Various Strategies of Load Balancing Techniques and Challenges in Distributed Systems

Abhijit A. Rajguru
Research Scholar at WIT, Solapur
Maharashtra (INDIA)

Dr. Mrs. Sulabha. S. Apte
WIT, Solapur
Maharashtra (INDIA)

Abstract— A distributed system is a collection of resources interconnected by communication network to form a large loosely coupled computing system. Apart from information sharing another important use of distributed systems is sharing of the processing power among various computing nodes. This can be done by migrating a local process and executing it at a remote node of the network. When the demand for computational power increases, the load balancing problem becomes very important. The problem of task scheduling and load balancing in distributed system are most important and challenging area of research in computer science & engineering. Task Scheduling and load balancing in distributed system plays very important role in overall system performance. Load balancing is the process of improving the performance of system through a redistribution of load among processor. The main purpose of this paper is to provide a platform in design of new load balancing algorithms in future by studying existing load balancing techniques.

Index Terms— Distributed Computing, Load Balancing, Fuzzy Logic, Genetic Algorithm, Graph theory, artificial neural network.

I. INTRODUCTION

1.1 Distributed Systems

In parallel and distributed systems more than one processor executes parallel jobs. The amount of processing time needed to execute all processes assigned to a processor is called workload of a processor [1]. Load balancing involves the distribution of jobs throughout a networked computer system, thus increasing throughput without having additional or faster computer hardware. [2] Load balancing is to ensure that every processor in the system does approximately the same amount of work at any point of time. [1] An important problem here is to decide how to achieve a balance in the load distribution between processors so that the computation is completed in the minimum possible time.

II. LOAD BALANCING APPROACHES

Depending on the current state of the system, load balancing algorithms can be divided into 2 categories as static and dynamic load balancing algorithms.

2.1 Static Load Balancing Algorithm

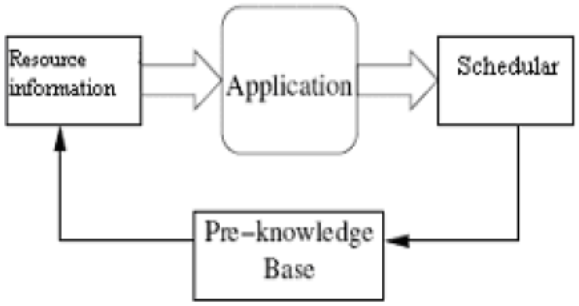

Figure 1: Static load balancing

Static load balancing policies are generally based on the information about the average behavior of system [3]. Static load balancing schemes use a priori knowledge of the applications and statistical information about the system. In static load balancing, the performance of the processors is determined at the beginning of execution. Then depending upon their performance the work load is assigned by the master processor. The slave processors calculate their allocated work and submit their result to the master. A task is always executed on the processor to which it is assigned that is static load balancing methods are non preemptive. The goal of static load balancing method is to reduce the execution time, minimizing the communication delays. A general disadvantage of static approaches is that the final selection of a host for process allocation is made when the process is created and cannot be changed during process execution to make changes in the system load. There are four types of static load balancing: - Round Robin algorithm, Randomized algorithm, Central Manager Algorithm, and Threshold algorithm.

2.2 Dynamic Load Balancing Algorithm

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 [3].

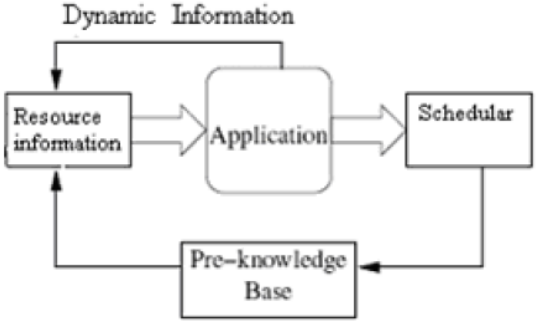

Figure 2: Dynamic load balancing

Dynamic load balancing algorithms can provide a significant improvement in performance over static algorithms. Many load balancing algorithms has been proposed in this field. It is difficult to achieve load balancing in distributed systems because of its dynamic nature. Many algorithms presented are based on centralized structure of system.

2.3 Load Balancing Strategies

There are three major parameters which usually define the strategy a specific load balancing algorithm will employ. These three parameters answer three important questions:

1) Who makes the load balancing decision?

2) What information is used to make the load balancing decision?

3) Where the load balancing decision is made.

2.4 Need for Load Balancing

Distributed system consists of number of computational resources and communication network. The tasks arrived in distributed systems are typically not uniformly distributed. Some of the node may be heavily loaded while the others may be lightly loaded. To increase the overall throughput of the system tasks of heavily loaded nodes can be transferred to lightly loaded nodes. The main aim of load balancing technique is to improve the overall performance of a distributed system, mainly in terms of resource availability or response time by distributing workload amongst the resources.

III. CHALLENGES FOR LOAD BALANCING

There are some qualitative metrics that can be improved for better load balancing in distributed system [4] [5].

