International Journal of Electrical and Computer Engineering (IJECE)
Vol. 8, No. 3, June 2018, pp. 1741~1746
ISSN: 2088-8708, DOI: 10.11591/ijece.v8i3.pp1741-1746

A Prolific Scheme for Load Balancing Relying on Task Completion Time

V. Anand[1], K. Anuradha[2]
[1]School of Computing, SASTRA Deemed University, Tirumalaisamudram, Thanjavur, India
[2]School of Electrical and Electronics Engineering, SASTRA Deemed University, Tirumalaisamudram, Thanjavur, India

Article Info

Article history:

Received Aug 28, 2017
Revised Dec 28, 2017
Accepted Jan 4, 2018

Keyword:

Computation time
CPU load
Load balancing
Network load
Task subtraction

ABSTRACT

In networks with lot of computation, load balancing gains increasing significance. To offer various resources, services and applications, the ultimate aim is to facilitate the sharing of services and resources on the network over the Internet. A key issue to be focused and addressed in networks with large amount of computation is load balancing. Load is the number of tasks‘t’ performed by a computation system. The load can be categorized as network load and CPU load. For an efficient load balancing strategy, the process of assigning the load between the nodes should enhance the resource utilization and minimize the computation time. This can be accomplished by a uniform distribution of load of to all the nodes. A Load balancing method should guarantee that, each node in a network performs almost equal amount of work pertinent to their capacity and availability of resources. Relying on task subtraction, this work has presented a pioneering algorithm termed as E-TS (Efficient-Task Subtraction). This algorithm has selected appropriate nodes for each task. The proposed algorithm has improved the utilization of computing resources and has preserved the neutrality in assigning the load to the nodes in the network.

Corresponding Author:

V. Anand,
School of Computing,
SASTRA University,
Tirumalaisamudram, Thanjavur 613401,
Tamil Nadu, India.
E-Mail: anandwithah@gmail.com

1. INTRODUCTION

An important issue in networks is load balancing. The load can be balanced dynamically or statically. In some situations, the load on a particular node cannot be foreseen [1]. This remains a key issue specifically in distributed networks. *Load Balancing* is a method of dispensing the requests to many servers or resources. This helps in enhancing the performance, efficient utilization of resources, avoids overloading, reduced response time and improved throughput.

In a distributed network, it not cost effective to have idle servers when there are multiple client requests. The presence of inactive or idle servers may result in inefficient utilization of resources. The same server cannot be overloaded by assigning all the client requests to the same server. An efficient load balancing algorithm should distribute the load among the servers in a balanced manner.

A distributed network is a combination of different types of networks. An emerging distributed computing technology is Cloud. Cloud computing ensures elasticity and scalability for operating and maintaining infrastructure, platform and software. With cloud computing services, client can scale up or scale down the utilization of resources relying on their demand. In cloud computing, resources are shared between different systems. This needs a method for allocating the tasks to systems. The tasks include handling of requests, make the data available to every system and process the response sent by the different processors

(systems). Resources can be managed efficiently with an effective algorithm for load balancing. An ineffective load balancing algorithm may end up in execution delay.

The paper is organized as follows: Section 2 offers an insight into cloud computing with deployment and service models; Section 3 presents the techniques and algorithms for load balancing with load balancers, scheduling algorithms and Map Reduce; Section 4 proposes the method E-TS with the algorithm; Section 5 presents and compares the results obtained; Section 6 concludes the work.

2. AN INSIGHT INTO CLOUD COMPUTING

Cloud computing can be defined as an on-demand web based service, in which shared resources are used for executing a task to obtain the outcome in least probable time. This can be done by sharing a dataset between all the linked processors. To enable the implementation of complex tasks with large-scale computation, cloud computing can be used for proper utilization of computing resources on the network. To accomplish efficiency, nodes for the tasks should be chosen based on the properties of the task [2].

Using virtual servers, cloud computing offers network-based services to clients in different locations. Virtual instances (i.e) servers, storage or other network resources can be created with the help of virtualization technologies. Tasks which involves big datasets, can be partitioned into multiple small datasets. Then, the small datasets can be assigned to numerous processors so that the tasks can be completed in least feasible time.

A cloud has a cluster of nodes with a specific configuration and infrastructure, which provides services requested by clients. A processing unit or node is identified as a master. All the nodes are linked to the master. The master allocates the tasks to the other nodes connected to it. A node designated as head node in the cloud, accepts the tasks. Then, the head nodes decomposes the tasks into sub tasks and allocates it back to the master. The master assigns the sub tasks to the nodes linked to it. To complete the tasks in the least possible time, the nodes share the datasets.

