Research Inventy: International Journal Of Engineering And Science
Vol.2, Issue 10 (April 2013), Pp 53-57
Issn(e): 2278-4721, Issn(p):2319-6483, Www.Researchinventy.Com

Fair Scheduling Algorithm with Dynamic Load Balancing Using In Grid Computing

[1] Mr.V. P. Narkhede, [2] Prof. S. T. Khandare
[1]*Lecturer , Department Of IT, Anuradha Engineering College, Chikhli, India*
[2]*Associate Professor, Department Of CSE, B.N.C.O.E., Pusad, India*

Abstract : *Grid computing is new emerging technology which can be used to increase the performance of Distributed Computing. It has emerged a new technology focusing on the resource sharing, utilizing parallelism, and exploiting throughput managing and to reduce response time through proper distribution of the application. Grid computing is a replica of distributed computing that uses geographically and disperses resources. To increase performance and efficiency, the Grid system needs competent load balancing algorithms for the distribution of tasks. The main goal of load balancing is to provide a distributed, low cost scheme that balances the load across all the processors. In this seminar, the algorithm describes multiple aspects of load balancing algorithm and introduced number of concepts which explains its broad capabilities. It also fulfils the objectives of the grid environment to achieve high performance computing by optimal usage of geographically distributed and heterogeneous resources.*

Keywords: *Grid Computing, Reliability, Resources etc.*

I. INTRODUCTION

The rapid development in computing resources has enhanced the performance of computers and reduced their costs. This availability of low cost powerful computers coupled with the popularity of the Internet and high-speed networks has led the computing environment to be mapped from distributed to Grid environments. In fact, recent researches on computing architectures are allowed the emergence of a new computing paradigm known as Grid computing. Grid is a type of distributed system which supports the sharing and coordinated use of geographically distributed and multi owner Resources , independently from their physical type and location, in dynamic virtual organizations that share the same goal of solving large-scale applications. In order to fulfil the user expectations in terms of performance and efficiency, the Grid system needs efficient load balancing algorithms for the distribution of tasks. A load balancing algorithm attempts to improve the response time of user's submitted applications by ensuring maximal utilization of available resources. The main goal is to prevent, if possible, the condition where some processors are overloaded with a set of tasks while others are lightly loaded or even idle[1,4].

Grid computing, individual users can retrieve computers and data, transparently, without taking into account the location, operating system, account administration, and other details. In Grid computing, the details are abstracted, and the resources are virtualized. Grid Computing should enable the job in question to be run on an idle machine elsewhere on the network. Grids functionally bring together globally distributed computers and information systems for creating a universal source of computing power and information. A key characteristic of Grids is that resources (e.g., CPU cycles and network capacities) are shared among various applications, and therefore, the amount of resources available to any given application highly fluctuates over time. Load balancing is a technique to enhance resources, utilizing parallelism, exploiting throughput improvisation, and to reduce response time through an appropriate distribution of the application . Load balancing algorithm are two type static and dynamic, Static load balancing algorithms allocate the tasks of a parallel program to workstations based on either the load at the time nodes are allocated to some task, or based on an average load of our workstation cluster. The decisions related to load balance are made at compile time[2,3].

II. DYNAMIC LOAD BALANCING

Load balancing is a technique to enhance resources, utilizing parallelism, exploiting throughput improvisation, and to reduce response time through an appropriate distribution of the application. Load balancing algorithms can be defined by their implementation of the following policies [1]:

- **Information policy**: It states the workload of task information to be collected, when it is to be collected and from where.

- **Triggering policy:** It determines the appropriate period to start a load balancing operation.
- **Resource type policy**: It orders a resource as server or receiver of tasks according to its availability status.
- **Location policy**: It uses the results of the resource type policy to find a suitable partner for a server or receiver.
- **Selection policy**: defines the tasks that should be migrated from overloaded resources (source) to most idle resources (receiver).

Load balancing algorithms are defined by two types such as static and dynamic load balancing algorithms to allocate the tasks of a parallel program to workstations. Multicomputer with dynamic load balancing allocate or reallocate resources at runtime based on task information, which may determine when and whose tasks can be migrated. In this seminar Dynamic Load Balancing Algorithm is implemented to multicomputer based on resource type policy [1]. Load balancing feature can prove invaluable for handling occasional peak loads of activity in parts of a larger organization. These are important issues in Load Balancing [1,4]:

- An unexpected peak can be routed to relatively idle machines in the Grid.
- If the Grid is already fully utilized, the lowest priority work being performed on the Grid can be temporarily suspended or even cancelled and performed again later to make room for the higher priority work.

Load balancing should take place when the scheduler schedules the task to all processors. There are some particular activities which change the load configuration in Grid environment. The activities can be categorized as following:

- Arrival of any new job and queuing of that job to any particular node.
- Scheduler schedules the job to particular processor.
- Reschedule the jobs if load is not balanced
- Allocate the job to processor when it's free.
- Release the processor after it complete the whole job

III. LOAD BALANCING APPROACHES

Load balancing problem has been discussed in traditional distributed systems literature for more than two decades. Various algorithms, strategies and policies have been proposed, implemented and classified. Algorithms can be classified into two categories: static or dynamic.

