ISSN:2231-0711

Bhawana Lakhani et al | IJCSET(www.ijcset.net) | 2018 | Volume 8 Issue 4, 56-60

Pros and Cons of Load Balancing Algorithms for Cloud Computing

Bhawana Lakhani ,Amit Agrawal
Medicaps University ,India.

Abstract:
Cloud Computing is growing rapidly and clients are demanding more services and better flexibility. For providing user demands, cloud computing require effective load balancing techniques in computing environment. Load Balancing is essential for efficient operations in distributed environments and it has become a very interesting and important research area. Many task scheduling algorithm has provided to enhance the overall performance of the cloud environment and provide users the more efficient services. In this paper, the different algorithms are studied which are used for resolving the issue of load balancing and task scheduling in Cloud Computing and also discussed pros and cons of the algorithms to provide an overview of the latest approaches in the field.

Keywords—cloud computing, load balancing, task scheduling, cloud storage.

INTRODUCTION

In the last few years Cloud Computing became very popular. It provides a flexible and easy way to keep and receive the cloud services. It makes a large data sets and files available for the spreading number of users around the world. The main aim of cloud computing is to provide the satisfactory level of performance to the user. In cloud computing there are various technique to handle the large services and operations perform on it. To improve the performance of the user operations and storage utilization, it is important to research some areas in the cloud computing. One important issue associated with this field is load balancing or task scheduling. There are various algorithms for the load balancing which are used in various environments. The main aim of the load balancing algorithm is to efficiently assigning task to the cloud nodes such than the response time of the request is minimum and request processing is done efficiently (Klaithem Al Nuaimi, 2012)

In cloud computing various additional challenges are present as compared to other environment such as heterogeneity and high communication delay. In load balancing algorithms, it classified as static and dynamic algorithms. Static algorithms are mostly used for homogeneous and stable environments. It can produce very good results in stable environments. However, static algorithms are not flexible and cannot accept the changes of attributes during execution time. Dynamic algorithms are more flexible in dynamic computing environments. It take into consideration different types of attributes in the environment both prior to and during run-time. (J. Srinivas, July 2012)

These algorithms can consider all the changes and provide better results in heterogeneous and dynamic environments. However, when these algorithm consider all changes during runtime it become more complex and dynamic to handle. At some condition, such algorithm results into the more decreasing performance of the services. In this paper, we present a survey of the current load balancing algorithms developed specifically for the Cloud Computing environments. We provide an overview of these algorithms and discuss their properties. In addition, we compare these algorithms based on the following properties: spatial distribution of cloud node, algorithm complexity, storage of data, and point of failure in algorithm.

The rest of this paper is organized as follows. We discuss we the current literature and discuss the algorithms proposed to solve the load balancing issues in Cloud Computing in section. After that, we discuss and compare the relevant approaches in Section III. In Section IV, we conclude the paper and show possible areas of enhancement and our future plan of improving load balancing algorithms.

LOAD BALANCING ALGORITHMS REVIEW

In this section we discuss the most known load balancing algorithm for task scheduling management in Cloud Computing. We classify the load balancing algorithms into two types: static algorithms and dynamic algorithms. We first discuss the static load balancing algorithms that has been developed for private and public Cloud Computing. Then, we will discuss the dynamic load-balancing algorithms.

A. Static Load Balancing Algorithms

Static scheduling algorithms assume all tasks arrive at the same instant of time and they are independent of the system resource’s states and their availability. The basic scheduling policies like First-Come-First-Serve and Round Robin methods are implemented in static mode. FCFS methods receive the tasks and queue them until resources are available and once they become available the tasks are allotted to them depending on their arrival time are allotted to them depending on their arrival time. No other criteria for scheduling are considered in this technique this makes it less complex in nature. (Sarkar, October 2012)

In round robin algorithm, To schedule processes fairly, a round-robin scheduler generally employs time-sharing, giving each job a time slot or *quantum* (its allowance of CPU time), and interrupting the job if it is not completed by then. The job is resumed next time a time slot is assigned to that process. If the process terminates or changes its state to waiting during its attributed time quantum, the scheduler selects the first process in the ready queue to execute. In the absence of time-sharing, or if the quanta were large relative to the sizes of the jobs, a process that produced large jobs would be favored over other processes. Issue found that it forcefully prompt the task once the time slice completed.