2.1. Deployment models of cloud

A cloud deployment model embodies a particular type of cloud setup, mainly differentiated by access, size and ownership. Models of cloud computing [3] broadly falls into three categories, viz.: Public, Private and Hybrid Cloud.

A Public cloud is accessible publicly and may be owned by one or more organizations (Third-party service providers). In a public cloud, the infrastructure and resources are provisioned to consumers through cloud delivery models.

Private cloudis owned and being operated by a private organization. In a private cloud, the systems, storage and other network resources are accessible to the members of the organization. The owner (i.e) the organization has the authority for maintaining the infrastructure of the cloud.

A hybrid cloudis a mix of deployment models. A consumer can maintain a portion of the cloud infrastructure (with sensitive data) in the organization and consumes services(less sensitive) from the public clouds.

Deployment models have merits and demerits as well. The main benefit with a Public cloud is, the service provides is responsible for operations, maintenance and services in the cloud. The problem with a public cloud is, consumers depends totally on the service provider for accessing the infrastructure and other resources. Private cloud enables users to govern the infrastructure and resources, since it is owned by the organization itself. Hence, the users can consume the infrastructure based on their varying demand. The key issue is, organization has to invest lot of funds for setting up a private cloud, operations and maintenance. In a Hybrid cloud, deploying an architecture is an intricate task for two reasons: (a) because of the probable differences incloud environments, (b) duties of resource management are shared among the public cloudprovider and the private cloud provider (i.e) the organization itself.

2.2. Service models in cloud

Service models in cloud vary based on the requirements of customers. Various service models in the cloud are SaaS (Software as a Service), PaaS (Platform as a Service) and IaaS (Infrastructure as a Service).

Software as a service is a model which provides software services to the customers. In SaaS, customers are not permitted to alter the the applications. Platform as a Service model lets the users to use Application Program Interfaces (API) in various languages. This enables the users to develop and customize the applications for their requirements.

Infrastructure as a Service model allows customers to use and customize the storage and other computing resources pertinent to their needs. This model allows the customers to scale up or down the computing resources based on their demand.

3. TECHNIQUES AND ALGORITHMS FOR LOAD BALANCING

Two kinds of algorithms are available for load balancing. They are static and dynamic. They are differentiated by the decisions that are taken. Sometimes, decisions rely on the present state of the system (dynamic) or always the same (static) [4]. The proposed work has studied the static methods presented in [5]-[7] and dynamic presented in [8]-[14]. In distributed methods, the nodes communicate with each other through a non-cooperative or co-operative scheme [15]. The distributed scheme proposed in [15] reveals that each node in a system execute the algorithm and also share the responsibility of load balancing.

An algorithm named as LB3M presented in [16], has combined load balancing methods and completion time. By maintaining the load in a balanced state, LB3M has offered effective utilization of resources in a cloud computing setup.

When a query is received on a node, it is sent to all the nodes in the cloud. This kind of distribution should be done in such a way that reduces the waiting time for the response from the nodes. Taking into account the cloud environment, assessment of execution time becomes necessary with devices of different types. Some of the significant factors should be considered for task scheduling on a network with many users and heterogeneous devices. These factors include heterogeneity, network usage and task execution time [17].

Asymmetric load distribution is a key feature in a load balancing method. Based on the computational efficiency, high amount of workload may be allocated to a node. But, allocation of tasks should not be done only based on the computational efficiency of a node. In this context, a load balancing algorithm can be applied for allocating tasks to nodes in a heterogeneous setup like a cloud. A work proposed in [18] offers a competent local search method for scheduling tasks in heterogeneous computing setup. The authors claim that, this methods obtains schedules of short duration.

An improved dynamic load balancer presented in [19] has obtained good results. Through the results, the implementation has demonstrated an improved resource utilization, performance and load balancing.

To preserve the performance of computing, a load balancing algorithm with 'Priority Activation' can be used. That is, allocate the tasks to nodes with less priority when nodes with high priority are down beyond a certain level.

An important aspect ofload balancing is persistence. It is about handling information across many requests in a session. All the nodes cannot access the data if it is stored on a particular server. So, the successive request to other nodes consumes some space. This becomes an issue related to computing performance and has to be addressed. This issue can be solved by sending the requests to the same node ‘n’ with the infrastructure and resources. But, this becomes a centralized approach. If the particular node ‘n’ with the data and other infrastructure fails, the sessions running on the node with the processes are also lost. This can be resolved by having a backup of the node ‘n’.

