Volume 5, Issue 7, July 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

A Hybrid Algorithm for Load Balancing

Rashmi Saini
Computer Science Department
Assistant Professor, GBPEC
Ghurdauri, India

Ashish Bisht
Computer Science Department
M.tech Student, GBPEC
Ghurdauri, India

Abstract— Load balancing aims to distribute the client requests over multiple web servers, thus preventing an overload to a single server. The increase in internet users leads to heavy traffic, so the load must be routed efficiently, for which the load balancing technique is used. Load balancing techniques routes and distributes the heavy traffic evenly across the multiple web servers so that a single server is not overloaded. This paper focuses on two commonly used load balancing techniques i.e., Round-Robin algorithm (Static Load balancing) and Least Load algorithm (Dynamic Load balancing). In this paper, these two algorithms are analyzed and compared and a Hybrid load balancing algorithm is introduced, which inherits the properties from both static and dynamic load balancing techniques and tries to overcome the limitation of both the algorithms.

Keywords— Load balancing, Dispatcher, Distributed Cluster Server, Static Load balancing, Dynamic Load balancing.

Abbreviation— Distributed Cluster Server (DCS), Distributed Web Server System (DWS), Static load balancing (SLB), Dynamic load balancing (DLB), First-Come-First-Serve (FCFS).

I. INTRODUCTION

In our modern world, more and more people rely on the internet for their daily activities. Consequently, organizations and companies are providing their services online. Since persistent improvement in hardware performance is no longer sufficient to cope up with the growing volume of client requests while preserving desirable service quality, it is now a common practice to use multiple servers to process client requests simultaneously. However, when multiple servers are working at the same time, load balancing becomes an important issue. If we cannot distribute incoming requests uniformly among servers effectively, some servers may become overloaded while the others may remain idle, leading to low server utilization and poor quality of services [2].

Popular Web sites cannot rely on a single powerful server or on independent mirrored-servers to support the ever-increasing request load. Distributed Web server architectures that transparently schedule client requests offer a way to meet dynamic scalability and availability requirements [9]. To enhance the availability and reliability, multiple servers are clustered to handle multiple client requests concurrently. The clustered servers offer the liberty of user transparency which allows the client machine to work with multiple servers without any particular changes in the usage. At the front-end, there is a load balancer (also known as Dispatcher) is used to distribute client requests to a specific server at the back-end. Reliability of the System is maximized as full host redundancy effectively removes any single-point-of failure, i.e., a failed host can be automatically replaced by another operational host within the cluster. To redirect the requests among servers in the cluster, load-balancing policy is imperative. Many popular policies used today include load-based distribution, round-robin distribution and random distribution.

A Distributed Cluster Server (DCS) is a collection of servers connected together via LAN. The client's requests are redirected to one of the servers in the DCS transparently. The main purpose of the load balancing is to decrease the total response time and maximize the overall throughput where all operations are transparent to the user. Load balancing is a computer networking technique for distributing client requests and workloads across multiple computers or a computer cluster, network links, central processing units, disk drives, or other resources. Successful balancing of load enhances resource use, maximizes throughput, minimizes response time, and avoids overload.

II. LOAD BALANCING IN DWS

The problem with web server farms is that a load imbalance may occur, where some processors are idle with nothing to execute, while other processors are busy and have tasks in their ready queues [1]. Any load imbalance will consequently result in lesser task response time. Scheduling policies determine which requests in the queue are serviced at any point of time, how much time is spent on each, and what happened when a new request arrives.

The objectives of scheduling policies are to minimize the mean round around time of the request and to behave evenly to all requests. Static algorithms are the fastest solution since they do not have overloading decision making time. Static algorithms can sometimes make poor assignment decisions, such as redirecting a request to a server node having a long queue of waiting load while there are other almost idle non-working nodes. Dynamic algorithms have the potential to outperform static algorithms by using some state information to help dispatching decisions [3].

III. PARAMETERS

A. Overload Rejection:

Additional overload rejection measures are needed if Load Balancing is not possible. When the overload situation ends then first the overload rejection measures are stopped. After a short period Load Balancing is also closed down.

B. Fault Tolerant:

Algorithm is able to tolerate faults or not. An algorithm can continue operating properly if any failure occurs. If the performance of algorithm decreases, the decrease is proportional to the seriousness of the failure, even a small failure can cause total failure in load balancing.

