WWJMRD 2017; 3(12): 211-215
www.wwjmrd.com
International Journal
Peer Reviewed Journal
Refereed Journal
Indexed Journal
UGC Approved Journal
Impact Factor MJIF: 4.25
e-ISSN: 2454-6615

Shaik Naseera
School of Computer Science and Engineering, VIT University, Vellore, India

G.K. Rajini
School of Electrical Engineering, VIT University, Vellore, India

Santhi H
School of Computer Science and Engineering, VIT University, Vellore, India

Gayathri P
School of Computer Science and Engineering, VIT University, Vellore, India

Gopichand G
School of Computer Science and Engineering, VIT University, Vellore, India

Geraldine Bessie Amali.D
School of Computer Science and Engineering, VIT University, Vellore, India

Madhusudhana Rao
School of Information Technology, Higher college of Technology, Muscat, Oman

Correspondence:
Shaik Naseera
School of Computer Science and Engineering, VIT University, Vellore, India

Load Balancing Using Fuzzy Inference System in Distributed Computing

Shaik Naseera, G.K. Rajini, Santhi H, Gayathri P, Gopichand G, Geraldine Bessie Amali.D, Madhusudhana Rao

Abstract
It is seen that, the organizations have been interested in the decentralization of processing while achieving the integration of the information resources within their distributed systems of database, applications and users. Distributed System is a technique which allows using large number of resources where it is possible to access the resources from anywhere and anytime. Balancing the load for efficient resource utilization becomes pivotal. We propose a system of distributed load balancing using fuzzy inference system. This method doesn't require an external module to look after the load in the network it is managed by the nodes within the network. This eliminates the problem of single point of failure and becomes a more effective way of balancing the load in the network. This paper discus about load balancing in intra and inter cluster nodes of the network.

Keywords: Load Balancing, Cloud Computing, Distributed Systems, Fuzzy Inference System, Resource Utilization

Introduction
Load balancing is a strategy to share jobs between multiple PCs, system terminals or different resources, so as to obtain maximum response time or resource usage. Utilizing different units with load balancing, rather than an solitary unit, may build dependability with the help of redundancy. Load balancing keeps all the processors busy thereby increasing system throughput. It also improve the overall system execution by moving jobs from heavily loaded nodes to other light loaded nodes. The 4 strategies which represent activity of a load balancing calculations when load unevenness is found includes information, transfer, location and selection [8]. There exists two types of load balancing strategies: static and dynamic. In static, the nodes performance is resolved toward the start of execution. At that point depending on their efficiency the work is circulated at the start by the master node. The slave processors figure its allotted jobs and present its outcome to the master. The jobs are constantly run on the node to which it is allocated which means static load balancing techniques are non-preemptive [8] whereas dynamic load balancing strategy contrasts from static algorithms in that the work is circulated among the nodes at execution time. The master allocates newest process to slaves taking into account latest data collected [9, 15]. Not at all like static algorithms, dynamic algorithms allot processes progressively when very few of the processors gets to be lightly loaded. Rather, it is stored in line on fundamental host and allocated progressively upon demands from remote hosts[8]. The techniques are comprised of Central Queue Algorithm and Local Queue Algorithm[16].
Conventional techniques of the load balancing system generally use some fix qualities to recognize workload (e.g. heavily-loaded or lightly-loaded). Numerous load balancing methodologies in view of this incomplete information have been presented in the past [18]. In traditional load balancing system, resource records is important to be the input dataset, and outcome can be chose indifferently. Be that as it may, the output remains unchanged; it can't demonstrate the workload's level. Additionally, there is a sharp refinement in between members and non-members; jobs reallocation activity will be done often near the maximum limit thereby resulting in unstable system which causes pointless overhead.

Besides, the load approximation of every node is extremely troublesome and tedious.

To solve such difficulties, we propose clustered based approach for load balancing which uses fuzzy logic. In our proposal different nodes are grouped together to form different clusters. Then using fuzzy logic we evaluate the workload of each node within a cluster (intra-cluster) using queue length and CPU utilization as the input parameters for fuzzy approach and present an arrangement of membership function. Our proposed algorithm also uses Gossip protocol and cost factor for inter cluster load balancing.