3.1 Static Load Balancing Algorithm

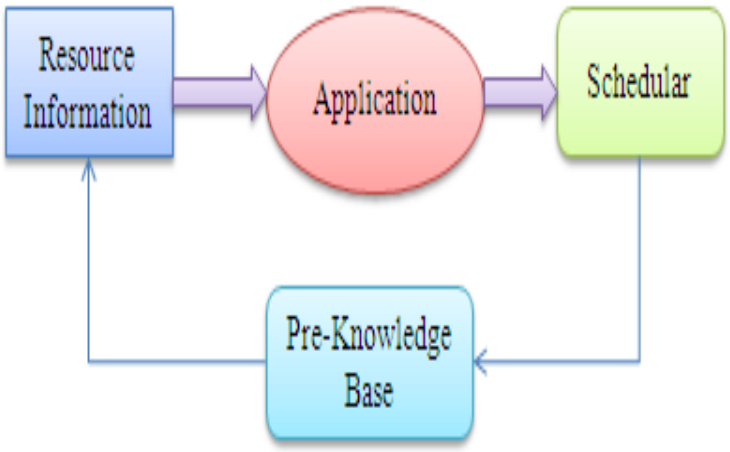

Fig.3.2: Static Load Balancing

Static load balancing algorithms allocate the tasks of a parallel program to workstations based on either the load at the time nodes are allocated to some task, or based on an average load of our workstation cluster. The decisions related to load balance are made at compile time when resource requirements are estimated. The advantage in this sort of algorithm is the simplicity in terms of both implementation as well as overhead, since there is no need to constantly monitor the workstations for performance statistics. However, static algorithms only work well when there is not much variation in the load on the workstations. Clearly, static load balancing algorithms aren't well suited to a Grid environment, where loads may vary significantly at various times [1,2,4].

3.2 Dynamic Load Balancing Algorithm

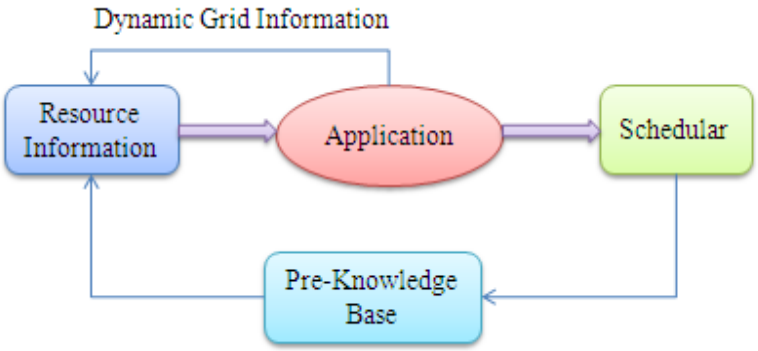

Fig.3.3: Dynamic Load Balancing

Dynamic load balancing algorithms make changes to the distribution of work among workstations at run-time; they use current or recent load information when making distribution decisions. Multicomputers with dynamic load balancing allocate/reallocate resources at runtime based on no a priori task information, which may determine when and whose tasks can be migrated. As a result, dynamic load balancing algorithms can provide a significant improvement in performance over static algorithms. However, this comes at the additional cost of collecting and maintaining load information, so it is important to keep these overheads within reasonable limits [1,2,4].

IV. SCHEDULING AND LOAD BALANCING

4.1 Fair Scheduling

The scheduling algorithms do not adequately address congestion, and they do not take fairness considerations into account. Fairness is most essential for scheduling of task. In Fair Scheduling, the tasks are allocated to multiple processors so that the task with unsatisfied demand get equal shares of time is as follows :

- Tasks are queued for scheduling according to their fair completion times.
- The fair completion time of a task is estimated by its fair task rates using a max min fair sharing algorithm.
- The tasks are assigned to processor by increasing order of fair completion time.

In this algorithm, tasks with a higher order are completed first which means that tasks are taken a higher priority than the others which leads to starvation that increases the completion time of tasks and load balance is not guaranteed. For this issue we propose a Load Balance (LB) Algorithm to give uniform load to the resources so that all tasks are fairly allocated to processor based on balanced fair rates. The main objective of this algorithm is to reduce the overall make span [1,5,6].

4.2 Segment Of Code related to Algorithm

Input: A set of N task and M number of processor with computational capacity C_j
Output: A Schedule of N tasks

1. Create a set of queues
2. qsize < N/M
3. For each queue q_i in Q
4. While there are tasks in the queue do,
5. Assign demand rate of the task X_i
6. k=C/N
7. If X_i <k
8. Assign X_i to i^{th} task as fair rate
9. Else
10. Assign k to i^{th} task as fair rate
11. Calculate fair completion time $t_i(x)$
12. End while
13. End loop
14. Arrange the task in increasing order based on their $t_i(x)$ and submitted to processor
15. While (Load of any processor is greater than average load processor)do
16. Calculate mean waiting time each scheduled task