C. Forecasting Accuracy:

Forecasting is the degree of compliance of calculated results to its actual value that will be produced after execution. The static algorithms provide more accuracy as compared to dynamic algorithms as in static algorithms, most assumptions are made during compile time and in dynamic algorithms, this is done during execution.

D. Stability:

Stability can be characterized in terms of the delays in the exchange of information between processors and the gains in the load balancing algorithm by obtaining faster performance within a specified amount of time.

E. Centralized or Decentralized:

Centralized schemes store global information at a designated node. All sender or receiver nodes access that particular designated node in order to calculate the total amount of load-transfers and also to check that tasks are to be sent to or received from. In a distributed load balancing, every node executes balancing individually. The idle nodes can obtain load during runtime from a shared global queue of processes.

IV. NATURE OF LAOD BALANCING

Static load balancing (SLB) assigns load to nodes probabilistically or deterministically without considering the runtime events. It is quite impossible to make predictions of arrival times of loads and processing times required for future loads. On the other hand, the load distribution is made during run-time based on current processing rates and network condition in case of dynamic load balancing. A DLB policy can use either local or global information.

A. Cooperative:

This parameter defines the extent of independence that each processor has in concluding that how should it can use its own resources. In the cooperative situation all processors have the accountability to carry out its own portion of the scheduling task, but all processors work together to achieve a goal of better efficiency. In the non-cooperative individual processors act as independent entities and arrive at decisions about the use of their resources without any effect of their decision on the rest of the system.

B. Process Migration:

Process migration parameter provides when a system decides to export a process. It decides whether to create it locally or create it on a remote processing element. The algorithm is capable to decide that it should make changes of load distribution during execution of process or not.

C. Resource Utilization:

Resource utilization include automatic load balancing A distributed system may have unexpected number of processes that demand more processing power. If the algorithm is capable to utilize resources, they can be moved to under loaded processors more efficiently.

V. LOAD BALANCING ALGORITHMS

A. Static Load Balancing

In this method, the performance of the processors is determined at the beginning of execution [5]. Then depending upon their performance, the work load is distributed in the start levels. Initially, all the processors are considered to be under loaded. When the load state of a processor exceeds a load level limit, then it sends messages regarding the new load state to all remote processors, regularly updating them as to the actual load state of the entire system. If the local state is not overloaded then the process is allocated locally.

Otherwise, a remote under loaded processor is selected, and if no such host exists, the process is also allocated locally. Threshold algorithms have low inter process communication and a large number of local process allocations. The later decreases the overhead of remote process allocations and the overhead of remote memory accesses, which leads to improvement in performance. A disadvantage of the algorithm is that all processes are allocated locally when all remote processors are overloaded.

FCFS (First Come First Serve) is one of the static load balancing algorithms used to ensure fairness in a number of application domains such as scheduling and Operating Systems. This is a non-preemptive technique. A single queue of ready processes is maintained, and the dispatcher always picks up the first one. This method does not emphasizes

throughput, since long processes are allowed to exclusively possess the CPU, for the same reason, the response time with FCFS can be high with respect to execution time, especially if there is a high variance in process execution times.

1) *Round Robin:* This is a simple technique in which the user's content access request is responded to by the load balance in a rotational manner, the first request grants access to the first available content server giving its IP address and the second to the second server IP address and so on. Whenever a server IP address has been given, instantly its IP address is moved to the back of the list of available IP addresses and gradually it moves back to the top of the list and becomes available again. The frequency that it returns to the top depends on the number of available servers in the round robin server cluster being used. A good way to think of this is a method of server allocation on a continuous looping fashion.

The round-robin scheduling algorithm sends each incoming request to the next server in its list. Thus in a three server cluster (servers A, B and C) request 1 would go to server A, request 2 would go to server B, request 3would go to server C, and request 4 would go to server A, thus completing the cycling or "round-robin" of servers. It treats all real servers as equals regardless of the number of incoming connections or response time each server is experiencing.

ALGORITHM:

```
BEGIN PROCEDURE ROUND_ROBIN_ALGO
SET Static Integer COUNT = 0;
SET integer QUANTUM =  q;
SET array SERVERS = {s1,s2,s3……sn};
Integer SQ = q*n;
WHILE ( request) DO
        Integer RANGE = COUNT % SQ;
        IF RANGE > 0 AND RANGE < = 1*Q THEN
                        GOTO SERVER[s1]
        ENDIF
        IF RANGE > 1*Q AND RANGE < = 2*Q THEN
                        GOTO SERVER [s2]
        ENDIF
        :
        IF RANGE > (n-1)*Q AND RANGE < = n*Q THEN
                        GOTO SERVER [sn]
        ENDIF
END WHILE
END PROCEDURE
```

B. Dynamic Load Balancing

It differs from static algorithms in that the work load is distributed among the processors at runtime. The master assigns new processes to the slaves based on the new information collected unlike static algorithms, dynamic algorithms allocate processes dynamically when one of the processors becomes under loaded. Instead, they are buffered in the queue on the main host and allocated dynamically upon requests from remote hosts.

