International Journal of Computer Applications (0975 – 8887)
National Conference on Trends in Advanced Computing & Information Technology-2016

Dynamic Load Balancing in Grid Computational Environment using Ant Algorithm

Sugandha Satija
Information Technology
KITS,Ramtek
Nagpur,India

ABSTRACT

Load Balancing is one of the major issues in computational Grids. Research has proved that load balancing on Grid Computational Environment is best solved by Heuristic approach. The main motive behind load balancing is to equally spread the load on each node of the Grid. In this paper, ASRank (Rank based Ant system) is proposed to provide shortest path from PE (Processing element) to RN (Resource node) while balancing the load on each RN. ASRank will determine the best resource to be allocated to the jobs, based on their paths as well as their load. ASRank reads the pheromone value, such that solutions with shorter paths will have higher pheromone value. This pheromone value is used by other PE's with the help of a process named as Stigmergy, to reach to the RN. This will maximize the efficiency of the Grid and will result in high throughput. Thus, it increases the performance in the Grid Computational Environment.

General Terms

Grid Computing, Ant Algorithm.

Keywords

Grid Computing, Load Balancing, ASRank ANT algorithm, Processing Element, Resource Node, Stigmergy, Pheromone Value

1. INTRODUCTION

In large scale computing systems such as Grid Computing, there are often large amounts of resources available to be used for computing jobs[3]. Grid Computing is the technique which uses the resources scattered across the world, connected through a network, for solving a computing problem. There are two main components in a Grid: PE (Processing Element) and a RN (Resource Node). A PE is the one which has a computing job but don't have enough resources to complete its job. The reasons behind this can be many but the main one is the Cost. So, Grid computing makes it possible to share and use resources like supercomputers, databases, data sources and specialized devices which are termed as RN. But, this comes with a major responsibility i.e. scheduling the jobs according to the load on RN's. Load balancing aims upon equal distribution of the load of different PE's on RN's. Thus, maximizing the resource utilization and minimizing the execution time [1]. It basically depends upon two conditions: Static or Dynamic

Static Load Balancing: In static load balancing, load is distributed according to a default policy, which remains fixed and does not change during the implementation. It uses the basic information i.e. characteristics of jobs, RN's, PE's, communication network and design a policy according to that and follow it throughout the implementation.

Dynamic Load Balancing: A dynamic load balancing algorithm attempts to use the runtime information to make the decisions depending upon the changing state. It quickly adjusts with workload **fluctuations** [1]

2. ANT ALGORITHMS IN GRID

In the natural world, ants roam randomly in the search of food and upon finding food return to their colony while laying down a chemical substance known as pheromone on the path. If other ants find such a path, they are likely not to keep travelling at random, but instead they will follow the trail. If they also found the food on reaching the destination then they will return on the same path and in return will reinforce the exciting pheromone. Over time, the path reaching to the food will have the highest pheromone and thus, will be selected again and again. So, pheromone density keeps on increasing.

But, the pheromone also evaporates over time thus reducing its density. A short path, by comparison, gets marched over more frequently and it results in higher pheromone density. In Ant algorithms, we assume an artificial ant which can simulate the pheromones and record the nodes that it passes. In future, that information is used by other ants to know the shortest path by a technique known as Stigmergy. Stigmergy is an indirect communication between two objects. Out of these one has performed any action which has stimulated the action by second object. The principle is that the trace left in the environment by an action stimulates the performance of a next action, by the same or a different agent. So, two ants communicate with each other by this technique. There are various types of Ant algorithms available for grid such as Elitist ant system (EAS), Max-Min ant system (MMAS), Ant colony system (ACS), Rank based ant system (ASRank), Continuous orthogonal ant colony (COAC) and Recursive Ant colony optimization (RACO). EAS gives us the best solution by depositing pheromone during all iterations. MMAS adds maximum and minimum pheromone value. ACS has been applied in solving many problems such as travelling Salesman problem (TSP), open shop problem etc. It is used in grid because it can be used for both static and dynamic conditions. In ASRank, all solutions are ranked according to their length. The amount of pheromone deposited is checked for each path, such that shorter path will deposit more pheromone than longer paths. COAC enables ants to search for the solution effectively by providing an orthogonal path to explore. RACO is a recursive form of Ant system which divides the whole problem into sub-problems and then solves it. The results from all the sub-problems are compared and the best of them is selected.