17. If $Z_x^y > 0$
18. Migrated tasks are determined by using criteria of processor capacity
19. Each processor which has least capacity is selected for migration
20. End If
21. End While

V. RESULTS AND DISCUSSION

When we started a work on our project, our main aim was to evaluate performance of various resources with various scenarios, the concept lies when evaluating performance is to have various scheduling algorithm and various resources with different characteristics. As the grid has become very popular in its short period of its emergence, the usage of performing scheduling operation is not up to the mark because the scheduling process exists as of now is only minimal i.e., when anyone trying to perform a operation he should make the system or scheduler to know the jobs which it has to be scheduled in advance. So this is not a permanent solution in this kind of environment. After having done a extreme literature survey we have started working on evaluating performance of the system with various scheduling algorithms such as FCFS, Earliest Deadline First, Easy Back filling and so on., our work is not only performing scheduling process though evaluating the performance is the major issue in our project but before that we have found a way to create a Jobs and Resource, after creating Jobs and resource with various characteristics, these has been used as the input for our processing. Finally each and every jobs has been given to the resources in order of not keeping any of machines idle i.e., jobs will be allocated to the machines based on their availability.

The GridSim toolkit provides a comprehensive facility for simulation of different classes of heterogeneous resources, users, applications, resource brokers, and schedulers. It can be used to simulate application schedulers for single or multiple administrative domains distributed computing systems such as clusters and Grids. Application schedulers in the Grid environment, called resource brokers, perform resource discovery, selection, and aggregation of a diverse set of distributed resources for an individual user. This means that each user has his or her own private resource broker and hence it can be targeted to optimize for the requirements and objectives of its owner. In contrast, schedulers, managing resources such as clusters in a single administrative domain, have complete control over the policy used for allocation of resources. This means that all users need to submit their jobs to the central scheduler, which can be targeted to perform global optimization such as higher system utilization and overall user satisfaction depending on resource allocation policy or optimize for high priority users.

VI. CONCLUSION

This algorithm has proved the best results in terms of makespan and Execution Cost In particular the algorithm allocates the task to the available processors so that all requesting task get equal amount of time that satisfied their demand. Through this proposed algorithm, we have described multiple aspects of load balancing algorithm and introduced numerous concepts which illustrate its broad capabilities. Proposed algorithm is definitely a promising tendency to solve high demanding applications and all kinds of problems. Objective of the grid environment is to achieve high performance computing by optimal usage of geographically distributed and But grid application performance remains a challenge in dynamic grid environment. Resources can be submitted to Grid and can be withdrawn from Grid at any moment.

VII. ACKNOWLWDGMENT

I am highly grateful to Prof. S. T. Khandare , for his sincere advice, encouragement and continuous guidance in my work. I warmly acknowledge and express my special thanks for him inspiring discussions and infallible suggestions.

REFERENCES

[1] U. Karthick Kumar, "A Dynamic Load Balancing Algorithm in Computational Grid Using Fair Scheduling, *International Journal Of Computer Science and Informatics Vol. 8, Issue 5, No 1,* September 2011 ISSN (Online): 1694-0814.

[2] Dr. K. Vivekanandan, D.Ramyachitra, "A Study on Scheduling in Grid Environment", *International Journal on Computer Science and Engineering (IJCSE), ISSN : 0975-3397 Vol. 3 No. 2* Feb 2011.

[3] Sachin Kumar, Niraj Singhal, " A Priority based Dynamic Load Balancing Approach in a Grid based Distributed Computing Network", *International Journal of Computer Applications (0975 – 8887) Volume 49– No.5,* July 2012 11

[4] Pawandeep Kaur, Harshpreet Singh, "Adaptive dynamic load balancing in grid computing an approach,"*International journal of engineering science & advanced technology , ISSN: 2250–3676 Volume-2, Issue-3, 625 – 632,* May-Jun 2012.

[5] R.Gogulan, A.Kavitha, U.Karthick Kumar, "Max Min fair scheduling algorithm using in grid scheduling with load balancing", *International Journal of Research in Computer Science eISSN 2249-8265 Volume 2 Issue 3* (2012) *pp. 41-49.*

[6] Prabhat Kr. Srivastava, "Improving Performance in Load Balancing Problem on the Grid Computing System", *International Journal of Computer Applications (0975 – 8887) Volume 16– No.1,* February 2011

[7] Qin Zheng Chen-Khong Tham Bharadwaj Veeravalli, "Dynamic Load Balancing and Pricing Dynamic Load Balancing and Pricing in Grid Computing with Communication Delay, November 2, 2010

[8] Belabbas Yagoubi and Yahya Slimani, "Dynamic Load Balancing Strategy for Grid Computing," *Journal of Information Technology and Applications Vol. 1 No. 4* Mar ch,2007, pp. 285-296.

[9] Belabbas Yagoubi and Yahya Slimani, "Dynamic Load Balancing Strategy for Grid Computing," World Academy of Science, Engineering and Technology 19 2006.

[10] A book ,"*An introduction to grid computing*", By Bart Jacob, Michael Brown, Kentaro Fukui, Nihar Trivedi, IBM/Redbooks December 2005.