1) Least Load: Another policy that provable optimal mean round around time for all requests is SRPT "Shortest Remaining Processing Time". Shortest-Remaining-Processing-Time (SRPT) scheduling policy is an optimal algorithm for minimizing mean response time. The job that has a least remaining process time will be served. There are two problems, this policy not fair. Jobs with large size may be waiting for a while and Dispatcher must know size of jobs beforehand.

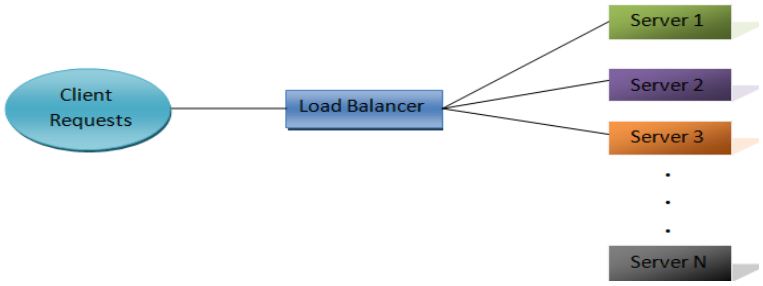

Figure 1 Least Load for client requests

TABLE 1 ORDER OF REQUESTS SERVICED BY SERVERS

Server	N	Next Request Serviced by
Server 1	**3**	
Server 2	**5**	⇐
Server 3	**2**	


```
Algorithm:
BEGIN PROCEDURE LEAST_LOAD_ALGO
SET array SERVERS = {s1,s2,s3……sn};
SET  array SERVER_LOAD  = {l1,l2,l3……..ln};
WHILE ( request) DO
        Integer POS = FIND_MIN( SERVER_LOAD);
        GOTO SERVER[POS];
END WHILE
END PROCEDURE
BEGIN PROCEDURE FIND_MIN (SERVER_LOAD [1 to n])
Set integer POS = 0;
For I = 1 to n-1 do
        IF SERVER_LOAD [POS] > SERVER_LOAD[I] THEN
                POS = I;
        END IF
RETURN POS;
END PROCEDURE
```

VI. LIMITATIONS OF ROUND ROBIN AND LEAST LOAD ALGORITHM

Both Least Load and Round Robin algorithms for load balancing works good in some particular criteria but both have their limitations. Round Robin is simple and runs fast but lack in precision when it comes to load balancing of various size and complexity of load or request. On other hand, Least Load algorithm consider the load at particular server before distributing it to the server, which results in a possibility when only a set of server gets load in case of sparse load condition thereby leaving some of system idol and decreasing the efficiency of the system.

TABLE 2 COMPARISON OF LAOD BALANCING ALGORITHMS

Properties	**Round Robin**	**Least load**
Dynamic/Static	Static	Dynamic
Stability	High	Medium
Cooperative	No	Yes
Resource Utilization	Low	Medium

VII. HYBRID ALGORITHM

In hybrid algorithm, functionalities from both the least load and round robin algorithms are inherited which makes this algorithm simple, fast and efficient. Some of the features of Hybrid load balancing algorithm are:

- It considers the server load before sending the request to the corresponding server.
- Oversee the previously selected server.
- This helps the selected server to not participate in next server load decision.
- The sparse requests can be easily distributed among the n servers, more evenly.

```
Algorithm:
BEGIN PROCEDURE HYBRID_ALGO
SET static integer LAST_SELECT = 0;
SET array SERVERS = {s1, s2, s3……sn};
SET array SERVER_LOAD  = {l1,l2,l3……..ln};
WHILE (request) DO
        Integer POS = FIND_MIN_H (SERVER_LOAD, LAST_SELECT);
        LAST_SELECT = POS;
        GOTO SERVER [POS];
END WHILE
END PROCEDURE
BEGIN PROCEDURE FIND_MIN_H (SERVER_LOAD [1 to n] , LAST_SELECT)
        SET integer POS = 0;
        For I = 1 to n-1 do
        IF I=LAST_SELECT THEN
        Continue;
        END IF
        ELSE IF SERVER_LOAD [POS] > SERVER_LOAD[I] THEN
                POS = I;
        END ELSE IF
        RETURN POS;
END PROCEDURE
```


