A Comparative Study of Various Load Balancing Strategies for Performance Analysis in Distributed System

[1] Mr. VipinWani, [1]Assistant Professor, [1]Department of Information Technology,
[1]SNJB's KBJ CoEChandwad , India.

Abstract
A distributed system is a collection of computing and communication resources shared by multiple active users. In this environment, a number of nodes or workstations or computers are linked through a communication network to form a large distributed computing system. When the demand for computing resources increases, the load balancing problem becomes important. The problems of load balancing in distributed system are most important and challenging area of research in computer engineering. This paper focuses on the problem of transparent redistributing the load of the system among its nodes so that overall performance is maximized. We discuss various key issues in load distributing for general-purpose systems, including the needs, strategies of load-distribution and load distribution policies. In addition, we discuss several load-balancing algorithms, Then Analyze and compare their performance. We also survey and draw conclusions about which algorithm might help in realizing the most benefits of load distributing.
Keywords: Load Balancing, Static and Dynamic load balancing, load distribution.

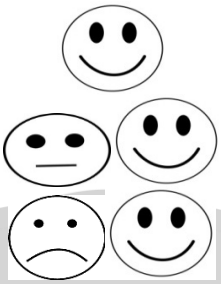
I.INTRODUCTION

In parallel and distributed systems more than one processor process parallel programs. The amount of processing time needed to execute all processes assigned to a processor is called workload of a processor [1]. The amount of processing time needed to execute all processes assigned to processor is called load of a processor. A distributed system provides the resource sharing as one of its major advantages. The development of effective techniques for distributing load/processes on multiple processors One of the biggest research issues in distributed computing systems. The main goal is to distribute processes among processors to maximize throughput, maintain stability, maximize resource utilization, minimize communication delays and fault tolerant[2]. Balancing the load distribution between processors is an important Challenge which is responsible for completing the computation in the shortest possible time.

Figure 1. A load Balancing Process

1.1 Need of Load Balancing

A distributed system contains number of processors working independently with each other and linked by communication channel. Some are not linked with any communication channel. Each processor has an initial load that is the amount of work to be performed, and each may have a different processing capacity. The workload has to be evenly distributed among all processors based on their processing speed so that time to execute all tasks gets minimized and idle time of each processor can be reduced. This is why we need load balancing. Load imbalance is also a main problem in data parallel applications and here also it mainly occurs due to the uneven distribution of data among the various processors in the system. Without good load distribution strategies and techniques, we cannot aim to reach good speedup and good efficiency[8]. Figure 1 and 2 Shows the Unbalanced and Balanced load system respectively.



Equally Loaded

Figure 2.Unbalanced Load System

1.2 Issues related to Load Balancing

The following issues are analyzed during load balancing[8]:

a) In distributed environment the communication channels are of finite bandwidth and the processing units may be physically distant hence load balancing need to take decision of whether to allow task migration or not.

b) A computing job may not arbitrarily divisible leading to certain constraints in dividing tasks.

c) Each job consists of several smaller tasks and each of those tasks can have different execution times.

d) The load on each processor as well as on the network can vary from time to time based on the workload brought about by the users.

e) The processors capacity may be different from each other in architecture, operation system, CPU speed, memory size, and available disk space. Taking above factors into consideration:

The load balancing can be generalized into four basic steps:

a) Monitoring processor load and state

b) Exchanging load and state information between processors

c) Calculating the new work distribution

d) Actual data movement

II.LOAD BALANCING STRATEGIES

Since more than two decades Load balancing problems are the most important and has been research topic in traditional distributed systems literature. Various Methods, strategies and algorithms and matching policies have been proposed, implemented and classified and analyzed.

- **Sender-Initiated vs. Receiver-Initiated Strategies**

Sender Initiated: In this type the load balancing algorithm is initialized by the sender. In sender Initiated algorithm the sender sends request messages till it finds a receiver that can receive or accept the load.

Receiver Initiated: In this type the load balancing algorithm is initiated by the receiver. In ReceiverInitiated algorithms the receiver sends request messages till it finds a sender that can migrate or send the load[1].

- **Centralized vs. De-centralized Strategies**

A load balancer is categorized as either centralized or distributed based on the location of the load balancer[3].

In a centralized scheme, the load balancer is located on Centralized place may be any master workstation or any node and all decisions are made there. In centralized approach A master node keep track of number of tasks to be performed, Tasks are sent to the execution node, When a executing process completes one task, it requests another task from the master node
In a de-centralized scheme, all workstations has the replica of the load balancer. The different algorithms are used in de-centralized scheme for job selection.

