International Journal of Current Engineering and Technology
ISSN 2277 - 4106

Available at http://inpressco.com/category/ijcet

Review Article

# A Study on Efficient Load Balancing Algorithms in Cloud Computing Environment

Uddalak Chatterjee[Á*]

[Á]Department of CSE, Bengal Institute of Technology And Management, Santiniketan, Pin- 73136

Accepted 11 November 2013, Available online 01 December 2013, **Vol.3, No.5 (December 2013)**

***Abstract***

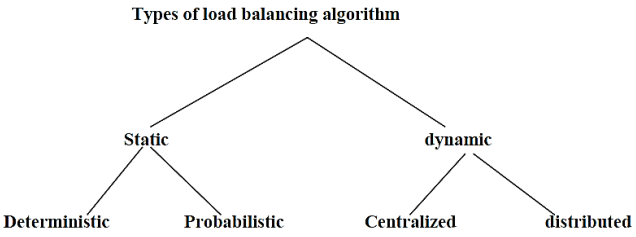
*Cloud computing is an on demand service in which shared resources, information, software and other devices are provided according to the clients requirement at specific time. It's a term which is generally used in case of Internet. The whole Internet can be viewed as a cloud. Capital and operational costs can be cut using cloud computing. Cloud computing is defined as a large scale distributed computing paradigm that is driven by economics of scale in which a pool of abstracted virtualized dynamically scalable , managed computing power ,storage , platforms and services are delivered on demand to external customer over the internet. cloud computing is a recent field in the computational intelligence techniques which aims at surmounting the computational complexity and provides dynamically services using very large scalable and virtualized resources over the Internet. It is defined as a distributed system containing a collection of computing and communication resources located in distributed data enters which are shared by several end users. It has widely been adopted by the industry, though there are many existing issues like Load Balancing, Virtual Machine Migration, Server Consolidation, Energy Management, etc. Central to these issues is the issue of load balancing that is a mechanism to distribute the dynamic workload evenly to all the nodes in the whole cloud to achieve a high user satisfaction and resource utilization ratio. In this study the various and only the most efficient existing algorithms to overcome the problems of load balancing has been discussed.*

***Keywords:*** *Cloud computing, Capital and operational costs*

## Introduction

It is a process of reassigning the total load to the individual nodes of the collective system to make resource utilization effective and to improve the response time of the job, simultaneously removing a condition in which some of the nodes are over loaded while some others are under loaded. Load balancing helps in preventing bottlenecks of the system which may occur due to load imbalance. When one or more components of any service fail, load balancing facilitates continuation of the service by implementing fair-over, i.e. it helps in provisioning and de- provisioning of instances of applications without fail. It also ensures that every computing resource is distributed efficiently and fairly. The load considered can be in terms of CPU load, amount of memory used, delay or Network load. The main goals of load balancing are to improve the performance substantially and to have a backup plan in case the system fails even partially. Another important goal of load balancing is to maintain system stability and to accommodate future modifications. In this paper a study is carried out on different algorithms exists for load balancing in cloud computing

## Types of load balancing algorithm




## Comparison between static and dynamic load balancing algorithms:

1) Static algorithms use only information about the average behavior of the system
2) Static algorithms ignore the current state or load of the nodes in the system
3) Dynamic algorithms collect state information and react to system state if it changed
4) Static algorithms are much more simpler

*Corresponding author**: Uddalak Chatterjee**

5) Dynamic algorithms are able to give significantly better performance

**Static load balancing algorithms**

*Round Robin and Randomized Algorithms*

In the round robin processes are divided evenly between all processors. Each new process is assigned to new processor in round robin order. The process allocation order is maintained on each processor locally independent of allocations from remote processors. With equal workload round robin algorithm is expected to work well. Round Robin and Randomized schemes work well with number of processes larger than number of processors. Advantage of Round Robin algorithm is that it does not require inter-process communication. Round Robin and Randomized algorithm both can attain the best performance among all load balancing algorithms for particular special purpose applications. In general Round Robin and Randomized are not expected to achieve good performance in general case.

