International Journal of Multidisciplinary Research and Development

Volume: 2, Issue: 4, 177-180
April 2015
www.allsubjectjournal.com
e-ISSN: 2349-4182
p-ISSN: 2349-5979
Impact Factor: 3.762

S. Gayathri
M. Phil Research Scholar
Department of Computer Science Vivekananda College of Arts and Sciences for Women, Namakkal, Tamil Nadu, India

M. P. Subashini
Asst. Professor,
Department of Computer Science & Applications Vivekananda College of Arts and Sciences for Women, Namakkal, Tamil Nadu, India

Correspondence:
S. Gayathri
M. Phil Research Scholar
Department of Computer Science Vivekananda College of Arts and Sciences for Women, Namakkal, Tamil Nadu, India

Traffic and dynamic load balancing in grid computing system

S. Gayathri, M. P. Subashini

Abstract
Grid computing systems are distributed systems that involve coordinate and involvement of heterogeneous resources with various characteristics where user jobs can be executed on either local or remote computer. These heterogeneous computing resources are used to run highly complex programs that require very high processing power. The main benefit of this idea was to decrease the amount of messages exchanged between Grid resources. We proposed a scheduling algorithm that manages the resources to improve the utilization of resource and minimize the job response time in computational grid system. So that no any resources will be heavily, low loaded or in some case will be in idle. We have realized a significant improvement in mean response time with a reduction of communication cost.

Keywords: Grid computing, load balancing, workload, load migration, schedule-DLB, job scheduling

1. Introduction
The term "Grid Computing" refers to a distributed computing infrastructure for advanced Science & Technology. Grid computing helps in the effective utilization of computing resources over the intranet/internet. Computational grid systems are distributed systems developed for heterogeneous resources, that heterogeneous computing resources can be personal computers, laptops, supercomputers, clusters, software's etc. In order to fulfill the user expectations in terms of performance and efficiency, the Grid system needs efficient load balancing algorithms for the distribution of tasks. Load balancing algorithms in classical distributed systems, which usually run on homogeneous and dedicated resources cannot work well in the grid architectures. Grids has a lot of specific characteristics, like heterogeneity, autonomy, scalability, adaptability and resources computation-data separation, which make the load balancing problem more difficult. Our aim to propose an efficient job scheduling algorithm for grid system that provides efficient utilization of resources. This scheduling algorithm tries to minimize the total service time of the jobs and improve the response time for arriving jobs.

2. Load Balancing
A typical distributed system will have a number of interconnected resources who can work I independently or in cooperation with each other. Load balancing methods in conventional parallel and distributed systems has been intensively studied they do not work in Grid architectures because these two classes of environments are radically distinct. Schedule of tasks on multiprocessors or multi computers supposes that processors are homogeneous and linked with homogeneous and fast networks. Load balancing algorithms can be classified into two categories: static or dynamic

- In static load balancing, a task is assigned to an available resource when it is generated or admitted to the system using a fixed schema.
- In contrast to static load balancing, dynamic load balancing allocate/reallocate tasks to resources at runtime based on no priori task information, which may determine when and whose tasks can be migrated.

3. Challenges of Load Balancing In Grid
Although load balancing methods in conventional parallel and distributed systems has been intensively studied, they do not work in Grid architectures because these two classes of environments are radically the schedule of tasks on multiprocessors or multi computers

suppose that processors are homogeneous and linked with homogeneous and fast networks.

Heterogeneity: Heterogeneity exists in both of computational and networks resources.

- First, networks used in Grids may differ significantly in terms of their bandwidth and communication protocols.
- Second, computational resources are usually heterogeneous (processors, resource capabilities memory size and so on). Also different software's, like operating systems, file systems; cluster management software may be heterogeneous.

Scalability: In Grid there are no limits for the resources and that reaches from few resources to millions. And that arises problems of potential performance as the size of a Grid increases.

