A Priority based Dynamic Load Balancing Approach in a Grid based Distributed Computing Network

Sachin Kumar
Computer Science Department
Guru Nanak Education Trust's Group of Institutions
Roorkee, Uttarakhand (India)

Niraj Singhal
Faculty of Electronics, Informatics and Computer Engineering
Shobhit University, Meerut, Uttar Pradesh (India)

ABSTRACT

Load balancing in grid based distributed computing environment increases the availability and scalability of entire system. Dynamic load balancing has the potential to perform better than static load balancing, but they are inevitably more complex. The overhead involved is much more but one can not negate their benefits. Load balancing strategies try to ensure that every processor in the system performs almost the same amount of work at any point of time. Process migration is one of the important tasks in dynamic load balancing which usually deals with the migration of task from overloaded computing nodes to under-loaded nodes. Although numerous works has been done on the issue of process migration and load balancing. Communication overhead is still a problem which is to be reduced in grid based networks. In this paper, we propose an algorithm that finds an under-loaded node whenever an overloaded node is found, and also takes the idea of assigning a priority to each computing node in the grid system based on their computing power. The proposed algorithm reduces the communication overhead and proves to be cost effective.

Keywords

Distributed Computing, Grid, Load Balancing, Task Migration, Communication Overhead.

1. INTRODUCTION

In a grid based distributed computing environment, where several autonomous systems are interconnected, equal load distribution is main concern. The traffic on particular machine could be large at a moment while on the other end, it could be negligible. A high degree of parallelism can be achieved if different tasks run on different machines simultaneously. In order to increase the performance of distributed computing environment, various researchers have proposed many schemes for load balancing. These include process, data or thread migration. Various schemes have been designed to support process migration. Although, the basis for designing all these systems have been purely problem driven; some of these systems are functional but others are purely hypothetical [2].

Load balancing based on the idea of migration of excess load from heavily loaded nodes to lightly loaded nodes. The problem starts with determining when to migrate a process or task. This solution is typically based on local load situation: for example, a simple procedure may be the comparison of the load between various nodes and a determination of the node to which the task is to be migrated. Static algorithms collect no information and make probabilistic balancing decisions, while dynamic algorithms collect varying amounts of state information to make their decisions. The most significant parameter of the system was found to be the cost of transferring a job from one node to another. It is the cost that limits the dynamic algorithms, but at the high end of complexity are the dynamic algorithms which do collect varying amount of information. Potentially, the more information an algorithm can collect, the better decision it will make. The problem with the complex balancing algorithms is that they cannot keep up with the rapidly changing state information of the system. Two broad categories of load balancing are commonly recognized. In source-initiative algorithms, the host where jobs arrive takes the initiative to transfer the jobs, whereas in receiver-initiative algorithms, the hosts are able and willing to receive transferred jobs.

In this paper, we propose an approach which provides a lightly loaded node whenever an over loaded node is found. To process the large parallel tasks, it is decomposed into a number of tasks to distribute them among processors and then computed outputs are gathered to produce final results. In a grid based network, an equal distribution of tasks to the processors may result the fast processors to finish their work earlier than slow processors, due to heterogeneity of processors in the network. In this situation, some processors are idle while others suffer from overload of work [2]. If we distribute the work load among the processors a according to their efficiency then we can achieve better response time. To check the efficiency of processors, a sequential program is run on all the processors and time taken to finish that work by all processors is collected and used to assign an assignment factor and priority to all the processors.

The organization of the paper is as follows: In section II, the related work on load balancing is discussed. Section III represents the proposed work. Finally, we conclude in section IV.