Another solution is, application of databases. The databases can be used for storing the resources. By taking a backup of databases, consequences of node failures can be resolved. By maintaining a backup, a better performance can be attained by distributing the requests to nodes with similar data and infrastructure.

3.1. Design of load balancing methods

Load balancing can done with lot of hardware and software available with different vendors. Some of them are Pound Reverse Proxy, nginx and Apache mod_proxy_balancer. To allocate large tasks to many nodes, Gearman can be used. This results in minimizing the completion time of tasks.

To produce a tree like structure, many layers of load balancing can be done. The higher level load balancing mechanism dispenses the task to computing nodes in the next level. These nodes apply their load balancers to allocate the tasks to nodes in the next level. This may end up in many levels depending on the size of the cloud (nodes and infrastructure).

3.2. Algorithms for scheduling

Numerous algorithms for scheduling the tasks are available. Based on factors such as computational efficiency and completion time, these algorithms allocate the tasks to various nodes. One of the algorithms is Round Robin

Scheduling Load Balancing algorithm. In this algorithm, the incoming requests are dispensed in a sequence to a set of servers. This algorithm considers all the servers equally regardless of their load or capacity. This algorithms needs all the servers with similar configuration since a server with a lesser configuration (weak server) may be overloaded while receiving multiple requests. This will affect the performance.

Many scheduling algorithms are available, which performs load balancing based on the following factors: active/inactive status of a node, number of active links, server’s load, server’s location, response time and the traffichandled recently.

The work presented in [20] has explored various algorithms for task scheduling based on quality of service (QOS), which enhances the efficiency by considering several parameters viz.: accepted rate, fairness completion time, cost, make span and minimum completion time. An algorithm termed as Extended Non-preemptive PP-aware scheduling (ENPP) was proposed in [21]. Relying on non-exclusive scheduling, the algorithm has scheduled the tasks by stopping the tasks with the highest penalty in the minimum time. Through the results, the work has achieved a reduction in task processing time and amount of tasks stopped. For optimal scheduling of tasks with minimum task execution delay in a dynamic cloud computing setup, an algorithm named as OTB-CSO (Orthogonal Taguchi Based-cat swarm optimization algorithm) was proposed in [22].

MapReduce is a framework, which processes big datasets through a number of nodes in a cluster in parallel. Each node in a cluster has its own storage. Map and Reduce is being done by this framework.

First is "Map", which converts a task into sub-tasks. Then the sub-tasks are dispensed to the nodes identified as “slave” in the cluster. Sometime, a tree structure with multiple levels can be formed by including more sub-tasks. This depends on the size of the task taken into account. Processing the sub-tasks is done by “slave” nodes and the results are returned to the “master” node.Next is “Reduce”, the “master” node combines the results returned by the “slave” nodes. Mapping can be done independently and also in parallel.

4. THE PROPOSED METHOD

Cloud is a heterogeneous environment, where all the nodes cannot complete a set of tasks in the same duration. That is, completion time of tasks depends on the capability of each node. The proposed method E-TS (Efficient-Task Subtraction) assigns the tasks to nodes based on the task completion time. This method efficiently allocates the tasks to nodes by computing the subtraction of completion time of tasks. The algorithm is presented in Section 4.1.

Algorithm

Step 1: All the nodes with their completion time for each of the tasks is taken. Then, the minimum values are subtracted from the maximum values. Then, the maximum task completion time of each node is chosen.
Step 2: Choose the node with the maximum subtraction value, if the completion time is same for multiple tasks.
Step 3: A node with minimum task completion time is allocated a task.
Step 4: If the completion time is same for more than one node, add the completion time of a specific task for all the nodes. Choose the node with the maximum value.
Step 5: The task is allocated to the chosennode for processing.
Step 6: Choose the next highest task completion time. Iterate the steps 2 to 4 till all the tasks are completed.

5. RESULTS AND DISCUSSION

The proposed method E-TS (Efficient-Task Subtraction) is demonstrated with 4 nodes and 4 tasks. Each entry in the table specifies the time taken for completing the task on that node. A step-wise illustration of the algorithm is clearly presented in tabular form.

Table 1 represents the completion time of 4 tasks for 4 nodes. Table 2 presents the computation of difference in the completion time of tasks for all nodes taken into account. Table 3 shows a node ‘N_3’ with maximum subtraction value is assigned the task ‘T_1’. Table 4 shows the assignment of task ‘T_2’ to node ‘N_2’ with the next highest task completion time. Similarly, Table 5 depicts the assignment of tasks ‘T_4’ to node ‘N_1’and ‘T_3’ to node ‘N_4’.