The work is organized as follows: In Section II the review of the related work for the domain is given. Section III discusses about our proposed approach and techniques. Section IV deals with simulated results and discussion of the experiment. Section V is the conclusion part.

Related work

Fuzzy Inference System (FIS)

It is the technique to map functions from the available input data to corresponding outcome data using methods of fuzzy. It consists of inputs and their membership functions; output and its membership functions; and laws for the memberships.

It was designed by L. A. Zadeh.

There exist three models of inference system of fuzzy logic:

- Mamdani fuzzy inference model.
- Tsukamoto fuzzy inference model.
- Sugeno fuzzy inference model.

These three models of fuzzy inference systems differ from each other in terms of output generation. Mamdani model is the most widely used amongst all the available inference systems. It was the first model and it was designed by Ebrahim Mamdani in 1975 [1].

The steps involved in the Mamdani Inference system are:

- Normalize the given input variables as required.
- Fuzzification of inputs.
- Calculating the membership function value for input variables.
- Determining output fuzzy descriptors using the rule base of *If … then rules.*
- Apply defuzzification logic to calculate the crisp output.
- De-Normalize the output value if required.

Gossip Protocol

Gossip protocols is used to tackle different issues in distributed systems in particular: consistency management in replicated databases [2], publish subscribe [4], failure detection [3] and application level reliable broadcast [5] [6].

In this method all members in the protocol work together, in the similar way, to disseminate data. So whenever a node wants to telecast message, it chooses t nodes indiscriminately and determine its target and telecast message to them. After accepting a message initially, a node follows the same procedure.

In the event that node gets the similar message two times - that is conceivable, as every node chooses its gossip focuses in an autonomous manner it just rejects the message. Permitting this, every node needs to stay informed regarding which messages it has as of now seen. Without deleting, the arrangement of message identifiers may rise persistently while protocol is executing.

There exist two major factors in association with the setup of gossip protocols:

Fanout: It is defined as the quantity of nodes that are chosen as gossip targets by a node for every message received initially [7].

Maximum rounds: It is defined as retransmission of an available gossip message by nodes to maximum number of times [7].

Cloud and Distributed System

The word cloud has been utilized to point to stages for distributed processing. Cloud computing is a specific type of Distributed Computing. In Cloud Computing the resources, for example, processors, memory, storage are totally disconnected from the customer. In this way the cloud's seller administration is in charge of the quality, performance, adaptability and security of the cloud facilities. For the designer this is the biggest advantage [7].

Cloud computing includes the applications represented as facilities over the Web and the hardware and framework techniques in the datacenters that give those facilities. The facilities are alluded to as Software as a Service (SaaS). The datacenter equipment and programming is treated as Cloud. At the point when a Cloud is made accessible in a pay-as-you-go way to the people around the globe, it is a Public Cloud and the facility that is sold is Utility Computing. The term Private Cloud is utilized to allude to interior datacenters of a business or different associations not made accessible to the end users. In this manner, it is whole of SaaS and Utility Computing, however does exclude Private Clouds. End Users can be clients or suppliers of SaaS, or clients or suppliers of Utility Computing [7].

Core properties

Cloud computing means distinctive things to diverse users. Nonetheless, we can extensively distinguish Cloud Computing as:

- A virtual impression of boundless computing resources accessible when required.
- An elimination of an up-front commitment by Cloud users.
- A facility to pay for utilization of computing resources as required.
- An ability to give local application response times and 100 percent uptime.
- An abstraction of hardware failures and peak load requirements.

Methodology

Cluster with nodes connected to each other which may be distant to each other geographically communicate with each other to perform tasks. Now if one of these nodes within a cluster is overloaded with requests it initiates a load balancing request for the nodes in the same cluster. Every Node has the following information by which it estimates its load. [Mamdani, E.H. and S. Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller," *International Journal of Man-Machine Studies*, Vol. 7, No. 1, pp. 1-13, 1975.

- Queue Length.

- Current CPU utilization.