2. RELATED WORK

Various scheduling algorithms have been proposed for parallel and distributed systems, as well as for Grid computing environment. For a dynamic load balancing algorithm, it is not acceptable to frequently change state information because of the high communication overheads. In [3] an estimated load information scheduling algorithm (ELISA) and Perfect Information Algorithm (PIA) is proposed. In PIA, when a job arrives, a processor computes the job's finish time on all buddy processors using exact information about the current load of a buddy processor, its arrival rate and service rate. The source processor selects a buddy processor with the minimum finish time and immediately migrate a job on that buddy processor, if it can finish the job earlier than this processor. In the decentralized load balancing algorithm proposed in [5] for a Grid environment. Although this work attempts to include the communication latency between two nodes during the triggering process on their model, it did not consider the actual cost for a job transfer. In [6, 7], a sender processor collects status information about neighbouring processors by communicating every load balancing moment. This can lead to frequent message transfers that results in large communication overhead which is undesirable. Preemptive and non-preemptive process migration techniques are proposed in [4]. In this process migration takes place efficiently by considering memory usage and load on processor.

3. PROPOSED WORK

In a grid based distributed computing environment where heterogeneity exists among processors, it is necessary to prevent the processors from suffering of overload of work and also to minimize the idle time of processors. There should be such distribution of work that all processors will finish their work at almost same time. This is an idle case, but the variation in the finishing time of all the processors can be minimized by assigning them work proportional to their computing power. The priority of each node can be assigned in advance by running a sequential program on each processor and maintaining the estimated time taken by the processors to complete that program. An example of priority assignment for total 'N' nodes is shown in Table 1.

Table 1: Priority assignment based on time taken to complete the sequential program by different processors

Node No	Estimated Time Taken to Complete Sequential Program	Priority
P1	8.4 ms	3
P2	20 ms	9
P3	9.4 ms	4
P4	10 ms	5
P5	16.6 ms	8
P6	9.4 ms	4
P7	11.1 ms	6
P8	6.6 ms	1
P9	14.2 ms	7
P10	7.1 ms	2

The nodes with high computing power allotted the high priority. Nodes with equal completion time get equal priority. If we distribute the tasks to nodes according to their priority then there will be least chances that a node will be overloaded. If still a node found overloaded the proposed algorithm will find a suitable node which is under loaded by looking first at the high priority node, if this node is not idle or heavily loaded then we keep on looking for other lightly loaded node in decreasing order of their priority and the task migration will take place. Using this approach, faster nodes make system balanced as early as possible.

In our proposed algorithm, a migrating server node (MSN) returns light weighted node whenever required. It is done by checking the status of all the nodes which are under-loaded. When a node is overloaded, it calls the MSN which then finds a suitable node and then performs the load balancing. Our approach is based on the principle that in a distributed environment at least one node should be there which is lightly loaded. CPU queue length is considered.

Algorithm:

```
Ni: List of computing nodes in decreasing order of priority
MSN ( ): MSN function which returns a lightly loaded node
every time a heavily loaded node found
Initialization:    Assume that each node is having some
                   load
                    Ni ← Load;  /*Load defines at least one
                   process is there*/

Procedure: Main ( )
        {
             Call MSN ( ) /* MSN will search a light
             weighted node
             Let Nt is overloaded node with load T
             Begin
               Available_Node ← MSN ( );
               Migrate_load (Nt, Available_Node)
             End
        }

Procedure: MSN ( )
    {
          Let E be the threshold level of CPU queue length of
          a node below which node is considered as lightly
          loaded.
          For j=1 to n do      /*n be the total number of
          computing nodes in a cluster*/
          If(CPU_queue_length(Nj) < E)
          {
                 Return Nj;
          }

    }

Procedure: Migrate_load(overloaded_node, under-loaded
node)
    {
          Under-loaded node = T; /*overload is  assigned to
          available node given by MSN ( )*/
          Load [Ni] = Load [Ni] – T; /*overload is reduced
          from the overloaded node*/
     }
```

The existing load balancing algorithms usually find the under-loaded node by their status information exchange between the nodes. In the proposed algorithm, we have a function called msn ( ) which finds the available under-loaded nodes by looking into a queue where all the processors are scheduled in the decreasing order of their computing power. Hence, the probability that first node in the queue will be the under-loaded node is high as the first node is having the highest computing power and it may assume that it has finished the assigned work and can be idle that time, if not msn ( ) will check the second node having second highest computing power and so on. Hence, it reduces the communication overhead as compare to other existing algorithms in which it is necessary to collect the status information of all the nodes to find out which node is under-loaded.

