Proc. of Int. Conf. on Current Trends in Eng., Science and Technology, ICCTEST

Survey On Dynamic Load Balancing algorithms in Distributed Systems

Manjula K
CSE dept , DonBosco Institutue of technology, Bangalore, India
kinnalmanjula@gmail.com

***Abstract*— Load balancing improves the distribution of workl oads across multiple computing resources, such a computers,a computerluster, network links, central processing units, or disk drives.Load balancing always aims to optimiallyutilizeresource use, maximize throughput, minimize response time, and avoid overload of any single resource**

***Keywords*— component; formatting; style; styling; insert (key words).**

I. INTRODUCTION

A distributed system is defined to be a collection of computing and Communication resources shared by active users. As the demand for Computing power increases the load balancing problem becomes critical. The load balancing problem can be stated as follows: Given the initial set of job Arrival rates at every node where the system identifies an allocation of jobs for the computers in order to minimize the response time of the entire system with the entire set of jobs. A distributed system has many number of processors which will be working independently with each other and will be linked by communication channel[10]. Each processor has an initial load which is the amount of work to be performed, and every node has a different processing ability

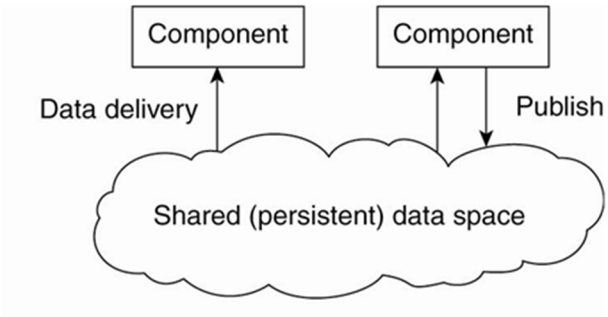

Figure 1. Distributed Systems

Grenze ID: 02.ICCTEST.2017.1.117

A. Necessity for load balancing

The workload need to be evenly distributed among all processors based on their processing speed thus the time to execute all tasks gets lesser and the idle time of each processor can be minimized .Thus load balancing becomes necessary . Load imbalance has been a main problem in data parallel applications and it mainly occurs due to the improper distribution of data amongst the various processors in the system. With out a good load distribution strategies and techniques it is difficult to achieve the objective.

B. Problems related to Load Balancing

The various Problems related to load balancing are:

a) In distributed environment the communication channels will be of finite bandwidth and the processing units may be physically far and hence load balancing need to take decision of whether to allow task migration or not
b) A computing job cannot be arbitrarily divided thus it may lead to certain problems in dividing tasks.
c) Each job consists of many smaller subtasks and each of those tasks might have different execution times.
d) The load on each processor also on the network might vary from time to time based on the workload assigned by the users.
e) The processors capacity may be different from each other in terms of architectural design, operation system, CPU speed, memory size, and available disk space.

C. Load balancing algorithms

Round Robin

Round-robin scheduling: The common Round-robin scheduling algorithm, which does not consider any system state information, belongs to static algorithms. Static policies do not consider any system state information. The common Round-robin makes use of a circular list and a pointer to the last selected server to make dispatching decisions, that is, if Si was the last chosen node, the new request is assigned to Si+1, where i = (i+1) mod N and N is the number of server nodes. Therefore, Round-robin utilizes only information on past assignment decision. [3] The advantage of the algorithm is its simplicity. To guarantee fast dispatching of large numbers of requests per second, the web switch cannot use highly sophisticated algorithms. They are the fastest solution to prevent the web switch becoming the primary bottleneck of the web cluster. However, static algorithms can also potentially make poor assignment decisions. A host if selected for process allocation once the process is created and thus one cannot change this selection during process execution to make changes in the system load. Rather than static algorithms, dynamic algorithms usually have state information to help dispatch requests.