Utilization = 100% - (% of time spent in idle task)
% of time spent in idle task
= (Avg_{NL} / Avg_{SL}) * 100
Avg_{NL} – Average period of background task with no load
Avg_{SL} – Average period of background task with some load
Using the above two parameters as input to the Fuzzy Inference Systems(FIS) every node calculates its current load and depending on the incoming requests it predicts that the node will be overloaded or not. If it predicts that the node will be overloaded it initiates a message to its neighboring nodes within the same cluster with the following information:

- Node ID.
- Execution Time. (of Process/Request to be migrated)

The other nodes in the cluster maintain a table of information

NODE ID	Load	Queue Length	CPU Utilization

These other nodes will add the new process of the overloaded node to its queue length and then calculate its load, if the load crosses the threshold (pre-defined) then a negative acknowledgment is sent otherwise the above information is sent to the overloaded node. The overloaded node then sorts these responses and then randomly chooses one node or the one with the least current node will be selected.

Inter Cluster Load Balancing

For intra cluster it may happen that all the nodes currently are overloaded so instead of waiting for an indefinite amount of time the overloaded node will repeat the above procedure for the nodes in different clusters. To avoid broadcasting of the request message and congesting the network we use a protocol for communication of nodes in different clusters. Gossip Protocol is a method which can be used to send request to multiple nodes without actually congesting the network.
Once the overloaded nodes gets the positive acknowledgment from the inter cluster nodes, its sorts these responses with increasing costs. Cost is the transmission time from the overloaded node to the under loaded node. Now, even though a node might have less amount of load currently but its cost from the overloaded is high, it won't be selected for process migration since it will take more amount of time which will result in low response time for the user which is undesirable.

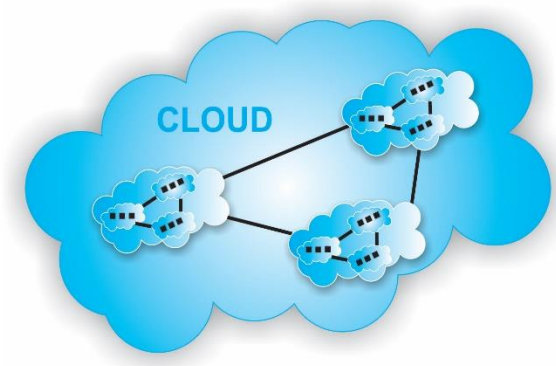

Fig.1: Cluster of Cloud

P_i – Node which wants to balance its load
Q {Q1, Q2,……., Qn} – Other nodes in the network
Suppose node P_i is overloaded with requests and wants to balance its load with its peers. P_i sends a message to its neighboring nodes using the "***Gossip Protocol***". The other nodes in Q reply back with the following information.
The node P_i will estimate the total load of each node Q using the following fuzzy logic.

Input Parameters

- Processor Queue Length.
- CPU utilization.

Input Descriptors (for both the inputs)

- VL (Very Low).
- L (Low).
- M (Medium).
- H (High).
- V (Very high).
- Range – 0% to 100%.

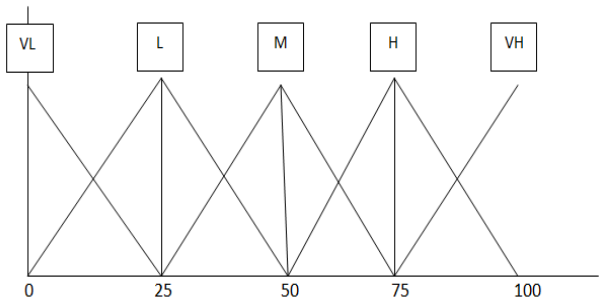

Fig.2: Membership Function – Triangular

Output (Load Descriptors)

- L (Low).
- M (Medium).
- H (High).
- VH (Very High)

Range – 0% to 100%
Q – Queue Length
U – CPU Utilization
The membership function and the descriptors for all the inputs and output remains the same as *Fig. (2)*

Rule Base

⇩Q \ U⇨	VL	L	M	H	VH
VL	VL	VL	L	L	L
L	VL	L	M	M	H
M	L	M	M	H	H
H	L	M	H	H	VH
VH	L	H	H	VH	VH

Fig.3: Fuzzy rule for each case

The above represent the rule base for the fuzzy logic design. The column elements represent the descriptors for Queue length and the row elements represent the descriptors for CPU Utilization. The above table represents every possible situation that may arise in terms of Queue length and CPU Utilization.
If Queue length is Very low and CPU Utilization is High

then the Load is Low.
If Queue length is Medium and CPU Utilization is Medium then Load is Medium.