*Central Manager Algorithm*

In this algorithm, a central processor selects the host for new process. The minimally loaded processor depending on the overall load is selected when process is created. Load manager selects hosts for new processes so that the processor load confirms to same level as much as possible. From the on hand information on the system load state central load manager makes the load balancing judgment. This information is updated by remote processors, which send a message each time the load on them changes. This information can depend on waiting of parent's process of completion of its children's process, end of parallel execution the load manager makes load balancing decisions based on the system load information, allowing the best decision when of the process created. High degree of inter-process communication could make the bottleneck state. This algorithm is expected to perform better than the parallel applications, especially when dynamic activities are created by different hosts.

*Threshold Algorithm*

According to this algorithm, the processes are assigned immediately upon creation to hosts. Hosts for new processes are selected locally without sending remote messages. Each processor keeps a private copy of the system's load. The load of a processor can characterize by one of the three levels: underloaded, medium and overloaded. Two threshold parameters tunder and tupper can be used to describe these levels.
Under loaded - load < tunder
Medium - tunder ≤ load ≤ tupper
Overloaded - load > tupper

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. Thresholds algorithm 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. A load on one overloaded processor can be much higher than on other overloaded processors, causing significant disturbance in load balancing, and increasing the execution time of an application.

**Dynamic Load balancing algorithms**

**CARTON:** R.stanojevic proposed a scheme for cloud control .It has two steps. First is the sub gradient method that allocates the workloads such that the cost is minimized. And second, a Distributed rate limiting algorithm that allocates the server capacities in a manner that ensures that performance levels at all the servers are equal. CARTON for cloud control that unifies the use of Load Balancing and Distributed Rate Limiting .Load Balancing is used to equally distribute the jobs to different servers so that the associated costs can be minimized and DRL (Distributed Rate Limiting) is used to make sure that the resources are distributed in a way to keep a fair resource allocation. DRL also adapts to server capacities for DRL also adapts to server capacities for the dynamic workloads so that performance levels at all servers are equal.

**Decentralized Content Aware Load Balancing Algorithm:** The content-aware load balancing strategy uses the services or content requested for the scheduling of a request. It considers a special scenario where the application server and database server are deployed over a Distributed Computing Environment . H. Mehta et al proposed a new content aware load balancing policy named as workload and client aware policy (WCAP). This method is a hybrid approach of client aware policy and workload aware request distribution policy discussed below. A parameter unique and special property (USP) is defined to specify the property of the service provider's computing nodes and the consumer's requests for data The WCAP is the hybrid of the WARD and CAP policy. It uses the concept of division of web pages based on their access frequency into two categories core files and part files as described in WARD. The core files are highly accessed files and these are associated with all the nodes in the all the clusters and are intended to be stored in the cache memory of a node. On the other hand part files are divided among the nodes so that all node receive equal number of files. The CAP classifies the web pages into four categories. The WCAP uses the

classifications for the improvement in the search process. By using the content information to narrow down the search, this technique improves the searching performance and hence overall performance of the system. It also helps in reducing the idle time of the computing nodes hence improving their utilization.

**Central load balancing policy for virtual machines (CLBVM):** In this approach each guest is denoted as a job and each Server as a node.A load-balancing policy with a central dispatcher called Central Load Balancing Policy for Virtual Machines (CLBVM) is proposed by Abhay Bhadani et.al, which makes load balancing decisions based on global state information. This policy has centralized information and location rules. The transfer rule is partially distributed and partially centralized. This algorithm balances the load evenly in a distributed virtual machine/cloud computing environment. This policy improves the overall performance of the system.

**Honeybee Foraging**: M. Randles et al. proposed a decentralized honeybee-based load balancing technique, It is one of a number of applications inspired by the behaviour of a colony of honeybees foraging and harvesting food. Forager bees are sent to search for suitable sources of food; when one is found, they return to the hive to advertise this using a display to the hive known as a waggle dance. The suitability of the food source may be derived from the quantity or quality of nectar the bee harvested, or its distance from the hive. This is communicated through the waggle dance display. Honey bees then follow the forager back to the discovered food source and begin to harvest it. Upon the bees' return to the hive, the remaining quantity of food available is reflected in their waggle dances, allowing more bees to be sent to a plentiful source, or exploited sources to be abandoned. This biologically-inspired technique is now used as a search algorithm in cloud computing; seeming particularly scalable on a fluctuating underlying system. As such, when applied to load balancing; as the demand for Web Services fluctuates, it is desirable to dynamically allocate servers to regulate the system against demand. The considered honeybee-based load balancing technique uses a collection of servers arranged into virtual servers, each serving a virtual service queue of requests. A profit is calculated by each server serving a request from a queue - representative of the bees measure of quality. This profit measure is calculated based on the server's cost in serving the virtual server