The proposed algorithm by Kumar (Sarkar, October 2012) is an updated version of the algorithm presented in .In both algorithms the ants' behavior is used for to gather information about the nodes in public and private cloud to assign the task to a optimal node. However, there is issue of ants' synchronization (Archana mantri, July 2011) and this issue is solved by adding the feature 'suicide' to the ants (Ranjan). In proposed algorithm (Ranjan) once a request is send from user the ants and pheromone are initiated and the ants start to move from the 'head 'node to end node .A node move from node to another node to check if it is overloaded or not. Moreover, if the ant finds an under loaded node, it will continue to move to check the next node. If the next node is an overloaded node, the ant will use the backward movement to get to the previous node. When it finds the target node the ant will commit suicide which avoids the unnecessary backward movements (Ranjan).

The algorithm proposed in (S.Nagadevi1, 2013) is called as the Map Reduce algorithm [12]. MapReduce have two main tasks: It Mapping tasks and Reduces tasks results. Moreover, three methods are included in this model. The three methods are part, comp and group. In MapReduce the part method is initiate the Mapping of tasks. In this step the request is partitioned into different parts. Then, the key of each part is saved into a hash key table. The comp method does the comparison between the parts. The next method is group method in which it groups the parts of similar entities using the Reduce tasks. But it should have overloaded problem. For reducing problem one more level should be added called as load balancing level which reduce task to decrease the overload on these tasks.

Junjie proposed a load balancing algorithm (Upendra Bhoi, April 2013) is based on mapping in between the virtual machine to physical machine in public and private cloud computing. The algorithm includes a central scheduling controller and a resource monitor for calculating the optimal node. The scheduling controller is used for calculating available resources and assigning the task to specific resource. However, the resource monitor is used for collecting the details about the resources availability. The algorithm also include four main phases for assign task which are: accepting request of virtual machine, then getting the resources details using the resource monitor. After that, the controller calculates the resources ability to handle tasks and the resource that gets the highest score is the one receiving the task. Finally, the client will be able to access the application.

B. Dynamic Load Balancing Algorithms

Dynamic Load Balancing Algorithms is more accurate and more efficient load balancing techniques. Dynamic load balancing algorithm includes the node capabilities and network bandwidth. Dynamic load balancing algorithm depends on combination of knowledge based on all gathered information about the nodes and different properties of the selected nodes process and task on that node in public and private cloud. By using gathered information and calculation dynamic load balancing algorithm assign the task and for some condition, it should be reassign them. Some dynamic load balancing algorithms require the status of the node and task current situation and progress .Such algorithms are usually harder to implement. Kumar Nishant suggested an algorithm (S.Nagadevi1, 2013) of ant colony optimization. In ACO (S.Nagadevi1, 2013) algorithm when the request in initiated the ant start its movement.

Movement of ant is of two ways:

Forward Movement:

Forward Movement means the ant in continuously moving from one overloaded node to another node and check it is overloaded or under loaded, if ant find an over loaded node it will continuously moving in the forward direction and check each nodes.

Backward Movement: If an ant find an over loaded node the ant will use the back ward movement to get to the previous node, in the algorithm if ant finds the target node then ant will commit suicide, this algorithm reduced the unnecessary back ward movement ,overcome heterogeneity, is excellent in fault tolerance.

Genetic algorithm (Buyya, May 2007) is also a nature-inspired algorithm. Pop et al. (Teng, 2011)modify it, to make it a reputation-guided algorithm. They evaluated their solution by taking load balancing as a way to calculate the optimization offered to providers and makes pan as a performance metric for the user. Another such algorithm is the Bees Life Algorithm (BLA), (Gan Guo-ning, 2010) which is inspired by bee's food searching and reproduction. This concept is further extended to specifically address the issue of load balancing in [30]. The Honeybee behavior inspired load balancing (HBBLB) algorithm manages the load across different virtual machines for increasing throughput. Tasks are prioritized so that the waiting time is reduced when they are aligned in queues. The honey bee foraging behavior and some of its variants are listed in (Li, 2011).