Inter Cluster Load balancing
Inter-Cluster load balancing will only be considered if the current overloaded node's intra cluster nodes are overloaded. The fact that the load needs to be balanced there is also an implicit barrier of need for low response time. Due to this factor intra cluster node is solved first without thinking about the nodes present in the inter cluster.
If all the nodes in Q are overloaded then move on to the inter cluster nodes. Here the process is repeated but with an additional parameter.

Cost
Time taken to migrate from one node to another node. After calculating the load of the nodes in the clusters, sort the nodes according to their cost from the over loaded node. The least cost node will be selected for migration.

Migration Policy
Migration policy is to determine when and what process to migrate. There are a number of questions that a node needs to answer before taking a decision of migrating that particular process to another node. The following are the questions that a node should answer:

1. When to migrate a process?
   - Current node is overloaded.
   - Doesn't have the resources.
2. How to migrate a process?
   - Total copy.
   - Address space.
3. Which process to select for migration?
   - Highest execution time.
   - Highest demand for resources.
   - IPC dependency.
4. Does the migrating node have the resources?

Example
Consider there is only one cluster and there are 3 nodes. The following table shows the current status of each node.

Table.2: Information maintained by each node

Node ID	Queue Length (%)	CPU Utilization (%)
1	25	65
2	25	10
3	75	90

Now, calculate the load of each node using fuzzy Inference System. After calculation of load for each node, check whether any of the nodes are overloaded or not.

Table.3: Load of each node in the cluster

Node ID	Load (%)
1	50
2	10
3	90

From the above table it is clear that Node 3 is overloaded with 90% as its load. So now node 3 will send request to other nodes *viz.* Node 1 and 2 to balance out the load.
The node 1 and 2 will send a reply back to Node 3 with their current load and then node will choose the node with the least node amongst all the under loaded nodes. In this case it will be Node 2.
Results and discussion
We simulated the environment considering inter and intra cluster nodes. The experiments were performed considering all possible practical cases of load on the network. The parameter used were

- Number of clusters.
- Number of nodes in each cluster.
- Current Queue Length of each node.
- Current CPU Utilization.
- Cost or Time taken for migration

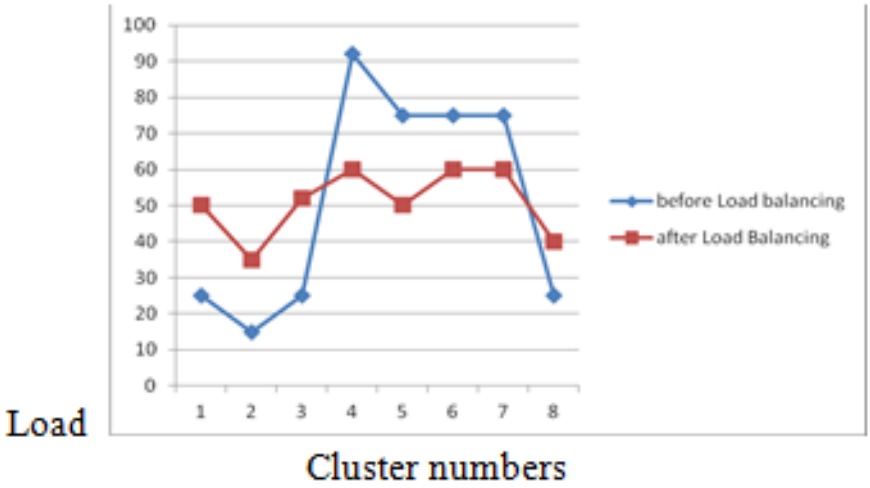

Fig.3: Load balancing

The above graph represent the output of a case where there are 8 nodes in a cluster and there are 4 nodes which are overloaded *viz.* Node 4,5,6 and 7.(Threshold for load is considered to be 66%).
By applying the fuzzy algorithm we balance out the load among the other underloaded nodes in the cluster. The output is the balanced load in the cluster. Though the overall load on the node remains the same but now none of the nodes are overloaded.