VIII. RESULTS AND ANALYSIS:

TABLE 3 ROUND ROBIN

	Total Data (MB)	Request/sec	Bandwidth (Bytes/sec)	Busy Threads
Main server	168	5.78	480.23	8
Server 1	66	2.89	151.678	2
Server 2	78	3.032	167.35	4
Server 3	99	2.732	154.897	3

TABLE 4 LEAST LOAD

	Total Data (MB)	Request/sec	Bandwidth (Bytes/sec)	Busy Threads
Main server	189	6.078	512.899	7
Server 1	57	3.189	168	3
Server 2	65	3.032	176.789	4
Server 3	57	3.94	164.234	3

TABLE 5 HYBRID

	Total Data (MB)	Request/sec	Bandwidth (Bytes/sec)	Busy Threads
Main server	180	8.67	497.567	7
Server 1	57	4.78	170	4
Server 2	63	3.986	168.095	4
Server 3	69	4.013	164.234	3

Above table compares the performance and efficiency of all mentioned algorithms. Table (1) depicts that the main server has 8 busy threads while server 1 has only 2 busy threads, which concludes that load is not distributed evenly to the servers. In table (2), requests are more evenly distributed among servers as compared to Round-Robin in table (1), but total data (in MB) processed by the servers 1, 2 and 3 varies from each other but in table (3), the data and bandwidth is most evenly distributed as compared to Round-Robin and Least load algorithms.

IX. CONCLUSIONS

This paper discussed about different algorithms that are used to deal with the distribution of load among servers, caused by requests from users/clients to servers. Apparently the above comparison table depicts that static load balancing algorithms are more stable in compare to dynamic but it is not cooperative and also has low resource utilization, while on other hand, despite of the fact that dynamic distributed algorithms are more complex and have communication overheads. But dynamic distribution methods are always considered better than static algorithms as they are closely related to real life scenarios and real time processing. Result shows that hybrid algorithm performs better in terms data and bandwidth that is more evenly distributed as compared to Round-Robin and Least load algorithms.

REFERENCES

[1] Harikesh Singh, Dr. Shishir Kumar *"Dispatcher Based Dynamic Load Balancing on Web Server System*, International Journal of Grid and Distributed Computing, Vol. 4, No. 3, September, 2011.

[2] Daniel A.Menascé, George Mason University *"Trade-offs in Designing Web Clusters"* IEEE Internet Computing1089- 7801/ 02 ©2002 IEEE.

[3] O.K.Tonguz and E.Yanmaz, ―*On the Theory of Dynamic Load Balancing*, in Proc. IEEE Global Telecom. Conf. (GLOBECOM'03), vol. 7, pp. 3626-3630, Dec. 2003.

[4] Barazandeh and I.Mortazavi. S.*S " Dynamic Load Balancing algorithms in Distributed Systems"* Proceedings of 2009 second International Conference on Computer and Electrical engineering pp 516-521.

[5] Hua-fengdeng, yun - sheng liu, ying-yuanxiao. *" A novel algorithm for load balancing in distributed systems"* proceedings of the 2007 eight ACIS International Conference on software engineering, Artificial Intelligence, Networking and parallel/distributed computing, pp 15-19.

[6] Neeraj Nehra, R.B. Patel, V.K .*Bhat "A multi-agent system for distributed dynamic load balancing on cluster."* Proceedings of the International Conference on Advanced Computing and communications, 2006, pp 135-138.

[7] Cardellini, Colajanni, and Philip S. Yu, ―*Dynamic Load balancing on web-server systems*, published in IEEE internet Computing, vol. 3, no. 3, pp 28-39, 1999.

[8] D. Grosu, A. T. Chronopoulos and M. Y. Leung, "Cooperative Load Balancing in Distributed Systems", *Journal of Parallel and Distributed Computing*, Submitted 2002.

[9] H.W. D. Chang andW. J. B. Oldham. Dynamic task allocation models for large distributed computing systems. *IEEE Trans. Parallel and Distributed Syst.*, 6(12):1301–1315, December 1995.

[10] Y. C. Chow and W. H. Kohler. Models for dynamic load balancing in a heterogeneous multiple processor system. *IEEE Trans. Comput.*, C-28(5):354–361, May 1979.

