A Performance Analysis of Load Balancing Algorithms in Cloud Environment

Ms. Kunjal Garala
PG Student
B. H. Gardi College of Engineering & Technology
Rajkot, Gujarat, India.
kmgarala@gardividyapith.ac.in

Ms. Namrata Goswami
PG Student
B. H. Gardi College of Engineering & Technology
Rajkot, Gujarat, India
namrata.goswami022@gmail.com

Prof. Prashant D. Maheta
Asst. Professor: CSE Department
B. H. Gardi College of Engineering & Technology
Rajkot, Gujarat, India
pdmehta@gardividyapith.ac.in

Abstract— Distributed systems in a general term can be defined as a mix a combination of resources which are categorized in two ways, computing and communication. Communication network is used to link a number of workstations or nodes to build a large loosely coupled distributed environment. Load Balancing is essential for efficient operations in distributed environments. As Cloud Computing is growing rapidly and clients are demanding more services and better results, load balancing and task scheduling for the Cloud has become a very interesting and important research area. Many algorithms were developed to satisfy all clients' requests efficiently by cloud servers so that overall performance of cloud system can be increased. In this paper, we investigate the different algorithms proposed to resolve the issue of load balancing and task scheduling in Cloud Computing considering two approaches, static and dynamic, for example. The goal of this paper is to help in developing a new algorithm after studying almost all available algorithms.

Keywords: Static and dynamic load balancing techniques, comparative parameters

I. INTRODUCTION

The immense advancement in internet world gave popularity to cloud computing environment. With high flexibility and great retrieval of data as per users' requirements, it provides numerous services. To handle a very large amount of data several techniques to optimize load and streamline operations are needed to achieve desired performance level for the users. The workload of a processor can be defined as the total time required by the processor to execute all the assigned processes [1].

Load balancing involves the distribution of jobs throughout a networked computer system, thus increasing throughput without having to obtain 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]. In Cloud Computing the main concerns involve efficiently assigning tasks to the Cloud nodes such that the effort and request processing is done as efficiently as possible [3], with taking care of other affecting constraints such as heterogeneity and high communication delays.

The main problem is to allocate load in distributed manner to different node of the whole system so that the computation is completed in the optimum possible time.

II. LOAD BALANCING

The general term used for transferring larger processing load to smaller processing node is Load balancing, which enhances the overall performance of the system. In a cloud environment, by using various load balancing techniques improvement in resource utilization, job response time can be measured. An ideal load balancing algorithm avoids overloading or under loading of any specific node. However, for the selection of load balancing algorithm in a cloud computing environment, the additional constraints like security, reliability, throughput and so on, need to be consider. Thus, the algorithm should ensure the normal load of any specific node of the system.

A. Issues of load balancing and scheduling

In distributed systems we don't have any central authority which allocates work to different processors; they have more issues as mentioned below,

1) All the interdependent requirements such as stability, scalability and a small over head to system need to be maintain properly [4].
2) When process migrate from one node to another node running state it became critical to ensure equal workload [1].
3) The completion of execution in shortest possible time is a

very crucial problem of load distribution.

4) Algorithm has to assume that the information stored at node is correct which will avoid the problem of granularity. [1]
5) When the processor is in a state when information can't be share at that time it has to wait till the other processor finish their assigned tasks [1].
6) The exact load requirements and the location of shared data often conflict in the distributed system so it is a major problem for load balancing algorithm [5].
7) As the distributed systems are always non-uniform and non-preemptive so, balancing load of the process in terms of job and tasks in distributed environments is a critical factor which affects the efficiency of whole system. [5]

B. Types of Load Balancing Algorithms

Load balancing algorithms can be broadly categorized into two types:

1) Cooperative: - The common goal of this type of algorithm is to optimize the response time.

2) Non – cooperative: - All running tasks are independent of each other and thus improving the total response time for the local task.