Modified Round-robin scheduling

Classical job selection policies do not actively correct such a resource imbalanced state . Rather than the common Round-robin scheduling algorithm, one can consider the system state information to optimally dispatch the results. Modified Round-robin uses a circular list and a pointer to the last selected server to make dispatching decisions. When a request proceeds towards a web switch, if Si is the last chosen node, we calculate the sum of load on each server, if Si+1 is not the largest, the new request is assigned to Si+1. However, if Si+1 is not the largest, the new request is assigned to Si+2 to achieve load-balance. In order to show the algorithm more clearly, the flow chart of modified Round-robin scheduling algorithm is shown in the following Fig. below. The goal of the modified Round-robin load-balancing method is to reduce the overall execution time of a concurrent programs.

Weighted Round Robin

The Weighted Round Robin looks similar to the Round Robin where the way by which requests are assigned to the nodes is will be with a little set of constraints.[8] The node with the higher priorities will be having an ability to serve a greater number of requests, which will be in turn communicated to the load balancer, where one can assign weights to each node. The node with the higher priority should will be given the higher weight. One can usually indicate the weights in proportion to actual capacities. Thus for example, if Server 1's capacity is 5x more than Server 2's, then we can can assign it a weight of 5 and Server 2 a weight of 1

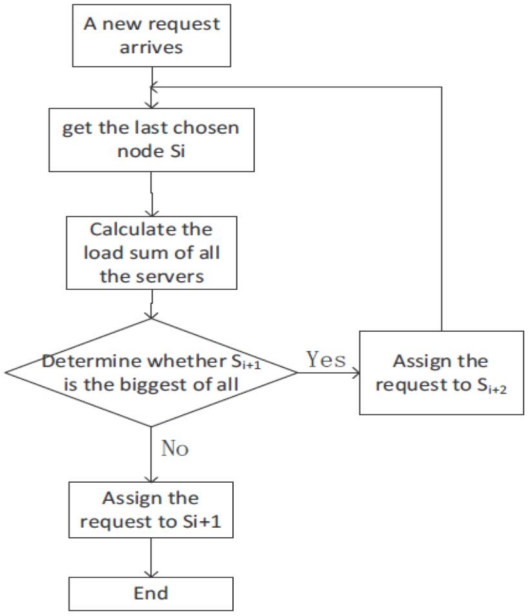

Figure 1. Modified Round Robin

Least Connections

The least-connection scheduling algorithm directs requests received from network to the node with the least number of established connections. This is one of dynamic scheduling algorithms as it requires to count live connections for every node dynamically. At every node where there is a collection of nodes with similar performance, the least-connection scheduling will be preferable for smooth distribution when the lots of requests will be varying, since all long requests will not be having a chance to be directed to a node. Firstly, the least-connection scheduling can also perform optimally even if there are nodes of various 3 processing capacities, since the faster node will get more connections [4] In reality it cannot perform very well in network because of "TCP's TIME_WAIT" state. The TCP's "TIME_WAIT" is usually 2 minutes, between this 2 minutes a busy web site many number of times get thousands of connections, for example, the node A is twice as powerful as the node B, the node A has processing thousands of requests and kept them in the "TCP's TIME_WAIT state", but the node B will be crawling to get its thousands of connections finished. So, the least-connection scheduling cannot get load well balanced among nodes with various processing capacities.

Weighted Least Connections

The weighted least-connection scheduling is a superset of the least-connection scheduling where one can assign a performance weight to every node. The nodes with a higher weight value will receive a larger percentage of live connections at any given time. The node administrator will be having the provision to assign a weight to each node, and network connections are scheduled to each node in which the percentage of the current number of live connections for each node will be the ratio to its weight.

Biased Random(BRALB)

'BRALB is implemented on a square grid based network topology with each node having four neighbors except the border nodes'. In BRALB a node will have a message counter for each neighbor. When node A sends a message to say node B, then node A will increase the counter designate for node B by a value of 1. On the other hand the receiving node B will update its counter designated for node A; increasing it by a value of 4.