1) ***Throughput:*** It is the total number of jobs that have completed execution for a given period of time. It is required to have maximum through put for better performance of the system.

2) ***Associated Overhead:*** It describes the amount of overhead during the execution of the load balancing algorithm. It consists of migration of tasks, inter process and processor communication. For effective load balancing technique, minimum overhead should be there. Static load balancing algorithm has minimum associated overhead than dynamic.

3) ***Adaptability:*** This factor is used to check whether the algorithm is adaptive to varying or changing situations i.e. situations which are of dynamic nature.
Static load balancing algorithms are not adaptive.

4) ***Fault tolerant:*** It is the ability to perform load balancing by the appropriate algorithm without any failure. Every load balancing algorithm should have good fault tolerance approach.

5) ***Migration time:*** It is the amount of time for a process to be transferred from one system to another system for execution. For better performance of the system this time should be always less.

6) ***Response time:*** It is the time taken by a particular load balancing technique to respond. This time should be minimized for better performance. Static load balancing algorithms has low response time than dynamic.

7) ***Cooperative:*** This parameter gives that whether processors share information between them in making the process allocation decision other are not during execution. A good load balancing algorithm should be cooperative.

8) ***Resource Utilization:*** It is the parameter which gives the information within which extant the resource is utilized. For efficient load balancing in system, optimum resource should be utilized. Dynamic load balancing techniques has better resource utilization than static.

9) ***Scalability:*** It is an ability of load balancing algorithm for a system with any finite number of processor or machines. This parameter can be improved for better system performance.

10) ***Performance:*** It is the overall efficiency of the system. If all the parameters are improved then the overall system performance can be improved.

IV. VARIOUS STRATEGIES OF LOAD BALANCING TECHNIQUES

4.1 Load Balancing using Fuzzy Logic

Dr. Zadeh proposed fuzzy logic concept in 1965 [6]. Fuzzy logic is a powerful tool in representing linguistic information and is very useful to solve problems that do not have a precise solution and the conventional methods cannot solve them very well. For example, computer load. However, fuzzy systems have the problem of determining its parameters. Ones of the most important parameters of fuzzy system are the Membership Functions (MF). The fuzzy inference engine in conjunction with the control rules to determine an appropriate output response then uses the value ranges.

The algorithm for the fuzzy-based load balancing has the following steps.

1) Fuzzification

Every node should evaluate its own load status. First, the workload of a node is determined. Then it is used as the input to its membership functions for finding its degree.

2) Applying the inference rules

A heavily loaded node may send a request for finding a lightly loaded node for the load transfer using the inference rules.

3) Defuzzification

For defuzzification generally centroid method is used to determine the output crisp value (the load transfer probability). Once the workloads for the sender and receiver are known and the inference rule is applied, then the output crisp value is found from the output membership functions.

4) Find the suitable node for the load transfer

The easiest way for a requesting node to find a suitable node is to choose the lightly loaded node that is the first one to reply. There might be several nodes that are lightly loaded for load transfer and the requesting node may choose the least lightly loaded node. One method is to choose the node that has the highest crisp value after the defuzzification process.

5) Transfer the loads from the sender to the receiver

4.2 Load Balancing using Graph theory.

Directed graph are useful in the context of load balancing. Nodes can represent tasks and the links representing data or communication dependencies. In load balancing using graph partition of graph is need to minimize execution time. The graph partition problem is formally defined on data represented in the form of a graph G = (V,E) with V vertices and E edges. As optimal graph partitioning is NP complete, so use heuristics.

Formal Models in Load Balancing: Task Graphs

A task graph is a directed acyclic graph where

- Nodes denote the concurrent processes in a concurrent program
- Edges between nodes represent process communications/synchronization
- Nodal weight is the computational load of the process the node represents
- Edge weight between two nodes is the amount of communication between two processes represented by the two nodes.

Formal Models in Load Balancing: Processor Graphs

- The processor graph defines the configuration of the parallel or distributed system.
- Each node represents a processor & the nodal weight is the computation speed of this processor.
- The edges between nodes represent the communication links between the processors represented by the nodes.
- Edge weight is the speed of this communications link.

4.3 Load Balancing using Fuzzy Neural Network

The main advantage of FNN [7] is its ability to combine the advantages of fuzzy system to model a problem domain using a linguistic model instead of complex mathematical model with the learning capabilities of neural network. In addition, the black box nature of neural network paradigm is resolved, as the connection structure essentially defines fuzzy rules. The combinations of neural and fuzzy systems createan adaptive system with sensory and cognitive components. When using fuzzy neural network technique with load balancing of computer network by employing its feather above can increase scalability and high performance for network. fuzzy neural network (FNN) has been proposed and successfully applied to solve these problems such as classification, identification, control, pattern recognition, and image processing, etc.

The structure of fuzzy neural network (FNN) is consists of 4 layers.

1) Input layer: It transmits the input linguistic variables X to the output without change.