Based on initiation of the process the load balancing algorithms again divided into 3 categories:

1) Sender Initiated: - Overloaded node sends the message for request until it gets a proper node which can accept its load. As mentioned it is initiated by sender only.

2) Receiver Initiated: - Under loaded node sends the request messages till it finds an overloaded node. As mentioned it is always initiated by receiver.

3) Symmetric: - It is a combination of both the above mentioned algorithms.

Load balancing algorithms again broadly divided into two categories based on the present system state.

1) Static: - This type of algorithm does not consider the earlier states as well as performance of a node while doing load balancing.

2) Dynamic: - This type of algorithms considers prior states and performance of a node while distributing load.

III. STATIC LOAD- BALANCING

Static load balancing algorithms are depending on a priori information of the applications and static information about the load of the node. For transferring load from one node to another these algorithms use average behavior of system. They do not consider the existing state of system; rather they consider properties and capabilities like processing power, memory and storage capacity and recently known communication performance.

Static algorithms are basically suitable for homogeneous and steady environments. Moreover, these algorithms are not flexible towards the dynamic changes of the system's load.

Static algorithms always work in master – slave manner, where the performance of any processor is determined before starting the actual execution. After calculating the work load, the master processor assigns work load to their slaves. The slaves calculate their assigned workload and submit it back to the master. As static algorithms are non preemptive, a task will always carried out at the assigned processor. Using static algorithms the reduction in execution time and minimizing the communication delays can easily be achieved.

The main disadvantage of this approach is, the node selection for a process allocation is made at the time of creation of process and it cannot be changed during the execution of a process. This may lead to overloading of a node sometimes and results in poor performance of the all over system.

Different examples of static load balancing algorithms are: Round Robin algorithm, Randomized algorithm, Central Manager Algorithm, Threshold algorithm.

A. Round Robin Algorithm

The Round Robin order is used for job assignment in Round Robin [6] algorithm, meaning processor choosing is performed in sequence. Thus, the allocation of the process will back to the first processor of the queue if the last process of the queue is busy in doing another assigned work. To reduce inter process communication; choosing processor is always performed locally on each processor, independent of allocation of other processor.

Running time of any process is not known earlier in distributed environment so; we can't use this algorithm for a distributed environment, because some nodes may get heavily loaded. Thus, Round Robin is not expected to achieve good performance in general case.

Hence, weighted round-robin algorithm [7] was proposed to solve this problem. Here load will assign depending on the weights assigned to all the nodes. When each node has equal weights, they will receive same traffic. However, it is difficult to predict the execution time so this algorithm is not perfect for cloud computing environment. This algorithm is normally used in web servers where similar HTTP requests are there.

B. Randomized Algorithm

Randomized algorithm [6] uses random numbers based on a statistic distribution to choose slave processors. For particular special purpose applications Randomized algorithm gives the best result compared to all available load balancing

algorithms. This algorithm works well in case of processes are of equal loaded. It does not maintain deterministic approach and provide best performance when Round Robin algorithm generates overhead for process queue.

C. Central Manager Algorithm

In this algorithm, during each iteration, central processor will choose a slave processor for assignment of a job. The central processor is having knowledge of all slaves' load and choosing of node is performed based on this information. The chosen node will always have the least node in the system. The exchange of the load information will create very much inter process communication which will result in poor performance of the system by generating bottleneck at central processor. Especially when different hosts create dynamic activities this algorithm performs better compared to the parallel applications.

D. Threshold Algorithm

In Threshold algorithm [1], when a new process is generated it will be assigned directly to the machine where it is created. For selection of first processor for any process no remote messages are exchanged and the decision is taken locally. Each processor maintains a copy of the system's load. There are three levels to describe the load of a processor: under loaded, normally loaded and overloaded. Two threshold factors are used to determine aforesaid levels and they are t_under and t_upper.

under loaded = load of processor< t_under ,
normal loaded = t_under ≤ load of the processor ≤ t_upper ,
overloaded = load of the processor > t_upper

Firstly, all the processors are under loaded. When the load of a processor exceeds the upper load level limit, then it sends messages regarding it's new load state to all other processors and thus regularly updating the actual load state of the entire system.