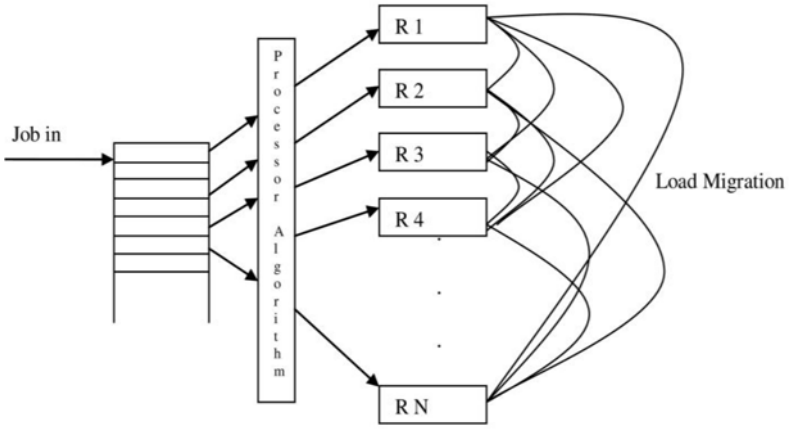

Fig 1: Load balancing to avoid overload

Adaptability: In a Grid, if failure accrues for a resource, that is not exception. These properties make the load balancing problem more complex than in traditional parallel and distributed systems, which offer homogeneity and stability of their resources. Also interconnected networks on grids have very disparate performances and tasks submitted to the system can be very diversified and irregular. These various observations show that it is very difficult to define a load balancing system which can integrate all these factors.

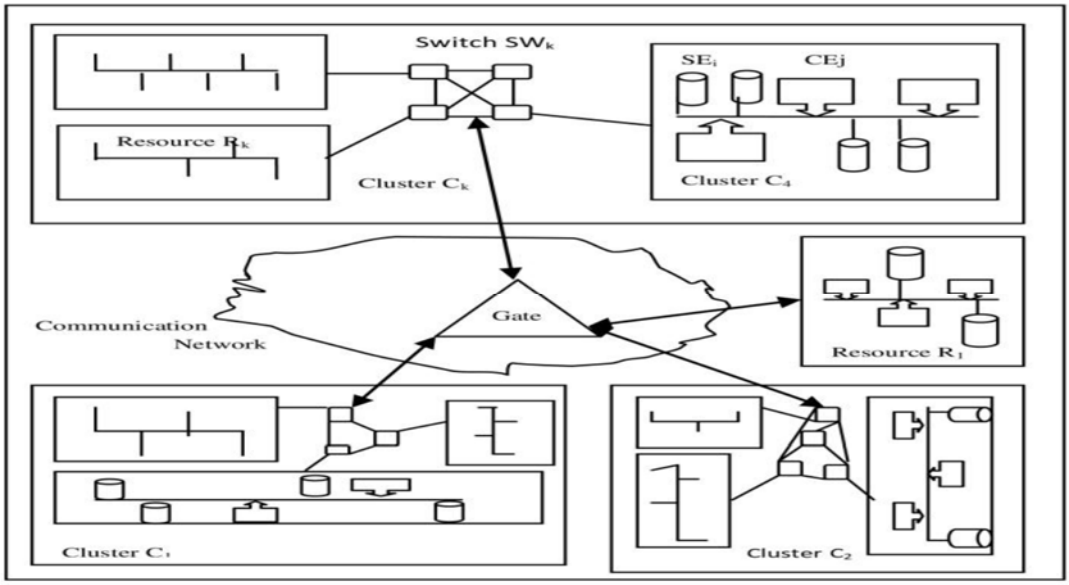

Fig 2: Grid Topology

In the grid computing (Fig. 2) there is a finite set of clusters Ck, which are interconnected by gates, where i,j,k ∈ {0, ...,G − 1}, where each cluster contains one or more resources Sj interconnected by switches SWk and each resource contains some Computing Elements CEj and some Storage Elements SEi, interconnected by a local area network.

4. Tree-Based Balancing Model

In order to well explain the proposal model, we must define the topological structure of a Grid computing.

Mapping a Grid into a tree-based model:

The load balancing strategy proposed in this is based on a mapping of any Grid into a tree-based model. It is build as follows:

- First, for each site we create a two levels subtree. The leaves of this sub tree correspond to the Computing Elements of a site and the root of this sub tree is a virtual node associated to the site. The role of this virtual node is to manage the workload of a site. In practice, this management function is processed by a computing element within the site.

- Second, the subtrees corresponding to all sites of a cluster are aggregated to generate a three levels subtree.
- Third, these subtrees are connected together to build a four levels tree. **Level 0:** In this first level (top level), we have a virtual node that corresponds to the root of the tree. It is associated to the Grid and it manages the workload on the whole Grid.