2.3 Static Vs Dynamic
combination of both sender and receiver initiated type of load balancing algorithm are called Symmetric algorithms . On the bases of the current state of the system, load balancing algorithm can be categories into two:

2.3.1 Static Load Balancing
In static algorithms [4], prior knowledge about the system is already known which includes processing power, memory, performance and data about user's requirements. These algorithms do not need the information regarding current state of the system. This type of algorithms have serious drawbacks in case of sudden failure of system resource, tasks and also task can't be shifted during its execution for load balancing. Round robin is a example of static load balancing algorithm which divides the traffic equally among servers. A lot of problems were appearing in round robin algorithm to overcome these problems a new algorithm is proposed called Weighted Round Robin. The main concept behind this algorithm is that each server has been assigned a weight and then, server having the highest weight receives more connections. In equal weighted condition, servers will receive balanced traffic. This approach is generally defined during the design or implementation period of the system.

- **Round Robin Algorithm:** Round Robin algorithm[6] distributes jobs evenly to all slave processors. All jobs are assigned to slave processors based on Round Robin order, meaning that processor choosing is performed in series and will be back to the first processor if the last processor has been reached. The Selection of Processors is done on each processor individually and independently.
- **Randomized Algorithm** Randomized algorithm [5] uses random numbers to choose slave processors. The slave processors are chosen randomly following random numbers generated based on a statistic distribution. for particular special purpose applications Randomized algorithm can attain the best performance among other load balancing algorithms.
- **Central Manager Algorithm:** In CMA algorithm[2]Central processor will choose a slave processor to be assigned a job. The chosen slave processor is the processor having the least load. The load information of all slave processors is gather by central processor, thereof the selecting based on this algorithm are possible to be performed. Based on the system load information the load manager makes load balancing decisions, which allowing the best decision when of the process is created.
- **Threshold Algorithm:** In Threshold Algorithm[9, 10] (TA) tasks or processes are assigned just after its creation to the computing nodes (processors). Then without sending any remote messages Computing nodes for new processes are selected locally. Each node keeps a private copy of the system's load information. The load classification of a computing node can be done by one of the three levels which are: under loaded, medium and overloaded. t_under and t_upper these two threshold parameters can be used to describe these levels :

1. **Under loaded**: workload <t_under**,**
2. **Medium** :t_under≤ workload ≤ t_upper**,**
3. **Overloaded**: workload >t_upper.

2.3.2 Dynamic Load Balancing
Dynamic algorithms are decision concerning load balancing algorithms in which any prior knowledge about the system is not required they are based upon the current state or information of the system . This will overcome the draw backs of static approach. The dynamic algorithms are complex, but they can provide better performance and fault tolerance.

- **Central Queue**

In a Central Queue method new activities and unfulfilled requests are stored as a cyclic FIFO queue on the main host. Every new arriving activity at the queue manager is inserted into the queue. Then, whenever a request for an activity is received by the queue manager, it removes the first activity from the queue and sends it to the requester. The request is buffered If there are no ready activities in the queue, until a new activity is available. If a new activity arrives at the queue manager while there are unanswered requests in the queue, the first such request is removed from the queue and the new activity is assigned to it[6].

- **Local Queue**

In Local queue algorithm[6] static allocation of all new processes with process migration initiated by a host when its load falls under threshold limit, is a user-defined parameter of the algorithm. The parameter defines the minimal number of ready processes the load manager attempts to provide on each processor.

POLICIES OR STRATEGIES IN DYNAMIC LOAD BALANCING

There are different policies in dynamic load balancing [3]:

1. Transfer Policy: The part of the dynamic load balancing algorithm which selects a job for transferring from a local node to a remote node is referred to as Transfer policy or Transfer strategy.

2. Selection Policy: The processors involved in the load exchange is specified by this policy

3. Location Policy: The part of the load balancing algorithm which selects a destination node for a transferred task is referred to as location policy or Location strategy.

4. Information Policy: The part of the dynamic load balancing algorithm responsible for collecting information about the nodes in the system is referred to as Information policy or Information strategy.

5. Load estimation policy: it determines how to estimate the workload of a particular node of the system.

6. Process transfer policy, it determines whether to execute a process locally or remotely.

7. Priority assignment policy: it determines the priority of execution of local and remote processes at a particular node.