**Biased Random Sampling** :M. Randles et al. proposed a distributed and scalable load balancing approach that uses random sampling of the system domain to achieve self-organization. In this algorithm the load on a server is represented by its connectivity in a virtual graph. The initial network is constructed with virtual nodes to represent each server node, with each in-degree mapped to the server's free resources edges are created, connected from randomly-selected nodes. This approach creates a network system that provides a measure of initial availability status, Edge dynamics are then used to direct the load allocation procedures required for the balancing scheme. The increment and decrement process is performed via Random Sampling. The sampling walk starts at a specific node; at each step moving to a neighbor node chosen randomly. The last node in the sampling walk is selected for the load allocation.

**Active clustering**: M. Randles et al. Proposed a self-aggregation load balancing technique that is a self aggregation algorithm to optimize job assignments by connecting similar services using local re-wiring. The performance of the system is enhanced with high resources, thereby increasing the throughput by using these resources effectively. Active Clustering consists of iterative executions by each node in the network. At a random time point a node becomes an initiator and selects a matchmaker node randomly from its current neighbors; the only condition being that the matchmaker is of a different type. The matchmaker node then causes a link to be formed between one of the matchmaker's neighbors that match the type of the initiator node.

**Two-phase load balancing algorithm (OLB + LBMM)** - S.-C. Wang *et al*. proposed a two- phase scheduling algorithm that combines OLB (Opportunistic Load Balancing) and LBMM (Load Balance Min-Min) scheduling algorithms to utilize better executing efficiency and maintain the load balancing of the system. OLB scheduling algorithm, keeps every node in working state to achieve the goal of load balance and LBMM scheduling algorithm is utilized to minimize the execution time of each task on the node thereby minimizing the overall completion time. This combined approach hence helps in an efficient utilization of resources and enhances the work efficiency.

**Join-Idle-Queue (JIQ):** Yi Lu *et.al* proposed this algorithm for large-scale load balancing with distributed Dispatchers. The central idea is to decouple discovery of lightly loaded servers from job assignment. This involves idle processors informing dispatchers at the time of their idleness, without interfering with job arrivals. This removes the load balancing work from the critical path of request processing. Join-Idle-Queue load balancing algorithm for dynamically scalable web services. This algorithm provides large-scale load balancing with distributed dispatchers by, first load balancing idle processors across dispatchers for the availability of idle processors at each dispatcher and then, assigning jobs to processors to reduce average queue length at each processor. By removing the load balancing work from the critical path of request processing, it effectively reduces the system load, incurs no communication overhead at job arrivals and does not increase actual response time.

**A Lock-Free Solution for Load Balancing**: Liu et al. proposed a lock-free multiprocessing load balancing solution that avoids the use of shared memory in contrast to other multiprocessing load balancing solutions which use shared memory and lock to maintain a user session. It is achieved by modifying Linux kernel. This solution helps in improving the overall performance of load balancer in a multi-core environment by running multiple load-balancing processes in one load balancer. By modifying Linux kernel

a new socket option named sock level hash is added for listening socket and provide a lock-free multiprocessing load balancing solution. When enable sock level hash option, kernel will always forward the requests from the same client to the same process by using hash function. This avoids the using of shared memory and lock when building multi-process load balancing system

**Server based load balancing for Internet distributed services** : Massaru Nakai, *et.al* proposed a New load balancing solution that reduces service response times by redirecting requests to the closest remote servers without overloading them. The load balancing policy runs in the server side. Any overloaded server replica can redirect requests to other replicas. It helps in reducing the service response times by using a protocol that limits the redirection of requests to the closest remote servers without overloading them. Solution avoids this problem by combining a new strategy based on limited rates of request redirection and a heuristic that helps web servers to tolerate overload caused by abrupt load peaks and/or partial failures.