Conclusion

In this paper a load balancing technique using fuzzy inference system is discussed. Our method for balancing the load in the network can be used for both inter and intra cluster nodes. The two parameter viz. Queue length and CPU Utilization are pivotal in estimating the load of the node. Though the other parameter do play a part in estimating the same but the only the highest impact elements are considered.

References

1. Mamdani, E.H. and S. Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller," *International Journal of Man-Machine Studies*, Vol. 7, No. 1, pp. 1-13, 1975.
2. Demers, A., Greene, D., Hauser, C., Irish, W., Larson, J., Shenker,S., Sturgis, H., Swinehart, D. & Terry, D. (1987). Epidemic algorithms for replicated database maintenance. In PODC ’87: Proceedings of the sixth annual ACM Symposium on Principles of distributed computing, 1–12, ACM Press, New York, NY, USA
3. Renesse, R.V., Minsky, Y. & Hayden, M. (1998). A gossip-style failure detection service. Tech. Rep. TR98-1687, Dept. of Computer Science, Cornell University.
4. Eugster, P.T., Guerraoui, R., Handurukande, S.B., Kouznetsov, P. & Kermarrec, A.M. (2003). Lightweight probabilistic broadcast. ACM Trans. Comput. Syst., 21, 341–374.
5. Ganesh, A.J., Kermarrec, A.M. & Massoulie, L. (2001). SCAMP: Peertopeer lightweight membership service for large-scale group communication.In Networked Group Communication, 44–55.
6. Voulgaris, S., Gavidia, D. & Steen, M. (2005). Cyclon: Inexpensive membership management for unstructured p2p overlays. Journal of Network and Systems Management, 13, 197–217.
7. João Antunes Carlos Leitao, “Gossip-based broadcast protocols”.
8. Sandeep S, Sarabjit S, and Meenakshi S, "Performance Analysis of Load Balancing Algorithms," International Journal of Computer, Electrical, Automation, Control and Information Engineering Vol:2, No:2, 2008. Malik, "Dynamic Load Balancing in a Network of Workstation," 19 November 2000 2000.
9. Abbas Karimi, Faraneh Zarafshan, Adznan b. Jantan1,A.R. Ramli1, M. Iqbal b.Saripan, “A New Fuzzy Approach for Dynamic Load Balancing Algorithm”.
10. Z. Xu and R. Huang, "Performance Study of Load Balancing Algorithms in Distributed Web Server Systems " CS213 Parallel and Distributed Processing Project Report.
11. R. Motwani and P. Raghavan, "Randomized algorithms," ACM Comput. Surv., vol. 28, pp. 33-37, 1996.
12. P. L. McEntire, J. G. O'Reilly, and R. E. Larson, Distributed Computing: Concepts and Implementations. New York: IEEE Press, 1984.
13. W. I. Kim and C. S. Kang, "An adaptive soft handover algorithm for traffic-load shedding in the WCDMA mobile communication system," presented at WCNC'2003, 2003.
14. n. Y.-T. Wang and A.-R. J. T. Morris, "Load Sharing in Distributed Systems," IEEE Transactions on Computers, vol. 34, pp. 204-217, 1985.Distributed Systems," IEEE Transactions on Computers, vol. 34, pp. 204-217, 1985.
15. W. Leinberger, G. Karypis, and V. Kumar, "Load Balancing across Near-Homogeneous Multi-Resource Servers," presented at s, Cancun, Mexico, 2000.
16. Grace Ramamoorthy, “Distributed System and Cloud Computing”.
17. Kun-Ming V. Yu, Chih-Hsun Chou, and Yao-Tien Wang. “A Fuzzy-Based Dynamic Load-Balancing Algorithm”.