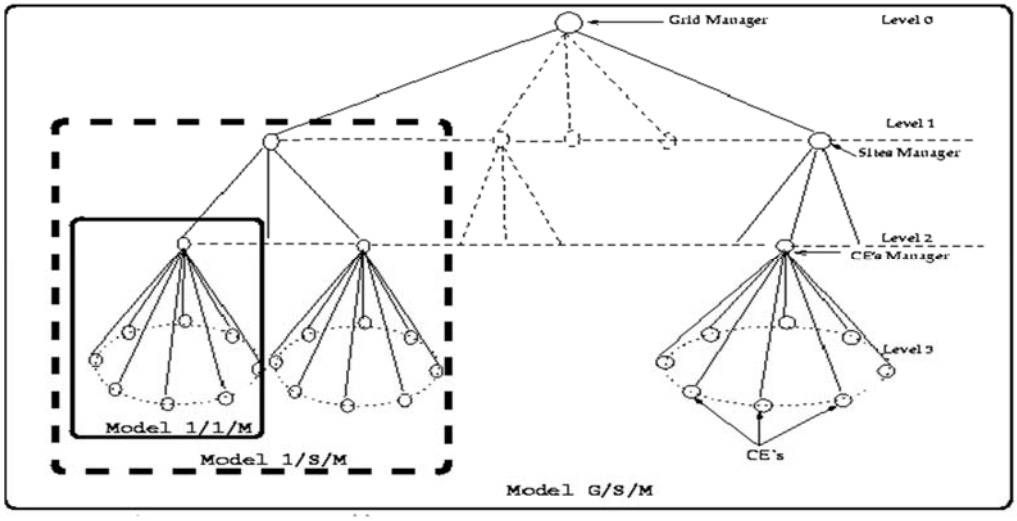

Fig. 3: Tree-based representation of a grid

Level 1: This level contains G virtual nodes, each one associated to a physical cluster of the Grid. In our load balancing strategy, this virtual node is responsible to manage its sites.
Level 2: In this third level, we find S nodes associated to physical sites of all clusters of the Grid. The main function of these nodes is to manage the workload of their physical Computing Elements.
Level 3: At this last level (leaves of the tree), we find the M Computing Elements of a Grid linked to their Respective and clusters. The final tree is denoted by *G/S/M*, where *G* is the number of Clusters that compose the Grid, *S* the number of Sites and *M* the number of CE's. As This generic tree can be transformed in turn into three specific trees:*G/S/M*, *1/S/M* and1/1/M, depending on the values of G, *S* and *M*. The mapping function generates a non cyclic connected graph where each level has specific functions.

5. Load Balancing Algorithm

There are some different algorithms for load balancing in Grid system.

Sender-Initiated Algorithm: In the sender initiated algorithms let the heavily loaded resources take the initiative to request the lightly loaded resources to take the jobs .There are three basic decisions that need to be made before a transfer of a job can take place.
1. Transfer policy: This policy decides when a resource become the sender and request to other resource.
2. Selection policy: This policy decides how any other resource chooses for transfer the job which is low loaded.
3. Location policy: This policy finds the location of the desired resource in the cluster. A resource can activate the sender initiated algorithms when its queue size exceeds over some threshold value when job is arrived.

Receiver-Initiated Algorithm: Receiver initiated algorithms is just like sender initiated algorithm in this algorithm low loaded resources search the heavily loaded resources and request to send their jobs to overcome the workload. It uses the similar transfer policy as the sender-initiated algorithm, which activates the pull operation when its queue length falls below a certain threshold, upon the departure of a job receive. Performance of the receiver initiated algorithm better than the sender-initiated algorithm for load balancing in grid system.

Symmetrically- Initiated Algorithm: Symmetrically initiated algorithms basically work on both the above algorithm and combine the advantages of both sender and receiver initiated algorithms to improve the efficiency and response time for the jobs. This algorithm can perform the symmetrically and generally works better in almost all cases when the queue size exceeds or bellow some threshold value.

Central Scheduler Algorithm: Central scheduler was an effective algorithm that can handle all load-balancing decisions with minimal inter processor communication. Each processor can send tasks to any of its neighboring processors without having any prior knowledge of the system .There are several heuristics algorithms have been proposed to minimize the total completion time of the tasks in grid systems.

6. Proposed Methodology

Load balancing policy which works in Grid System to balance the work load on different clusters distributed in Grid. This Algorithm first search that clusters which average load of resources be minimum and then to that resource which will be Idle or low loaded resources. The workload at each resource can be determined as: CPU utilization, CPU speed, expected completion time and queue length. There are number of clusters present in the Grid System. And in each clusters there are numbers of resources are present which may be different in each clusters.