When the process is created and if the local state of the node is not overloaded then it is allocated locally. However, if the node is overloaded then process is allocated to remote node which is under loaded, but if the remote node is not found then the process is allocated locally. This algorithm has very low inter process communication and a large number of local process allocations which, decreases the performance of overall system. The later decreases the overhead of remote process allocations and the overhead of remote memory accesses, which leads to improvement in performance [6].

IV. DYNAMIC LOAD BALANCING ALGORITHMS

When the work load is distributed among the processor at runtime then it is known as dynamic load balancing. In that mechanism , master assigns new processes to the slaves based on the new information collected [8]. Dynamic load balancing algorithms can work in two ways: distributed and non-distributed.

In distributed manner all the nodes execute the load balancing algorithm and task of load is shared among them. Nodes interact in two ways: cooperative and non-cooperative. Cooperative way nodes working together to achieve common goals and improve response time, while in non-cooperative way each node works independently. As in dynamic load balancing algorithm each nodes of the system have to communicate with every other node of system, it will generate more messages then non-distributive ones. The main advantage here is, the total load balancing process will get affected, if, one or more node stop working it will just affect the overall performance of system in a certain manner.

In non-distributed type, the task of load balancing is done by either single node or group node. Non-distributed load balancing takes two forms: centralized and semi-distributed. In centralized form one node is solely responsible for load balancing of the whole system and other nodes simply interact with the central node. However in semi-distributed form, nodes are partitioned into clusters and again centralized form is used for load balancing among each cluster. Here the central node is elected in each cluster by using some specific algorithms.

In centralized dynamic algorithm number of messages for interactions are decreases drastically as compared to the semi distributed case and so it reach to a decision very fast. The disadvantage of centralized algorithm is, it causes a bottleneck at the central node and if the central node crashes it became useless, therefore it is useful only for small network.

A. Policies in dynamic load balancing

There are following policies in dynamic load balancing.

1) Transfer Policy: - it selects a job for transforming from local node to remote node.

2) Selection Policy: - it specifies the processors'' selection.

3) Location Policy: - it selects the destination node for the transferred task to be executed.

4) Information Policy: - it is responsible for collecting information about the nodes in the system.

5) Load Estimation Policy: - determines the exact workload of a particular node.

6) Process Transfer Policy: - determines whether to execute process locally or remotely

7) Priority assignment Policy: - determines the priority of local and remote process for a particular node

8) Migration Limiting Policy: - determines the total number of times a process can migrate from one node to another.

Two types of dynamic load balancing: - Central Queue Algorithm and Local Queue Algorithm

B. Central Queue Algorithm

Central Queue Algorithm [9] works on the principle of dynamic distribution environment. For storing new requests and activities the main host maintains a cyclic FIFO. As and when a new activity or request arrives the queue manager inserts it into queue. Whenever a request comes for an activity the queue manager removes the first activity from the queue and sends it to the requester. When there is not a single activity present in the queue the request will be stored in buffer until the new activity is available. Again when the activity arrives at the queue manager, the first request will be assigned the activity.

When a processor finds that the state is under load it will send a request for a new activity to the central load manager and it will answers the request immediately if a ready activity is available in the *process-request queue.*

C. Local Queue Algorithm

Main feature of Local Queue algorithm [9] is dynamic process migration support. Basically it supports static allocation of all new processes with process migration which is initiated by a host when it is under loaded. The load capacity is defined by a user defined parameters. The parameter defines how many minimum ready processes the load manager can attend. When a new process is created on the main host it will be allocated on under loaded hosts. As the first parallel construct of activities is sufficient for allocation of load to all remote hosts, all other processes created on the main host and other hosts then after are allocated locally. When a host is under loaded it randomly sends request with the number of local ready processes to remote load managers. When a load manager receives this type of request, it will compare the local ready processes with the received number and if the local processes are grater then some running processes will be transferred to the requester. The requesting node will send a confirmation to the sender after receiving the processes.