The reason for the four step increment is to try for a and predict the message sent by that particular neighbor. It is a know probability fact that when one have an equal chances of selecting four neighbors to send some messages then after some time if one of those neighbors received say 5 messages then it is reasonable to say

that the remaining three neighbors may as well get each 5 messages each. In this way it is a fair guess to say that the sending nodes must have sent 20 messages in all [5]. By this way then one can say for every message received from a neighbor then one can fairly guess that the neighbor may have sent 4 messages hence the four step increment. In this way all the messages sent and received by a neighbor can be predicted and therefore the energy level of that particular node can be predicted as there is a correlation between node energy and messages processed by that node. Therefore, when node A with four neighbors wants to send a message to the sink which is say six hops away from itself, it will first inspect its counters and select the node with the least count. This process is repeated until the message reaches the Sink. In it has been proven statistical that in application whereby the message to be sent to the Sink is comparable to the size of the inquiry message then this routing mechanism can route to the Sink by using the same energy as the SPF algorithm.

The messages sent and received represent the energy used in the network and therefore by biasing the nodes to forward to the nodes with fewer messages (both sent and received) the network load balances the energy of the network. This will avoid using the shortest path all the time by distributing the energy usage fairly within the network and hence will avoid partitioning of the network.

Hash

The hash algorithm uses the IP address of the client or the value of an HTTP header as the basis for server selection.With an HTTP header, use the Load Balancer Hash Header property to identify the header to read. When you configure a service without the wizard, this property is available on the Main tab. This property is available for only the following services: Multi-Protocol Gateway,Web Service ProxyWith the hash algorithm, the same client is served by the same server. Use this algorithm only when clients access applications that require the storage of server-side state information, such as cookies. Hashing algorithms cannot ensure even distribution.[6]

D Comparitive analysis of the algorithms

TABLE I. COMPARITIVE ANALYSIS

Algorithms	Merits/Demerits
Round Robin	Simple
Modified Round Robin	Faster Execution rate
Weighted Round Robin	Fast Dispatching ability
Least Connections/ Weighted Least Connections	Does smooth distribution of tasks, cannot get load well balanced among nodes with various processing capacities
Biased Random	Distribution of energy usage fairly
Hash	Cannot perform even distribution

REFERENCES

[1] Load Balancing Algorithms for Internet Vhdeo and Audio Server Dusit Niyato Chutim.et Srinilta
[2] A modified Round-robin Load Balancing algorithm for cluster-based Web Server XUZongyu1,WANG Xingxuan1
[3] A Predictive Modified Round Robin Scheduling algorithm for web server clusters XU Zongyu1 , WANG Xingxuan
[4] Least-Connection Algorithm based on variable weight for multimedia transmission YU SHENGSHENG, YANG LIHUI, LU SONG, ZHOU JINGLI
[5] Biased Random Algorithm for Load Balancing in Wireless Sensor Networks (BRALB)
[6] Distributed Hashing for Scalable Multicast in Wireless Ad Hoc Networks Saumitra M. Das IEEE
[7] K. H. Yeung, K. W. Suen and Y. K. Wong. “Least Load Dispatching Algorithm for Parallel Web Server Nodes.” IEE Proceedings on Communications, vol. 149, no.4, pp. 223- 226, 2002.
[8] A Task Scheduling Strategy Based on Weighted Round-Robin for Distributed Crawler Dajie GeDept. of Comput. Sci. & Technol., Tongji Univ. Shanghai, Shanghai, China
[9] Efficient and Scalable Multiprocessor Fair Scheduling Using Distributed Weighted Round-Robin Tong Li Dan Baumberger Scott Hahn
[10] Dynamic Load Balancing in Distributed Systems in the Presence of Delays: A Regeneration-Theory Approach Sagar Dhakal, Majeed M. Hayat, Senior Member, IEEE, Jorge E. Pezoa, Cundong Yang, and David A. Bader, Senior Member, IEEE