mechanisms. Many conventional techniques are available to perform job scheduling which tend to work well in handling certain tasks, but they do not produce satisfactory results in handling more number of tasks. It is clear that only a few algorithms work well in scheduling jobs and provide good results. The proposed approaches provide suitable job scheduling algorithms for improving the performance of the grid system minimizing the overall execution time of all scheduled jobs.

Fault tolerant mechanisms are also concentrated upon for improving the system performance and for satisfying the demands of the users. The results obtained from simulations are plotted and compared with
the results of the conventional techniques and found that the proposed strategies outperform the conventional techniques in reducing the makespan, minimizing the execution time and processing time with the maximum utilization of the prevailing resources.

5.2 CONTRIBUTION OF THE THESIS

In order to meet the deterministic performance of the computation grid system, an Adaptive Job Scheduling algorithm is developed and simulated with effective fault tolerance, a RCDA and an improved LCDA for job scheduling having an effective load balancing and resource fault identification along with a Weighted Rank Based Scheduling for MBs. The excerpt of the above findings and their performance enhancement are compared with the existing system and presented in this section.

The ultimate objective of the proposed algorithm aims to

- Reduce the makespan
- Increase the average resource utilization
- Reduce the time for the execution of jobs
- Reduce the cost for processing the tasks

The algorithm ensures tolerance towards faults by completing the user’s tasks within time and reducing the total time of the execution of jobs. Consequently, the load must also be balanced by an increased average resource consumption.
5.2.1 An Adaptive Job Scheduling with Efficient Fault Tolerance Strategy

Developing an Adaptive Job Scheduling algorithm, the performance of the system is compared with that of other scheduling algorithms. The proposed scheduling strategy is considered to be efficient, especially for the light weight or fine grained jobs.

Computing the jobs is measured as Million Instructions Per Second (MIPS) with a fixed threshold value, generated upon estimating an average computational capability of the system. If any incoming job is less than the fixed threshold, then, the jobs are considered as fine grained jobs or light weight jobs else the job will be treated as a normal job. The usage of Adaptive Job Scheduling algorithm is based on the above classification.

It is possible to frequently update the information stored into the grid resource monitoring over an instance of time and also during the entry of heavy loaded jobs into the resource pool as per the schedule of the resource scheduler. This technique is based on Grid Resource Information Management (GRIM) prototype and Grid Resource Information Representation (GRIR) protocol and the characteristics of resources on grouping techniques which make use of mediators.

The Maxspan and Flow Time of the jobs are calculated each time, when the jobs are assigned to the resources during the execution time. The goal of the resource scheduler is attained by minimizing the Maxspan and Flow Time of allocated tasks respectively. The result obtained in this algorithm proves that the resources are effectively utilized for light weight jobs in minimizing the total execution time of the jobs.
5.2.2 Randomized Chunk Distribution Algorithm and Log Based Chunk Distribution Algorithm for Efficient Job Scheduling and Load Balancing

Novel scheduling strategies namely RCDA and LCDA are developed and considered in this thesis for improving the overall performance of the grid.

These algorithms use an RB as a middleware, which receives the job from the user and routes it to the clusters consisting of resources. These techniques improve the efficiency of the system by allowing the RB to gather prior knowledge regarding the duration of jobs. These algorithms divide the submitted user job into several equal-sized chunks within the RB.

RCDA is randomly distributed to the corresponding clusters with the computing resources, until all the chunks are distributed assuming that the waiting time of jobs in the RB is negligible, compared to that of its execution time.

LCDA works just like the above model, but the distribution of chunks are carried out by knowing the status of the clusters through implementing of log policies. Each cluster holds a cluster log and the same is maintained with the RB. The GIS holds the fault log and all the logs are updated frequently, maintaining current status of the resources. The RB refers to both the cluster log and fault log before routing any job and to know the current status of the resources.

The results are compared with those of the existing scheduling algorithms with varying loads to evaluate the performance of the system. The LCDA shows a better than RCDA and Serialized Chunk Distribution Algorithm.
5.2.3 Weighed Rank Based Scheduling Using Meta Broker for Scheduling and Load Balancing

The proposed Weighed Rank Based Scheduling (WRBS) is found to be effective for the grid systems, when there are very high, uneven and continuous workloads.

This system consists of an MB connected with AGIS and multiple RBs, each connected with its corresponding GIS. All the RBs are registered with the MB and each RB has its own set of resources termed as PEs registered to it. Ranks are assigned to the RBs based on the computing capacity of the PEs.

The MB collects the user jobs and then forwards it to the next level of RBs. The WRBS algorithm, implemented in MB, routes the jobs to the corresponding RBs by referring to the AGIS, helping to map the input jobs to a suitable resources based on load factor evaluated for each resource class. The MB receives the incoming jobs and assigns it to the RBs based on the job size to be matched with the load factor assigned to the corresponding RBs. The jobs are further routed to the corresponding cluster for processing it. The proposed algorithm helps to considerably reduce the execution time of the user’s jobs.

The analysis clearly proves that the performance of the WRBS with single RB and WRBS with multiple RBs are approximately similar for lesser number of jobs and WRBS with multiple RBs shows an excellent improvement in performance with an increased number of jobs.
5.3 FUTURE RESEARCH DIRECTIONS

There are several interesting scopes still open in grid computing which requires further research, and some of them are discussed here briefly. The thesis has mainly focused on minimizing the execution time of jobs, optimum load balancing by proper utilization of resources, grid monitoring schemes for resource fault identification and obtaining high tolerance level.

As far as the job scheduling is concerned, the proposed algorithm splits the entire jobs into smaller size chunks and the chunk sizes are fixed. The future work can focus on the variable chunk size, meaning that the chunk sizes of a particular job should be equal and should be varying for next incoming jobs depending on the size of the job. The broker needs to correlate with the job size, chunk size and capacity of the resource, reducing the communication overhead helping improve the system performance further.

As far as the fault tolerance mechanism is concerned, most of the literature says that most of the fault tolerance mechanism depends mostly on two major approaches namely, backup approach and check pointing approach. But, backup approach has a drawback of space related issues and checkpoint mostly results with communication overhead, if there are no limitations on the number of check points.

Since the resource fault is inevitable and directly degrades the performance of the system, it is better to propose a smart code for resource fault identification to help the system to run on all computing resources and fault needs to be automatically reported to the RBs to identify and tolerate resource faults.

Any hybrid approach incorporating best scheduling with the smart code for fault tolerance can greatly improve the grid system performance.
5.4 CONCLUSION

The methodologies proposed in this research for the performance improvement of the computational grid system focused on job scheduling, workload balancing and fault tolerance. These approaches consider different cases with varying workloads and satisfy the user demands along with the QoS constraints. The proposed algorithms also obtain significant results with minimized makespan, increased system throughput through optimum utilization of resources and thus improves the overall performance of computational grid in a heterogeneous grid environment. The performances of the proposed algorithms has been published and are compared with the existing techniques considering various parameters. From the thesis, it is concluded that the above proposed algorithms provide a better performance for the grid systems.