Table 1. Task completion time for 4 tasks with 4 nodes

	Nodes			
Tasks	N_1	N_2	N_3	N_4
T_1	12	11	13	9
T_2	23	15	24	12
T_3	29	24	31	11
T_4	23	16	30	16

Table 2. Difference in the completion time of tasks for all nodes

	Nodes			
Tasks	N_1	N_2	N_3	N_4
T_1	12	11	13	9
T_2	23	15	24	12
T_3	29	24	31	11
T_4	23	16	30	16
Result of Subtracting the maximum and minimum completion time	17	13	18	7

Table 3. Choose the node with maximum subtraction value

	Nodes			
Tasks	N_1	N_2	N_3	N_4
T_1	12	11	13	9
T_2	23	15	24	12
T_3	29	24	31	11
T_4	23	16	30	16
Result of Subtracting the maximum and minimum completion time	17	13	18	7

Table 4. Chooses the node with the next highest task completion time

	Nodes			
Tasks	N_1	N_2	N_3	N_4
T_1	12	11	**13**	9
T_2	23	**15**	24	12
T_3	29	24	31	**11**
T_4	**23**	16	30	16
Result of Subtracting the maximum and minimum completion time	17	13	18	7

Table 5. Tasks 'T_4' and 'T_3' assigned to nodes 'N_1' and 'N_4'

	Nodes			
Tasks	N_1	N_2	N_3	N_4
T_1	12	11	**13**	9
T_2	23	**15**	24	12
T_3	29	24	31	**11**
T_4	**23**	16	30	16
Result of Subtracting the maximum and minimum completion time	17	13	18	7

With regard to completion time, the results of the proposed method (E-TS) are evaluated with LBMM [2] and MM [23]. LBMM is Load Balance Min-Min method and MM is Min_min method. The results obtained for comparison are shown graphically in Figure 1. The proposed method allocates tasks to all the nodes whereas the LBMM [2] and MM [23] approaches have not allocated the tasks to some of the nodes. From the results, it is evident that the proposed method E-TS (Efficient-Task Subtraction) has accomplished an improved performance when compared with the other methods (LBMM, MM) taken into account.

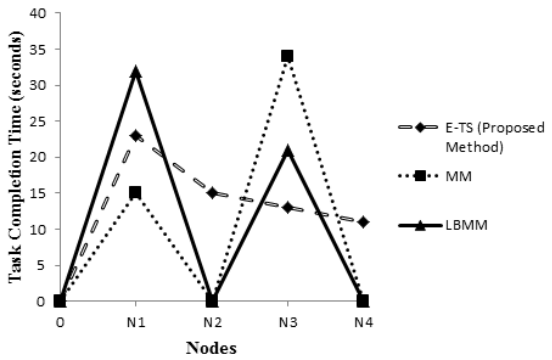

Figure 1. Comparison of E-TS (Proposed Method) with LBMM [2] and MM [23] Methods

6. CONCLUSION

Cloud computing is in the limelight and load balancing is a key area to be addressed in cloud computing. So, the proposed method E-TS (Efficient-Task Subtraction) is developed for enhancing the resource utilization and performance by maintaining a balance in allocating the load to the nodes. The proposed method E-TS (Efficient-Task Subtraction) assigns the tasks to different nodes based on the completion time. The proposed method claims its advantage by finding an appropriate node for each of the tasks to be completed. This work has accomplished the reduction in the total time taken for completion of tasks (i.e) makespan. Relying on the completion time, the proposed method has allocated the tasks to nodes. This method can be enhanced by allocating the tasks based on location of the server, server's capacity, and the amount of traffic handled by each node.

REFERENCES

[1] Y.F. Hu, R.J. Blake and D.R. Emerson, "An optimal migration algorithm for dynamic load balancing", *Concurrency - Practice and Experience*, vol. 10, no. 6, pp. 467-483, 1998.

[2] S. Wang, K. Yan, W. Liao, and S. Wang, "Towards a load balancing in a three-level cloud computing network", in *Computer Science and Information Technology (ICCSIT), 2010 3rd IEEE International Conference on*, 2010, vol. 1, pp. 108-113.

[3] C.C. Rao, M. Leelarani, and Y.R. Kumar, "Cloud: Computing Services and Deployment Models", *International Journal Of Engineering And Computer Science*, vol. 2, no. 12, pp. 3389-3390, 2013.