8. Migration limiting policy: it determines the total number of times a process, can migrate from one node to another.

MEASURABLE PARAMETERS

Following are the few measurable parameters of load balancing.

4.1. Nature This factor is related with determining the nature or behavior of load balancing algorithms that is whether the load balancing algorithm is of static or dynamic nature, pre-planned or no planning

4.2. Overload Rejection If Load Balancing is not possible additional overload, rejection measures are needed. When the overload situation ends then first the overload rejection measures are stopped. Load Balancing is also closed down after short guard period. Static load balancing algorithms incurs lesser overhead as once tasks are assigned to processors, no redistribution of tasks takes place, so no relocation overhead. More overhead relatively is observed Dynamic Load Balancing algorithms as relocation of tasks takes place.

4.3. Reliability This factor is related with the reliability of algorithms in case of some machine failure occurs. Static load balancing algorithms are comparatively less reliability The reason is no task/process will be transferred or migrated to another host in case a machine fails at run-time. Dynamic load balancing algorithms are more reliable. If the failure occurs processes can be transferred to other machine.

4.4. 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 as this method fails in varying nature problems i.e. situation in which number of processes are not fixed. Comparatively towards every situation Dynamic load balancing algorithms are adaptive whether numbers of processes are fixed or varying one.

4.5. Stability can be classified in terms of the delays in the transfer of information between processors and the gains in the load balancing algorithm by obtaining faster performance by a specified amount of time. As no information regarding present workload state is passed among processors Static load balancing algorithm considered as stable.

4.6. Predictability is to predict the outcome of the algorithm this factor is related with the nondeterministic or deterministic factor. Static load balancing algorithm's behavior can be predicted as most of the things like average execution time of processes and workload assignment to processors are fixed at compile-time. As everything has been done at run time in Dynamic load balancing algorithm its behavior is unpredictable.

4.7. Forecasting Accuracy Forecasting is the degree of conformity of calculated results to its actual value that will be generated after execution.

4.8. Cooperative This parameter describe whether processors share information between them in making the process allocation decision. Cooperative 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. The non-cooperative individual processors act as independent entities and make the decisions about the use of their resources without any effect of their decision on the rest of the system.

4.9. Fault Tolerant It enables an algorithm to continue operating properly in the event of some failure. If the performance of algorithm is proportional to the seriousness of the failure, even a small failure can also cause total failure in load balancing.

4.10. Resource Utilization It include automatic load balancing A distributed system may have unexpected number of processes that demand more processing power. Resources can be moved to under loaded processors more efficiently If the algorithm is capable to utilize resources. Static load balancing algorithms utilizes comparatively less amount of resources as static load balancing methods just tries to assign tasks to processors in order to achieve minimize response time ignoring the fact that some processors finish their work early and sit idle due to lack of work. As dynamic load balancing take care of the fact that load should be equally distributed to processors so that no processors should sit idle they relatively have higher resource utilization.

4.11. Process Migration this parameter provides when does a system decide to export a process? It decides whether to create it locally or create it on a remote processing element. The algorithm is capable of making decision about change in load distribution during execution of process or not.

4.12. Response Time How much time a distributed system using a particular load balancing algorithm is taking to respond? The response time in Static load balancing algorithms is shorter as one should not forget that in Static load balancing there is lesser overhead, so emphasis is totally on executing jobs in shorter time rather than optimally utilizing the available resources. Relatively higher response time is observed in Dynamic load balancing algorithms as sometimes redistribution of processes takes place. Some time is being consumed during task migration

4.13. Preemptiveness This factor is related with checking the fact that whether load balancing algorithms are inherently non-preemptive as no tasks are relocated. Dynamic load balancing algorithms considered as both non preemptive and preemptive.

4.14. Waiting Time Waiting Time is the sum of the periods spent waiting in the ready queue.

4.15. Turnaround Time describe the interval from the time of submission of a process to the time of completion is the turnaround time.

4.16. Execution System Centralized schemes store global information at a designated node. To calculate the amount of load-transfers all sender or receiver nodes access the designated node and also to check that tasks are to be sent to or received from. In a distributed load balancing, every node executes load balancer separately. During the runtime the idle nodes can obtain load from a shared global queue of processes.

4.17. Through put is the amount of data moved successfully from source to destination in a given time period.

4.18. Processor Thrashing It plays the role when most of the processors of the system are spending most of their time for migrating processes without accomplishing any useful work in an attempt to properly schedule the processes for better performance. As no relocation of tasks place Static load balancing algorithms are free from Processor thrashing. Dynamic load balancing algorithms incurs substantial processor thrashing.