4. CONCLUSION AND FUTURE WORK

The main goal in distributed system is to execute the process at minimum cost i.e. time is most important factor that can be considered in cost estimation. This is attributable to the fact that new dynamic load balancing policy achieves a higher success, in comparison to the previously used load balancing techniques, in reducing the likelihood of nodes being idle while there are tasks in the system [16]. Although the main objective is to propose load balancing algorithms using parameter estimation for heterogeneous grid environments, this work can be extended by providing fault tolerance into the system as fault tolerance is a very important characteristic for any distributed system. Our future work considers the implementation and evaluation of the complexity of the proposed approach for load balancing.

5. REFERENCES

[1] Livny M. and M. Melman, “Load Balancing in Homogeneous Distributed Systems”, Proc. ACM Computer Network Performance Symp., vol 11, 1982.

[2] S. Sharma, S. Singh and M. Sharma, “Performance Analysis of Load Balancing Algorithms”, World Academy of Science, Engineering and Technology, vol 38, 2008

[3] Linda F. Wilson and Wei Shen, “Experiments in Load Migration and Dynamic Load Balancing in SPEEDS”, Proceedings of the Winter Simulation Conference, 1998.

[4] L. Anand, D. Ghose and V. Mani, “ELISA: An Estimated Load Information Scheduling Algorithm for Distributed Computing Systems”, International Journal of Computer and Mathematics with Applications, April 1999.

[5] P. Kanungo and M. Chandwani, “A Process Migration Methodology for Distributed Computing Environment”, Indian Journal of Computing Technology, May 2006.

[6] M. Arora, S.K. Das and R. Biswas, “A Decentralized Scheduling and Load Balancing Algorithm for Heterogeneous Grid Environment”, Proceedings International Conference of Parallel Processing Workshops (ICPPW ’02).

[7] H. Shan, L. Oliker and R. Biswas, “Job Super Scheduler Architecture and Performance in Computational Grid Environments”, Proceedings ACM/IEEE Conference of Super Computing, Nov. 2003.

[8] L. Oliker, R. Biswas, H. Shan and W. Smith, “Job Scheduling in Heterogeneous grid Environment”, Technical Report LBNL-54906, Lawrence Berkeley National Laboratory, 2004.

[9] Yagoubi B. and Y. Slimani, “Dynamic Load Balancing Strategy for Grid Computing”, Proceedings of World Academy of Science, Engineering and Technology, May 2006.

[10] Sachin Kumar and Niraj Singhal, “A Study on the Assessment of Load Balancing Algorithms in Grid Based Network”, International Journal of Soft Computing and Engineering, March 2012.

[11] N. G. Shivratri, P. Krueger, and M. Singhal, “Load Distributing for Locally Distributed Systems”, Computer, Vol. 25, 1992.

[12] Ali M. Alakeel, “A Guide to Dynamic Load Balancing in Distributed Computing Systems”, International Journal of Computer Science and network Security, June 2010.

[13] J. Lee, P. Keleher and A. Sussman; “Decentralized Dynamic Scheduling across Heterogeneous Multi-core Desktop Grids”, IEEE, May 2010.

[14] T. Amudha, T. T. Dhivyaprabha; “QoS Priority Based Scheduling Algorithm and Proposed Framework for Task Scheduling in a Grid Environment”, IEEE International Conference on Recent Trends in Information Technology, MIT, Anna University, Chennai, June 2011.

[15] Sameer Singh Chauhan, R. C. Joshi, “QoS Guided Heuristic Algorithm for Grid Task Scheduling”, International Journal of Computer Applications, June 2010.

[16] E. Saravanakumar and P. Gomathy, “A Novel Load Balancing Algorithm for Computational Grid”, International Journal of Computational Intelligence Techniques, Vol. 1, No. 1, 2010.

[17] Said Fathy El-Zoghdy, “A Capacity Based Load Balancing and Job Migration Algorithm For Heterogeneous Computational Grids”, International Journal of Computer Networks & Communication (IJCNC) Vol. 4, No. 1, January 2012.