2) *Fuzzification layer:* It is an membership layer represents the input values with the membership functions:

3) Product layer: It is a rule layer which implements the fuzzy inference mechanism, and each node in this layer multiplies the input signals and outputs the result of the product.

4) Normalization layer: The output (ui) of product layer is normalized in this layer through dividing its value by the summation of all the outputs of all rules.

5) *Defuzzification layer:* the summation of all normalized values (ūi) is multiplied by the corresponding weight wij which represents the consequent part of the rules, to produce the center of gravity (CoG) defuzzification operation.

4.4 Load Balancing using Ant Colony

K. Nishant et al. [8] introduced a static load balancing technique called Ant Colony Optimization. This technique uses the Ants behavior to collect information of node to assign task to the particular node. Ants move in forward direction in search of the overloaded or under loaded node. As the overloaded node is traversed, then ants move back to fill the recently encountered under loaded node, so a single table is updated every time. In this technique, a pheromone table was being designed which was updated by ants as per the resource utilization and node selection formulae.

4.5 Load Balancing using Honey Bee

Dhinesh Babu and P.Venkata Krishna [9] proposed an algorithm for Honey Bee inspired load balancing(HBB-LB) of tasks in cloud computing environments which aims to achieve well balanced load across Virtual machines for maximizing the throughput. This whole algorithm is based on the process of honeybees finding the food and alarming others to go and eat the food. First forager bees go and find their food. After coming back to their respective beehive, they dance. After seeing the strength of their dance, the scout bees follow the forager bees and get the food. The more energetic the dance is, the more food available is. So this whole process is mapped to overloaded or under loaded virtual servers . The server processes the requests of the clients which is similar to the food of the bees. As the server gets heavy or is overloaded, the bees search for another location i.e. client is moved to any other virtual server. In this way, this whole technique works.

4.6 Load Balancing using Genetic Algorithm

Singh et al.[9] have proposed a load balancing algorithm with Genetic Algorithm. There are various factors which are needed in load balancing with genetic algorithm such as load measure, fitness function, coding method and algorithm. Genetic algorithm was used to allocate loads to processors, with the objective to minimize the

difference between the loads of each processor. With the help of genetic algorithm suitable candidate receiver is decided to which request message should be sent off. In terms of achieving the goals of maximum processor utilization and minimum total completion time, genetic based load balancing algorithm performs really well.

V. Conclusion and Future Work

Computing resources are rapidly developing, and growing uses of heterogeneous system with parallel or distributed computing issue of load imbalance has emerged and load balancing is solution to such issues. Here we presented that load balancing distributes the load evenly among nodes, hence increasing overall performance. In this paper, we have discussed various load balancing techniques for distributed computing environment. The purpose of load balancing is to improve the performance of a distributed system through an appropriate distribution of the application load. This ensures that every resource is distributed efficiently and evenly. An efficient load balancing is vital for system performance, to maximize throughput, to maintain system stability and fault tolerant.

References

[1] Sandeep Sharma, Sarabjit Singh, and Meenakshi Sharma, "Performance Analysis of Load Balancing Algorithms", World Academy of Science, Engineering and Technology, 2008.
[2] Hisao Kameda, El-Zoghdy Said Fathyy and Inhwan Ryuz Jie Lix, "A Performance Comparison of Dynamic vs. Static Load Balancing Policies in a Mainframe { Personal Computer Network Model", Proceedings of the 39th IEEE Conference on Decision and Control, 2000.
[3] Abhijit A Rajguru, SS Apte, "A Comparative Performance Analysis of Load Balancing Algorithms in Distributed System using Qualitative Parameters" , International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-1, Issue-3, August 2012.
[4]. Foster, I., Y. Zhao, I. Raicu and S. Lu, "Cloud Computing and Grid Computing 360-degree compared," in proc. Grid omputing Environments Workshop, pp: 99-106, 2008.
[5]. Buyya R., R. Ranjan and RN. Calheiros, "InterCloud: Utilityoriented federation of cloud computing environments for scaling of application services," in proc. 10th International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP), Busan, South Korea, 2010.
[6] Ming-Chang Huang, S. Hossein Hosseini1 and K. Vairavan, "Load Balancing in Computer Networks"
[7] khulood Ahmed Nassar and ZainabSaadKaram AL-Musawi, "Fuzzy Neural Network for Dynamic load balancing of nodes for ad hoc network using", Journal of Basrah Researches ((Sciences)) Vol.(39). No.(2) .A (2013)
[8]. Nishant, K. P. Sharma, V. Krishna, C. Gupta, KP. Singh N. Nitin and R. Rastogi, "Load Balancing of Nodes in Cloud Using Ant Colony Optimization." In proc. 14th International Conference on Computer Modeling and Simulation (UKSim), IEEE, pp: 3-8, March 2012.
[9]. Dinesh Babu L.D, P.Venkata Krishna , "Honey Bee inspired Load Balancing of tasks in cloud computing environments", Applied soft computing 13,2292 2303, 2013