RESULT & ANALYSIS

Following table shows the comparative analysis of each load balancing algorithms.

Table 1: Comparative Analysis of Load Balancing Algorithms

Parameters	Round Robin	Random	Local queue	Central Queue	Central Manager	Threshold
Nature	Static	Static	Dynamic	Dynamic	Static	Static
Overload Rejection	No	No	Yes	Yes	No	No
Reliability	Less	Less	More	More	Less	Less
Adaptability	Less	Less	More	More	Less	Less
Stability	Large	Large	Small	Small	Large	Large
Predictability	More	More	Less	Less	More	More
Forecasting Agency	More	More	Less	Less	More	More
Cooperative	No	No	Yes	Yes	Yes	Yes
Fault Tolerant	No	No	Yes	Yes	Yes	No
Resource Utilization	Less	Less	More	Less	Less	Less
Process Migration	No	No	Yes	No	No	No
Preemptiveness	Non-preemptive	Non-preemptive	Preemptive and Non-preemptive	Preemptive and Non-preemptive	Non-preemptive	Non-preemptive
Response Time	Less	Less	More	More	Less	Less
Waiting Time	More	More	Less	Less	More	More
Turnaround Time	Less	Less	More	More	Less	Less
Execution System	Decentralized	Decentralized	Decentralized	Centralized	Centralized	Decentralized
Throughput	Low	Low	High	High	Low	Low
Processor Thrashing	No	No	Yes	Yes	No	No

CONCLUSION

The focus of this paper was to Analyze and compare different load balancing algorithms based on measurable parameters for the performance. In this Review we have gone through the analysis of different load balancing algorithms, various factors are used to analyze different load balancing algorithms. From this review we can conclude that load balancing algorithms are totally dependent upon situations in which workload is assigned, either compile time or execution time. From above comparison and analysis we can conclude that static load balancing algorithms are more stable than dynamic. But dynamic load balancing algorithms are always better than static when compared the parameters like overload rejection, reliability, adaptability, cooperativeness, fault tolerant, resource utilization, response & waiting time and throughput.

ACKNOWLEDGMENT

The preferred spelling of the word “acknowledgment” in Americais withou tan “e” after the “g”.Avoidthestiltedexpression, “Oneofus(R.B.G.)thanks...”
Instead,try“R.B.G.thanks”.Putapplicablesponsoracknowledgmentshere;DONOTplacethemonthefirstpageofyourpaperorasafootnote.

REFERENCES

[1].Abhijit A. Rajguru, S.S. 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.
[2] Abirami M S1, Niranjana G2,” Dynamic Load Balancing in Distributed Systems”, International Conference on Computing and Control Engineering (ICCCE 2012), 12 & 13 April, 2012
[3] Ali M. Alakeel, A Guide to Dynamic Load Balancing in Distributed Computer Systems, IJCSNSInternational Journal of Computer Science and Network Security, VOL.10 No.6, June 2010.
[4]. Nitin Kumar Mishra ,Nishchol Mishra “ Load Balancing Techniques: Need, Objectives and Major Challenges in Cloud Computing- A Systematic Review” *International Journal of Computer Applications (0975 – 8887) Volume 131 – No.18, December 2015*
[5]. Abhijit A. Rajguru, S.S. 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
[6]. Deepika, DivyaWadhwa, Nitin Kumar “Performance Analysis of Load Balancing Algorithms in Distributed System” Research India Publications, ISSN 2231-1297, Volume 4, Number 1 (2014), pp. 59-66
[7]. Sandeep Sharma, Sarabjit Singh, and Meenakshi Sharma “Performance Analysis of Load Balancing Algorithms” International Journal of Civil and Environmental Engineering Vol:2, No:2, 2008
[8]. Monika Kushwaha, Saurabh Gupta “Various Schemes of Load Balancing in Distributed Systems- A Review” International Journal of Scientific Research Engineering & Technology (IJSRET), ISSN 2278 – 0882 Volume 4, Issue 7, July 2015
[9]. S. F. El-Zoghdy and S. Ghoniemy, “A Survey of Load Balancing In High-Performance Distributed Computing Systems”, International Journal of Advanced Computing Research, Volume 1, 2014.
[10]. Daniel Grousa and Anthony T., ” Non-Cooperative load balancing in distributed systems”, Journal of Parallel and Distributing Computing, 2005.