[4] V. Anand, N.R. Raajan, and K. Anuradha, "Significant factors in the design of an efficient dynamic load balancing algorithm: An exploration", *ARPN Journal of Engineering and Applied Sciences*, vol. 12, no. 3, pp. 841-848, 2017.

[5] X. Tang and S.T. Chanson, "Optimizing static job scheduling in a network of heterogeneous computers", in *Proceedings of the Intl. Conf. on Parallel Processing*, 2000, pp. 373-382.

[6] A.N. Tantawi and D. Towsley, "Optimal static load balancing in distributed computer systems", *Journal of the ACM (JACM)*, vol. 32, no. 2, pp. 445-465, 1985.

[7] O. Franek, "A simple method for static load balancing of parallel FDTD codes", in *2016 International Conference on Electromagnetics in Advanced Applications (ICEAA)*, 2016, pp. 587-590.

[8] J.A. Stankovic, "An adaptive bidding algorithm for processes, clusters and distributed groups", in *Proc. of 4th Int. Conf. on Distributed Computing Systems*, 1984, pp. 49-59.

[9] A. Karimi, F. Zarafshan, A. Jantan, A.R. Ramli, and M. Saripan, "A new fuzzy approach for dynamic load balancing algorithm", *arXiv preprint arXiv:0910.0317*, 2009.

[10] D.J. Evans and W.U.N. Butt, "Dynamic load balancing using task-transfer probabilities", *Parallel Computing*, vol. 19, no. 8, pp. 897-916, 1993.

[11] D.L. Eager, E.D. Lazowska, and J. Zahorjan, "Adaptive load sharing in homogeneous distributed systems", *IEEE transactions on software engineering*, no. 5, pp. 662-675, 1986.

[12] B.A. Blake, "Assignment of independent tasks to minimize completion time", *Software: Practice and Experience*, vol. 22, no. 9, pp. 723-734, 1992.

[13] A. Barak and A. Shiloh, "A distributed load-balancing policy for a multicomputer", *Software: Practice and Experience*, vol. 15, no. 9, pp. 901-913, 1985.

[14] J.A. Stankovic, "Simulations of three adaptive, decentralized controlled, job scheduling algorithms", *Computer Networks (1976)*, vol. 8, no. 3, pp. 199-217, 1984.

[15] D. Grosu and A.T. Chronopoulos, "Noncooperative load balancing in distributed systems", *Journal of parallel and distributed computing*, vol. 65, no. 9, pp. 1022-1034, 2005.

[16] C. Hung, H. Wang, and Y. Hu, "Efficient load balancing algorithm for cloud computing network", in *International Conference on Information Science and Technology (IST 2012), April*, 2012, pp. 28-30.

[17] R.F. Freund *et al.*, "Scheduling resources in multi-user, heterogeneous, computing environments with SmartNet", in *Heterogeneous Computing Workshop, 1998.(HCW 98) Proceedings. 1998 Seventh*, 1998, pp. 184-199.

[18] G. Ritchie and J. Levine, "A fast, effective local search for scheduling independent jobs in heterogeneous computing environments", *Proceedings of the 22nd Workshop of the UK Planning and Scheduling Special Interest Group (PlanSIG)*, 2003.

[19] S. Acharya and D.A.D'Mello, "Enhanced dynamic load balancing algorithm for resource provisioning in cloud", in *2016 International Conference on Inventive Computation Technologies (ICICT)*, 2016, vol. 2, pp. 1-5.

[20] S. Potluri and K. Subba Rao, "Quality of Service based Task Scheduling Algorithms in Cloud Computing", *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 7, no. 2, p. 1088, 2017.

[21] F. Hoseini, M.G. Arani, and A. Taghizadeh, "ENPP: Extended non-preemptive PP-aware scheduling for real-time cloud services", *International Journal of Electrical and Computer Engineering(IJECE)*, vol. 6, no. 5, pp. 2291-2299, 2016.

[22] D. Gabi, A.S. Ismail, A. Zainal, and Z. Zakaria, "Solving task scheduling problem in cloud computing environment using orthogonal taguchi-cat algorithm", *International Journal of Electrical and Computer Engineering(IJECE)*, vol. 7, no. 3, pp. 1489-1497, 2017.

[23] T.D. Braun *et al.*, "A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems", *Journal of Parallel and Distributed computing*, vol. 61, no. 6, pp. 810-837, 2001.