**Ant colony algorithm**: Z. Zhang *et al* .Proposed a load balancing mechanism based on ant colony and complex network theory in an open cloud computing federation. Acting as a collective, ants manage to perform a variety of complicated tasks with great reliability and consistency. A colony of ants can collectively perform useful tasks such as building nests, and foraging. Drawing upon some of the computing techniques inspired by social insects such as ants , several mobile agent-based paradigms were designed to solve load balancing problems in distributed systems. Montresor *et al* have developed an inverse artificial ants system, where artificial ants disperse a group of tasks evenly on idle nodes. In the SearchMax process, an ant wanders across the network to look for an overloaded Node, then, in the SearchMin process, the ant wanders across the network to look for an under loaded node, finally, the ant transfers a task from the most overloaded node to the most under loaded one. This way it is used for load balancing in cloud computing environment.

## Conclusion

Load balancing is one of the main challenges in cloud computing. It is required to distribute the dynamic local workload evenly across all the nodes to achieve a high user satisfaction and re-source utilization ratio by making sure that every computing re-source is distributed efficiently and fairly. With proper load balancing, resource consumption can be kept to a minimum in this study the. Existing load balancing techniques that have been discussed mainly focus on reducing associated overhead, service response time and improving performance etc.

## References


Rade Stanojevi´c (2009), Load balancing vs. distributed rate limiting: an unifying framework for cloud control, Robert Shorten Hamilton Institute, NUIM, Ireland , *IEEE ICC proceedings*

Abhay Bhadani, Sanjay Chaudhary, Dhirubhai Ambani Institute of Information and Communication Technology, Performance Evaluation of Web Servers using Central Load Balancing Policy over Virtual Machines on Cloud

H Mehta, P Kanungo, M Chandwani (2011) , Decentralized Content Aware Load Balancing Algorithm for Distributed Computing Environments *International Conference and Workshop on Emerging Trends in Technology* (ICWET 2011) – TCET, Mumbai, India

Martin Randles, Enas Odat, David Lamb, Osama Abu-Rahmeh and A. Taleb-Bendiab (2009), A Comparative Experiment in Distributed Load Balancing , Second *International Conference on Developments in eSystems Engineering*.

Join-Idle-Queue: A Novel Load Balancing Algorithm for Dynamically Scalable Web Services Yi Lua, Qiaomin Xie, Gabriel Kliot, Alan Geller, James R. Larus, Albert Greenberg.

Xi Liu; Lei Pan; Chong-Jun Wang; Jun-Yuan Xie (2011), A Lock-Free Solution for Load Balancing in Multi-core Environment, *IEEE*.

Alan Massaru Nakai, Edmundo Madeira, and Luiz E. Buzato (2011), Load Balancing for Internet Distributed Services using Limited Redirection Rates , Latin-*American Symposium on Dependable Computing*.

Shu-Ching Wang, Kuo-Qin Yan , Wen-Pin Liao and Shun-Sheng Wang (2010), Towards a Load Balancing in a Three-level Cloud Computing Network 978-1-4244-5539-/10 / IEEE

Zehua Zhang and Xuejie Zhang (2010), A Load balancing Mechanism Based on Ant Colony and Complex Network Theory in Open Cloud Computing Federation, *2nd International Conference on Industrial Mechatronics and Automation*,

Zhong Xu, Rong Huang, Performance Study of Load Balancing Algorithms in Distributed Web Server Systems, CS213 Parallel and Distributed Processing Project Report.

Rahmawan, H. The simulation of static load balancing algorithms Electrical Engineering and Informatics, 2009. ICEEI '09. International Conference on (Volume:02 )

Soumya Ray and Ajanta De Sarkar (2012), Execution Analysis Of Load Balancing Algorithms In Cloud Computing Environment, International Journal on Cloud Computing: Services and Architecture (IJCCSA),Vol.2, No.5, October 2012 DOI : 10.5121/ijccsa.2012.2501 1

Abhijit A. Rajguru, S.S. Apte (Aug 2012), A Comparative Performance Analysis of Load Balancing Algorithms in Distributed System using Qualitative Parameters , *International Journal of Recent Technology and Engineering (IJRTE)*, Volume-1, Issue-3, August 2012