Sang proposed OLB is a static load-balancing algorithm that has the goal of keeping each node in cloud busy (Kun Li, 2011) . However, OLB does not calculate the execution time of the node, due to this the tasks to be processed in a slower manner and will cause bottlenecks since requests might be pending waiting for nodes to be free. Wang suggested an algorithm called LBMM (Laiping Zhao, 2011). LBMM has a three level load-balancing framework. In first level LBMM, architecture is the request manager, which is responsible for receiving the task and assigning it to service manager, when the service manager receives the request; it divides it into subtask and assigns the subtask to a service node based on node availability, remaining memory and the transmission rate, which is responsible for execution the task. M. Randles et al. (Chia-Ming Wu, 2013) investigated a decentralized honeybee-based load balancing technique that is a nature-inspired algorithm for self-organization.

It achieves global load balancing through local server actions. Performance of the system is enhanced with

increased system diversity but throughput is not increased with an increase in system size. It is best suited for the conditions where the diverse population of service types is required.M. Randles et al. (Yujia Ge, 2010) investigated a distributed and scalable load balancing approach that uses random sampling of the system domain to achieve self-organization thus balancing the load across all nodes of the system. The performance of the system is improved with high and similar population of resources thus resulting in an in-creased throughput by effectively utilizing the increased system resources. The paper in (Chenhong Zhao, 2009) (Simsy Xavier, Feburary 2013)proposes an algorithm called Load Balancing Min-Min (LBMM) based on the Opportunistic Load Balancing algorithm (OLB) (Chia-Ming Wu R.-S. C.-Y., 2013) (Xiaonian Wu, 2013).
OLB is a static load-balancing algorithm that has main aim to keep each node busy in the cloud. It does not consider the execution time of the node is the disadvantage of the OLB. LBMM improves OLB by considering a three-layered architecture to the algorithm. In the first level the request manager is receive the task and assign it to one service manager in the second level of LBMM. When the service manager accept the request, it divides it into subtasks to increase the processing that request. A service manager assign the subtask to a service node which is required for executing the task by using different attributes such as the remaining CPU space (node availability), remaining memory
Minimum Execution Time and Minimum Completion Time are other two heuristic approaches: in which MET maps tasks to machine depending on which machine takes less execution time and assigns it on the machines. This approach suffers from load imbalance as it selects the best machine for execution but avoids considering the availability of resources at the time of scheduling. Minimum Completion Time Algorithm chooses machines with minimum expected completion time of tasks among all the available machines. Machine examined for load before scheduling of task on that machine. Any two tasks cannot have minimum execution time on the same machine. Completion time of a task on a machine can be characterized as the aggregate of the execution time of the task on that machine and the ready time of that specific machine.

COMPARATIVE ANALYSIS OF ALGORITHMS

In this section, the different algorithms are discus and compare these algorithms based on the challenges such as spatial distribution of cloud node, algorithm complexity, storage of data, and point of failure in algorithm. As discussed earlier, there are various approaches for balancing the load on node for various situations. The static algorithms are usually very efficient in stable environment because they do not need to monitor the resources during runtime. In a stable environment, operational properties do not change over time and loads are generally uniform and constant at the running time. On the other hand, The Dynamic algorithms continuously monitor the resources at run time. Itoffers a much better solution that to adjust the load dynamically at run-time. The Dynamic algorithms mostly work on the observed properties of the resources at run time. However, this feature leads to high overhead on the system as constant monitoring and control will add more traffic and may cause more delays. Some newly proposed dynamic load balancing algorithms tries to avoid this overhead by utilizing novel task distribution models.

Algorithm	Pros	Cons
MAX-MIN	Initially proved to handlesome sort of dynamic load balancing	1. Complicated in terms of implementation. 2. Only certain parameters are considered such as distance and time complicated 3. No forecasting algorithm to identify the future behavior of the nodes.
HBB	Better throughput	Prediction algorithm requires existing data and has long processing time
OLB+LBMM	Better resource utilization	Inherits Round Robin issues such as not taking into consideration node

Table I. Pros and cons of load balancing algorithms

Table I shows a comparison among the reviewed various algorithms. The comparison shows advantages and disadvantages of each algorithm. For example, the ACO algorithm is a dynamic load-balancing algorithm. However, the provided algorithm is complicated to implement which could cause high implementation complexity.
To avoid such complexity, some changes may do in the structure, which results into a less complex algorithm. Furthermore, the CLDBM algorithm has a human administrator, which used to control the system. Therefore, the provided algorithm included a centralized controller. However, the centralized controller has a main importance. If the centralized controller fails any time the whole system will not be able to operate which will cause a system failure. The issue of failure in the CLDBM algorithm can solved to making a backup of the central controller. As for the PSO is a computational method with regard to a given measure quality. Its solves problem by having population of candidate solution. PSO is a metaheuristic as it makes few or no assumptions about the problem being optimized and can search very large spaces of candidate solutions.