7. Experiment and Result

GridSim toolkit

The GridSim toolkit used as the simulation for the java-based discrete-event grid simulation toolkit. It allows simulation and modeling in distributed heterogeneous and parallel computing system for the user application and

resource for evaluation for scheduling algorithm. It can also be used for the modeling and simulation of application scheduling on various classes of parallel and distributed computing systems such as clusters, grids and P2P networks there are some reasons why the GridSim toolkit was chosen over the other toolkit to simulate and evaluate our algorithm.

- It allows modeling of heterogeneous types of resources.
- Resource capability can be defined in the form if the Million Instructions Per Second (MIPS) and Standard Performance Evaluation Corporation (SPEC) benchmark.
- There is no limit on the number of application jobs that can be submitted to a resource.
- Network speed between resources can be specified.
- It supports simulation of both static and dynamic schedulers.
- Statistics of all or selected operations can be recorded.

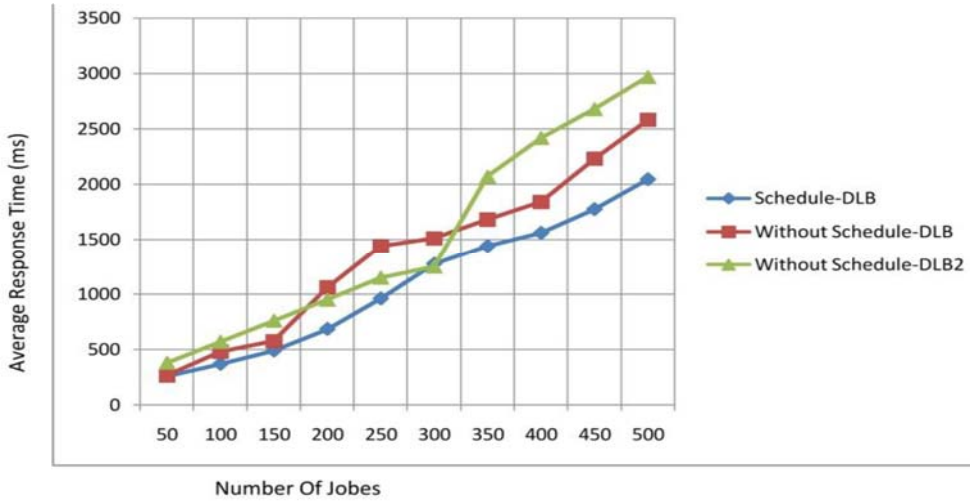

Fig 4: Analysis with other load balancing algorithm

8. Conclusion

In this study, we addressed the problem of load balancing in large scale distributed systems. We proposed a load balancing strategy based on a tree representation of a Grid. The model allows transforming any Grid architecture into a unique tree with at most four levels. From this generic tree, we can derive three sub-models depending on the elements that compose a Grid. Using this model, we defined a hierarchical load balancing strategy that privileges local balancing in first (load balance within sites without communication between sites). The first results of our experimentations are very promising and lead to a better load balancing between CE's of a Grid without high computing overhead. We have appreciably improved the metrics defined, in particular average response time.

This experimental result also shows that the proposed algorithm enhanced the load balancing and for future work.

It can introduce more effectively with resource utilization using better scheduling approach.

References

1. Albert Y. Zomaya, Yee-Hwei Teh, "Observation on Using Genetic Algorithms for Dynamic Load Balancing", IEEE, Volume 12, Issue 9, September 2001.
2. Roy D. Williams, "Performance of Dynamic Load Balancing Algorithms for Unstructured Mesh Calculations", Concurrent Supercomputing facilities, June 1990
3. Buyya, R., D.Abramson, J. Giddy and H.Stockinger, 2002. Economic models for resourcemanagement and scheduling in grid computing. J.Concurrency and Computation: Practice and Experience, 14: 1507-1542.
4. Foster, I., C. Kesselman and S. Tuecke, 2002. Theanatomy of the Grid: Enabling scalable virtualorganizations. Intl. J. High PerformanceComputing Applications, 15: 3.
5. Chervenak, A., I. Fosterand C. Kesselman, C.Salisbury and S. Tuecke, 2000. The data Grid:towards an architecture for the distributedmanagement and analysis of large scientificdatasets. J. Network and Computer Applications,23: 187-200.
6. Xu, C.Z. and F.C.M. Lau, 1997. Load Balancing in Parallel Computers: Theory and Practice. Kluwer,Boston, MA.
7. Fahd Alharbi, "Simple Scheduling Algorithm with Load Balancing for Grid Computing", Asian Transactions on Computers, Volume 2, Issue 2,May 2012