D. Ant Colony Optimization Algorithm

This algorithm [10] inspired on the behavior of real ants to find optimum solution. Ants have ability to find food , when it moving on the way, it laid pheromone on the ground that detect by other ants and follow that path. When more ant choose same path it create denser pheromone and that attracts more ants. Each ant creates a pheromone table according to resource utilization. Under loaded and overloaded nodes found when ants moves in forward direction. For updating pheromone table ant use one strategy, when it find overloaded node move reverse to inform under loaded node. Each ant has separate pheromone table and its update consistency of path after finished trip. Consistency remain same while proceed from source node to destination node. This algorithm [12] is used to find optimal resource allocation for each task in the dynamic cloud system. If ant traverse with poor path, pheromone value will be low, in other side, if ants traverse with good path, pheromone value will be high. Best case scenario is that the under loaded node is found at beginning of the search.

E. Honey Bee Foraging Algorithm

This is nature inspired and self-organizing algorithm [11] used to solve load balancing in dynamic cloud environment. Honeybees have ability to find their food and inform other bees for destination food in the bee's colony. Each forager bees find their food initially then go back to inform other bees. Forager bees used different dancing movements for location and direction of food stuffs. When bees found more food more energetic dance represent to scout bees then it follow forager bees and get food. This phenomena applied on overloaded and under loaded virtual server. When clients request to server if found overloaded it redirect request to other under loaded virtual server.

V. QUALITATIVE PARAMETERS

1) Nature: - Determines the behavior of load balancing algorithms for example, static or dynamic, pre-planned or no planning and so on.
2) Overload Rejection: - This parameter is used to decide the maximum load supported by any node so, it will be used when it is not possible to over load the node. After this parameter is decided a node will not accept any other load. In static load balancing, as no change takes place during the running state of a process there will be lesser overhead and no relocation overhead. Dynamic load balancing gain more overhead compared to static one.
3) Reliability: - It is related to providing reliability against some machine failure. Static load balancing are less reliable then dynamic one as in static algorithms, process will not be transferred in case of machine failure at run-time.
4) Adaptability: - It checks that whether an algorithm is capable of facing the changing situation or not? Static algorithm, as the name suggests they are not adaptive as in changing situation number of processes is not fixed. Whereas dynamic algorithms are completely adaptive.
5) Stability: - It is the amount of time required to transfer of information between processors and to

gain faster performance by a specified amount of time. Static load balancing algorithms are stable as information about the present work load of node is not passed to each other while in dynamic load balancing algorithms these information exchange among the processors.

6) Predictability: - Determines that whether we can predict the outcome of algorithm or not? As in static load balancing algorithm, work load assigned to node is fixed at starting, its outcome is predictable. While in dynamic load balancing the outcome is unpredictable as all decisions are taken at run time.
7) Cooperative: - Some processors communicate with each other while deciding the load balancing among them and some are not during the execution.
8) Fault tolerance: - If some failure occurs the algorithm should work properly this parameter determines the value by which the performance of the system is degraded when any type of fault occurs.
9) Resource Utilization: - When an algorithm is capable of utilizing resources properly it can manage under loaded nodes more efficiently. Resource utilization is less in static algorithms while it is comparatively more for dynamic one.
10) Process Migration: - It determines when a processor should transfer a process? Meaning whether to execute it locally or remotely.
11) Response time: - How much time a node is taking to respond to some particular request? Static load balancing algorithms are shorter in response while dynamics takes comparatively long.
12) Waiting time: - Time required spending in the waiting queue.
13) Turnaround time: - The total time between the submission and the exact completion of the process in the system.