Management	Automated tasks forwarding reduces the need for a human administrator	1. Single point of failure (if its fail, the whole process fails) 2. The threshold might not be applied to all cases.
ANT COLONY	1. Best-case scenario is that the under loaded node is found at beginning of the search. 2. Decentralized, no single point of failure 3. Ants can collect the information faster	1. Network overhead because of the large number of ants 2. Points of initiation of ants and number of ants are not clear 3. Nodes status

		change after ants visits to them is not taken into account 4. Only availability of node is being considered, while there are other factors that should be taken into consideration
MIN-MIN	More performance and improved effiency	1 .High processing time 2. Reducetasks capabilities are not taken into consideration
BRS (Biased Random Sampling)	Reliable calculation method	1. Single Point of failure 2. Does not take into account network load, and node capabilities
DDFTP	1. Fast Calculation 2. Reliable download of files	Full replication of data files that requires high storage in all nodes.
LBMM	Reliable tasks assignment to nodes	Slower than other algorithms because Work must pass through three layers to be processed.

In DDFTP, there is no centralized control and no run-time monitoring of node and its resources, which keep it, has a very efficient load-balancing algorithm. It provides a good approach, yet it still needs some improvements for better utilization of the available node and resource. For a better performance, it will reduce the level of replication, while maintaining the same level of performance. This may be possible with the consideration of partial replications with a certain level of overlap that will enable more efficient resource utilization and maintain minimum overhead for load balancing.

In HTV, it continuously monitoring the resources and finding the result using current information. More than two controllers are used in the provided algorithm. For the better performance of the algorithm, it uses two or more than two parameter and controller should be minimum number.

CONCLUSION

In this paper, different algorithms are studied for load balancing and discussed their pros and cons. Then, the exiting algorithms are compared on the basis of their challenges which are present in cloud environment. ESWLC concentrates on efficient load balancing and provides accurate results. CLDBM and Ant colony algorithm reduces need of the human administration and provide the faster information. LBMM and HTV algorithms use less response time for calculation and more efficient in terms of resource parameter utilization.

REFERENCES

[1] 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.

[2] Rimal, B. Prasad, E. Choi and I. Lumb, "A taxonomy and survey of cloud computing systems." In proc. 5th International Joint Conference on INC, IMS and IDC, IEEE, 2009.

[3] Buyya R., R. Ranjan and RN. Calheiros, “InterCloud: Utility-oriented federation of cloud computing environments for scaling of application services,” in proc. 10th International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP), Busan, South Korea, 2010.

[4] Foster, I., Y. Zhao, I. Raicu and S. Lu, “Cloud Computing and Grid Computing 360-degree compared,” in proc. Grid Computing Environments Workshop, pp: 99-106, 2008.

[5] Grosu, D., A.T. Chronopoulos and M. Leung, "Cooperative load balancing in distributed systems," in Concurrency and Computation: Practice and Experience, Vol. 20, No. 16, pp: 1953-1976, 2008.

[6] Ranjan, R., L. Zhao, X. Wu, A. Liu, A. Quiroz and M. Parashar, "Peer- to-peer cloud provisioning: Service discovery and load-balancing," in Cloud Computing - Principles, Systems and Applications, pp: 195-217, 2010.

[7] Radojevic, B. and M. Zagar, "Analysis of issues with load balancing algorithms in hosted (cloud) environments." In proc.34th International Convention on MIPRO, IEEE, 2011.

[8] Sotomayor, B., RS. Montero, IM. Llorente, and I. Foster, "Virtual infrastructure management in private and hybrid clouds," in IEEE Internet Computing, Vol. 13, No. 5, pp: 14-22, 2009.