3. RELATED WORK

In past, a lot of work has been done to effectively balance the load in a grid computational environment. Generally, there are two methods for performing load balancing in Grid computing: static load balancing and dynamic load balancing

[1]. In some algorithms, the combination of these two methods are used which are referred as combined algorithms. Ant Colony Optimization (ACO) is a meta-heuristic using artificial ant to find desirable solutions to difficult combinatorial optimization problems. The behaviour of artificial ants is based on the traits of real ants [2] as described above plus additional capabilities that make them more effective. Each ant of the "colony" builds a solution to the problem under consideration and uses information collected on the problem characteristics and its own performance to change how other ants see the problem [4]. In [2], many aspects of the social insects are told, such as, self-organizing. This means that complex group behaviour emerges from the interactions of individuals who exhibit simple behaviours by themselves. Examples of these collective activities among ants are finding food and building nests. To achieve this, self-organization relies on several components: (i) positive feedback (ii) negative feedback (iii) multiple interactions. There are often large amounts of resources available to be used for computing jobs. Since these resources can cost up to millions, maximizing their utilization is an important factor. Load balancing is used for this purpose [3]. For a dynamic load balancing algorithm, it is not acceptable to frequently change state information because of the high communication overheads. In [14] an Estimated Load Information Scheduling Algorithm (ELISA) and Perfect Information Algorithm (PIA) is proposed. In PIA, when a job arrives, a processor computes the jobs 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. Load balancing algorithms can be defined by their implementation of the following policies [15]

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).

4. PROPOSED ALGORITHM

In proposed Ant algorithm, the pheromone is associated with the shortest path from PE's to RN's, rather than resources. The increase and decrease of pheromone represent the path changing in dynamic time. The resource allocation is done according to the previous load on a particular node. The parameter used for this purpose is QueueLength. Maximum value for QueueLength is defined. After that value no load will be assigned to that node. Thus, it checks two parameters to do load balancing: first, the path from PE to RN and second, the queue length of a particular RN. 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 is free.

• Release the processor after it completes the whole job.

When a resource enters into a Grid Computational Environment, it is asked to specify its performance parameters, such as the number of processors, processing capabilities etc. then the following procedure can be used:

4.1 ASRANK ALGORITHM

1. Input the value of number of PE's and RN's.
2. Get the availability of all registered resources.
3. Initialize nextUser = 0 and QueueLength of each resource=0
4. Initialize MaxQueueLength=1
5. Find the resource R with the shortest path from the current PE using ASRank algorithm.
6. Check the QueueLength of R.
7. If QueueLength(R)< MaxQueueLength
8. Allocate the job of nextUser to R.
9. Set nextUser=nextUser+1.
10. QueueLength_R=QueuLength_R+1.
11. When task is completed and there are no there are no more requests from nextUser then QueueLength_R = QueueLength_R-1.

Thus, initialize load balancing by first getting the total number of PE's and RN's available in the network. After that, check the available resources in the network along with their characteristics. Once this is done, start with initialization of the parameters for the current task. Then check for the shortest path from the task to the resource. For this purpose pheromone value at a particular path is checked. Once that is found check its load queue, if it is filled to the maximum find the next shortest path. Else, allocate that resource for that task and increase its queue length by one. This procedure is repeated for all the tasks. The initial condition is as shown:

Table 1. Initial Status

NODE	RANK	Queue_Length
1	1	0
2	5	0
3	2	0
4	4	0
5	3	0

As it has the MaxQueueLength as One so, only 1 node will be allocated to a particular resource. If in the mean time, new request comes then it is allocated to the next RN in the rank order. Let two more requests come, then, how the resource is allocated depends upon the rank. As, first rank is already having QueueLength one, so, next node in the rank order is determined and then its QueueLength is checked. The three iterations are as Shown:

Table 2. First Iteration

NODE	RANK	Queue_Length
1	1	1
2	5	0
3	2	0
4	4	0
5	3	0

Table 3. Second Iteration

NODE	RANK	Queue_Length
1	1	1
2	5	0
3	2	1
4	4	0
5	3	0

Table 4. Third Iteration

NODE	RANK	Queue_Length
1	1	1
2	5	0
3	2	1
4	4	0
5	3	1

5. CONCLUSION AND FUTURE WORK

The proposed algorithm is expected to determine the best resource to be allocated to the jobs, based on the shortest path and the resource capacity and at the same time to balance the load in grid. The algorithm considers various resource parameters for calculating pheromone and QueueLength. The algorithm is expected to achieve better throughput and hence increase overall performance in grid environment. In future, the algorithm can be implemented using GridSim simulator.

6. ACKNOWLEDGMENTS

[1] Sowmya Suryadevera, Jaishri Chourasia, Sonam Rathore and Abdul Jhummarwala “Load Balancing in Computational Grids Using Ant Colony Optimization Algorithm “International Journal of Computer and Communication Technology ISSS(online): 2231-0371 vol-3,iss- 3,2012.

[2] Sandip Kumar Goyal and Manpreet Singh “adaptive and Dynamic load balancing in Grid using Ant Colony Optimization” International Journal of Engineering and Technology(IJET).

[3] D. Maruthanayagam and Dr. R. Uma Rani “Enhanced Ant colony algorithm for Grid Scheduling” International Journal of Computer Technology and Applications. Vol1 (1) 43-53.

[4] P. McMullen and P. Tarasewich, “Using ant techniques to solve the assembly line balancing problem,” Institute of Industrial Engineers *Trans.*, vol. 35, no. 7, pp. 605-617, 2003.

[5] Jagdish Chandra Patni, Dr. M.S.Aswal, Om Prakash Pal and Ashish Gupta, “Load balancing Strategies for Grid Computing” presented at 3rd international conference on electronics computer technology (ICECT), vol.3, pp. 239- 243, 2011.

[6] S. Fidanova and M. Durchova, "Ant algorithm for grid scheduling problem," Lecture Notes in Computer Science, vol. 3743, pp. 405- 412, 2006.

[7] A. Ali, M. A. Belal and M. B. Al-Zoubi, “Load Balancingof Distributed system Based on Multiple Ant ColoniesOptimization,”American Journal of Applied Sciences, vol.7(3), pp. 433-438, 2010.

[8] K. Sathish and A. Reddy, "Enhanced ANT Algorithm Based Load Balanced Task Scheduling in GRID Computing," IJCSNS, vol. 8, pp. 219, 2008.

[9] S. K. Goyal, R. B. Patel, and M. Singh, “Adaptive and dynamic load balancing methodologies for distributed environment: a review,” *International Journal of Engineering Science and Technology (IJEST),* vol. 3, no. 3, pp. 1835-1840, 2011.

[10] H. Yan, X. Shen, X. Li, and M. Wu, “An improved ant algorithm for job scheduling in grid computing,” in *Proc. 4th Inter. Conf. Machine Learning and Cybernetics,* 2005, pp. 2957-2961.

[11] H. J. A. Nasir, K. R. K. Mahamud, and A. M. Din, “Load balancing using enhanced ant algorithm in grid computing,” in Proc. 2nd Inter. Conf. Computational Intelligence, Modelling and Simulation, 2010, pp. 160-165.

[12] A. D. Ali, and M. A. Belal, “Multiple ant colonies optimization for load balancing in distributed systems,” in *Proc. Inter. Conf. (ICTA’07),* 2007.

[13] Dushyant Vaghela “An Advanced Approach On Load Balancing in Grid Computing”

[14] Nima Jafari Navimipour, Seyes Hasan Es-hagi, “LGR: The New Genetic Based Scheduler for Grid Computing Systems”,International Journal of Computer and Electrical Engineering, Vol. 1, No. 5, December, 2009, pp. 1793-8163.

[15] *U.Karthick Kumar “*A Dynamic Load Balancing Algorithm in Computational Grid Using Fair Scheduling” IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 5, No 1,September 2011