TABLE I. COMPARATIVE ANALYSIS OF LOAD BALANCING ALGORITHMS

Parameters	Central Queue	Local Queue	Honey Bee	Ant Colony	Round Robin	Random	Central Manager	Threshold
Nature	Dynamic	Dynamic	Dynamic	Dynamic	Static	Static	Static	Static
Response Time	More	More	More	More	Less	Less	Less	Less
Resources Utilization	Less	More	More	More	Less	Less	Less	Less
Fault Tolerant	Yes	Yes	Yes	Yes	No	No	Yes	No
Waiting Time	Less	Less	Less	Less	More	More	More	More
Reliability	More	More	More	More	Less	Less	Less	Less
Throughput	High	High	High	High	Low	Low	Low	Low
Turnaround Time	More	More	More	More	Less	Less	Less	Less
Adaptability	More	More	More	More	Less	Less	Less	Less
Stability	Small	Small	Small	Small	Large	Large	Large	Large
Process Migration	No	Yes	Yes	Yes	No	No	No	No
Predictability	Less	Less	Less	Less	More	More	More	More
Cooperative	Yes	Yes	Yes	Yes	No	No	Yes	Yes
Processor Thrashing	Yes	Yes	Yes	Yes	No	No	No	No

VI. CONCLUSION

In this paper based on different qualitative parameters' study and analysis we have compared different load balancing algorithms. Load balancing dependent on what time the work will assign, i.e. during compile time or at execution time. From the aforementioned assessment, we bring to a close that static load balancing algorithms are stable then dynamic one. However dynamic algorithms are always better compared to static ones because of the same above mentioned parameters. In future work, we need to implement all these algorithms and check for specific parameters to choose good load balancing algorithm.

According to analysis on different load balancing algorithms, we consider ant colony optimization algorithm as our future work. In this algorithm, main parameters minimize make span, response time and throughput. Our goal to

improve performance of all standard parameters and include more parameters like capacity of memory, overhead in communication, etc. in heterogeneous cloud environment.

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] Randles, M., D. Lamb and A. Taleb-Bendiab, "A Comparative Study into Distributed Load Balancing Algorithms for Cloud Computing," in Proc. IEEE 24th International Conference on Advanced Information Networking and Applications Workshops (WAINA), Perth, Australia, April 2010.

[4] Daniel Grosua, Anthony T. and Chronopoulosb,"Non-cooperative load balancing in distributed systems", Elsevier, Journal of Parallel and Distributed Computing, 2005.

[5] M. Nikravan and M. H. Kashani, "A Genetic Algorithm for Process Scheduling in Distributed Operating Systems Considering Load balancing", Proceedings 21st European Conference on Modelling and Simulation (ECMS), 2007.

[6] Hendra Rahmawan, Yudi Satria Gondokaryono, "The Simulation of Static Load Balancing Algorithms", 2009 International Conference on Electrical Engineering and Informatics, Malaysia

[7] Zenon Chaczko, Venkatesh Mahadevan, Shahrzad Aslanazadeh, and Christopher, IPCSIT Vol-14, IACSIT Press Singapore,2011.

[8] S. Malik, "Dynamic Load Balancing in a Network of Workstation", 95.515 Research Report, 19 November,2000.

[9] William Leinberger, George Karypis, Vipin Kumar, "Load Balancing Across Near Homogeneous Multi-Resource Servers", 0-7695-0556-2/00, 2000 IEEE

[10] Kun Li, Gaochao Xu, Guangyu Zhao, Yushuang Dong, Dan Wang,"Cloud Task scheduling based on Load Balancing Ant Colony Optimization" in 2011 IEEE Sixth Annual ChinaGrid Conference,pp.3-9,2011.

[11] Ruhi Gupta,"Review on Existing Load Balancing Techniques of Cloud Computing" in International Journal of Advanced Research in Computer Science and Software Engineering,Volume 4, Issue 2, February 2014

[12] Kwang Mong Sim and Weng Hong Sun,"Ant Colony Optimization for Routing and Load-Balancing: Survey and New Directions",IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART A: SYSTEMS AND HUMANS, VOL. 33, NO. 5, SEPTEMBER 2003.