[9] Nishant, K. P. Sharma, V. Krishna, C. Gupta, KP. Singh, N. Nitin and R. Rastogi, "Load Balancing of Nodes in Cloud Using Ant Colony Optimization."In proc. 14th International Conference on Computer Modelling and Simulation (UKSim), IEEE, pp: 3-8, March 2012.

[10] Zhang, Z. and X. Zhang, "A load balancing mechanism based on ant colony and complex network theory in open cloud computing federation." In proc. 2nd International Conference on. Industrial Mechatronics and Automation (ICIMA), IEEE, Vol. 2, pp:240-243, May 2010.

[11] Kolb, L., A. Thor, and E. Rahm, E, "Load Balancing for MapReduce- based Entity Resolution," in proc. 28th International Conference on Data Engineering (ICDE), IEEE, pp: 618-629, 2012.

[12] Gunarathne, T., T-L. Wu, J. Qiu and G. Fox, "MapReduce in the Clouds for Science," in proc. 2nd International Conference on Cloud Computing Technology and Science (CloudCom), IEEE, pp:565-572, November/December 2010.

13. Archana mantri, S. N. (July 2011). *High Performance Architecture and Grid Computing Computing.* Chandigarh: International Conference HPAGC 2011.

14. Buyya, J. Y. (May 2007). *Workflow Scheduling Algorithms for Grid Computing, Technical Report, GRIDS-TR-2007-10, Grid Computing and Distributed Systems Laboratory.* The University of Melbourne.

15. Chenhong Zhao, S. Z. (2009). *Independent Tasks Scheduling Based on Genetic Algorithm in Cloud Computing.* IEEE.

16. Chia-Ming Wu, R.-S. C.-Y. (2013). *A green energy-efficient scheduling algorithm using the DVFS technique for cloud datacenters.* Elvister.

17. Gan Guo-ning, H. T.-l. (2010). *Genetic simulated annealing algorithm for task scheduling based on cloud computing environment.* IEEE.

18. J. Srinivas, K. a. (July 2012). *Cloud Computing Basics.* International Journal of Advanced Research in Computer and Communication Engineering.
19. Klaithem Al Nuaimi, N. M. (2012). *A Survey of Load Balancing in cloud Computing.* IEEE Second Symposium on Network Cloud Computing.
20. Kun Li, G. X. (2011). *Cloud Task Scheduling Based on Load Balancing Ant Colony Optimization.* IEEE.
21. Laiping Zhao, Y. R. (2011). *A Resource Minimizing Scheduling Algorithm with Ensuring the Deadline and Reliability inHeterogeneous Systems.* IEEE.
22. Li, G. M. (2011). *An Improved Algorithm Based on MaxMin for Cloud Task Scheduling.* Yunnam University, China.
23. Ranjan, R. L. (s.d.). *Peer to- peer cloud provisioning: Service discovery and load-balancing," in Cloud Computing - Principles, Systems and Applications, pp: 195-217,201010.*
24. S.Nagadevi1, K. D. (2013). *A Survey On Economic Cloud Schedulers For Optimized Task Scheduling.* International Journal of Advanced Engineering Technology.
25. Sarkar, S. R. (October 2012). *Execution Analysis of Load Balancing Algorithms in Cloud Computing Environment.* International Journal on Cloud Computing Services and Architecture.
26. Simsy Xavier, S. L. (Feburary 2013). *A Survey of Various Workflow Scheduling Algorithms in Cloud Environment.* International Journal of Scientific and Research Publications, Volume 3, Issue 2 ISSN 2250-3153.
27. eng, F. (2011). *Resource allocation and scheduling models for cloudcomputing.* paris.
28. Upendra Bhoi, P. N. (April 2013). *Enhanced Max-min Task Scheduling Algorithm in Cloud Computing.* International Journal of Application or Innovation in Engineering & Management, Volume 2, Issue 4 pp 259-264.
29. Xiaonian Wu, M. D. (2013). *A Task Scheduling Algorithm based on QoS driven in Cloud Computing.* Information Technology and Quantitative management.
30. Yujia Ge, G. W. (2010). *GA-Based Task Scheduler for the CloudComputing Systems.* IEEE.