Effect of different defuzzification methods in a fuzzy based load balancing application

Sameena Naaz[1], Afshar Alam[2] and Ranjit Biswas[3]

[1]Department of Computer Science
Jamia Hamdard, New Delhi, India

[2]Department of Computer Science
Jamia Hamdard, New Delhi, India

[3]Manav Rachna International University
Faridabad, Haryana, India

Abstract

In a distributed environment the workload on the network has to be managed in such a way that the total throughput of the system can be maximized. For this to happen some of the jobs have to be migrated from one node to another. When, how and where a job has to be migrated depends upon the load balancing algorithm being used. But it is very difficult to precisely describe the behavior of a complex system as there are many factors which influence it. One way to deal with the uncertainty in the behavior of the system is to use fuzzy logic. Fuzzy logic uses the reasoning of the human mind which is not always in the form of a yes or no. The concept of linguistic variables is used to model the state of the system which is imprecise and uncertain. In this work, we have implemented the fuzzy load balancing algorithm and compared the effect of using different defuzzification methods, reported in the literature.

Keywords: Distributed Systems, Load Balancing, Fuzzy Logic, Defuzzification.

1. Introduction

Over the years the hardware technology has grown on a massive pace with the result of increase in the use of distributed systems. These systems have the advantage of sharing of resources as well as processing power. The processes arrive in the system in a random manner on different nodes. When the jobs are being executed in parallel on different systems a decision has to be made on to which system a newly arrived job has to be send. Load balancing is the technique which helps in even distribution of the jobs among the available nodes so that the throughput can be increased.

The load balancing algorithms can be categorized as static or dynamic in nature. Static algorithms collect no information and make probabilistic balancing decisions, while dynamic algorithms collect varying amounts of state information to make their decisions. Previous research on static and dynamic load balancing can be found in [l]-[5], [6, 7], respectively. It has been established from the previous studies that dynamic algorithms give better performance improvement as compared to static algorithms.

Different load balancing algorithms have different complexity which depends upon the amount of communication needed to approximate the least loaded node. In order to make a decision the information about the state of the different nodes has to be collected. However, since messages containing state information for individual nodes can only be exchanged at discrete intervals and are subject to variable latencies before reaching their destinations, the information used by nodes to estimate global system state is inevitably out of date. This uncertainty in global state has been a primary issue in the design of efficient distributed computing systems. Increasing the frequency of information exchange between nodes is not necessarily a practical solution since message overheads caused by the frequent exchange of state information may adversely affect the efficiency of the system. Moreover, the overheads of load balancing mechanisms can be highly detrimental to the performance of the system under heavy system load conditions.

When we are talking about large distributed systems there is huge amount of global state uncertainty present in it. Fuzzy logic based distributed load balancing algorithms reflect the effect of uncertainty in decision making process.

This approach has been discussed in [8]. The fuzzy logic approach for Distributed Object Computing Network has been studied in [9, 10]. Parallel and distributed computing environment is inherently best choice for solving/running distributed and parallel program applications. In such type of applications, a large process/task is divided and then distributed among multiple hosts for parallel computation. In [10] it has been pointed out that in a system of multiple hosts the probability of one of the hosts being idle while other host having multiple jobs queued up can be very high. In [11] the performance of a new Fuzzy Load balancing algorithm is compared with the existing algorithms.

In a distributed environment the processors are categorized according to workload in their CPU queues as heavily loaded (more tasks are waiting to be executed), lightly loaded (less tasks are waiting to be executed in CPU queue) and idle processors/hosts (having no pending work for execution). Here CPU queue length is used as an indicator of workload at a particular processor. The algorithms used for load balancing may require no information, or only information about individual jobs (static algorithm) or may make decisions based on the current load situation (dynamic algorithm).

In general, load balancing algorithm can be analyzed in a framework with four dimensions: selection policy, transfer policy, information policy, and location policy. Specifically, information and location policies have the most important roles.

Transfer policy: First of all the state of the different machines is determined by calculating it's workload. A transfer policy determines whether a machine is in a suitable state to participate in a task transfer, either as a sender or a receiver. For example, a heavily loaded machine could try to start process migration when its load index exceeds a certain threshold.

Selection policy: This policy determines which task should be transferred. Once the transfer policy decides that a machine is in a heavily-loaded state, the selection policy selects a task for transferring. Selection policies can be categorized into two policies: preemptive and non-preemptive. A preemptive policy selects a partially executed task. As such, a preemptive policy should also transfer the task state which can be very large or complex. Thus, transferring operation is expensive. A non-preemptive policy selects only tasks that have not begun execution and, hence, it does not require transferring the state of task.

Location policy: The objective of this policy is to find a suitable transfer partner for a machine, once the transfer policy has decided that the machine is a heavily-loaded state or lightly-loaded one. Common location policies include: random selection, dynamic selection, and state polling.

Information policy: This policy determines when the information about the state of other machines should be collected, from where it has to be collected, and what information is to be collected.

2. Fuzzy Logic Concept

In narrow sense, fuzzy logic is a logical system, which is the extension of multivalued logic. In a wider sense fuzzy logic is almost synonymous with the theory of fuzzy sets, a theory which relates to classes of object with unsharp boundaries in which membership is a matter of degree. Fuzzy inference is the process of formulating the mapping from a given input to an output using fuzzy logic. The mapping then provides a basis from which decisions can be made.

Basically a fuzzy logic system consists of the following 5 steps:

Fuzzification: Converting the crisp inputs to membership functions which comply with intuitive perception of system status.

Rules Processing: Calculating the response from system status inputs according to the pre-defined rules matrix (control algorithm implementation)**.**

Inference: Evaluating each case for all fuzzy rules

Composition: Combining information from rules

De-Fuzzification: Converting the result to crisp values.

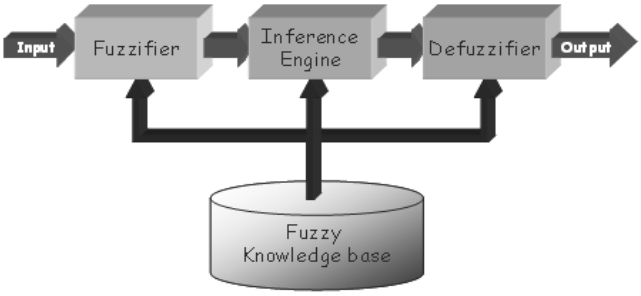

Figure 1: Fuzzy Inference System

In this paper we have compared the following five defuzzification methods

Centroid of area Z_{COG}
Bisector of area Z_{BOA}
Mean of maximum Z_{MOM}
Smallest of maximum Z_{SOM}

Largest of maximum Z_{LOM}

Centroid principle or Center of Gravity
This method is also known as center of gravity or center of area defuzzification. This technique was developed by Sugeno in 1985. This is the most commonly used technique. The only disadvantage of this method is that it is computationally difficult for complex membership functions. The centroid defuzzification technique can be expressed as

$$z_{COG} = \frac{\int_{Z} \mu_A(z) z dz}{\int_{Z} \mu_A(z) dz}$$

where z_{COG} is the crisp output, $\mu_A(z)$ is the aggregated membership function and z is the output variable

Bisector Method
The bisector is the vertical line that divides the region into two sub-regions of equal area. It is sometimes, but not always coincident with the centroid line.

$$\int_{\alpha}^{z_{BOA}} \mu_A(z) dz = \int_{z_{BOA}}^{\beta} \mu_A(z) dz$$

Largest of Maximum
Largest of maximum takes the largest amongst all z that belong to [z1, z2] as the crisp value called $Z_{LOM.}$

Smallest of Maximum
It selects the smallest output with the maximum membership function as the crisp value Z_{SOM}. In other words in Smallest of Maximum choose smallest among all z that belong to [z1, z2]

Mean of Maximum
In this method for defuzzification only active rules with the highest degree of fulfillment are taken into account. The output is computed as:

$$z^* = (a+b)/2$$

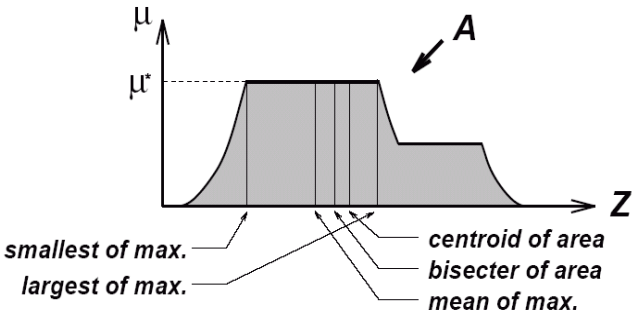

Figure 2: Results using different defuzzification methods for a particular function.

3. Distributed System Model

A simple model of a distributed system is presented here. This model consists of a decentralized decision making approach with cooperation from all the nodes. So the performance can be improved here purely by intelligent decision making and proper coordination. The various nodes of the system here are the resources and each of these resources can be in different states. A vector is used to give the state of a node which describes many characteristics of the node. The elements of this state vector are measures which imply a cost or penalty for using the resource.
The set of states of all the resources in the distributed system is known as the global system state. In distributed load balancing also the decisions are not always necessarily made using the complete global state information. In fact for each node under consideration only a subset of neighboring nodes may be needed to take a decision. Another important aspect is that a node can change state faster than the time taken to transmit state information from one state to another. Therefore there is always some amount of uncertainty in the state information used for making a decision. Hence it is necessary that the decision making process deals with these uncertainties. Fuzzy logic is one of the methods of dealing with this uncertain information and has been used in the work presented in this paper.
The Scheduler in this algorithm has to perform the following tasks.

Threshold Estimation

Decision Making

Scheduler has two functions, threshold estimation and decision making. When a scheduler is invoked, it estimates two numerical thresholds from the current states of uncertainty sources based on a fuzzy control base, and

making scheduling and state update decision using fuzzy consistency model.

We need to define fuzzy sets for the input parameters, 'load', and 'number of heavy load node' levels, and the output, 'status of load balancing node'. For this we define five membership functions for first input parameter i.e. 'load' and two membership functions for second input parameter i.e. 'number of heavy load node' and two membership functions for output parameter 'status of load balance node.

3.1 Threshold Estimation

The Threshold Estimation determines the limiting value for each membership function. Beyond this limiting value the membership function will change.

First Input parameter: Load (0-10)
Member Function 1: Very lightly (0-2)
Member Function 2: lightly (1-5)
Member Function 3: moderate (4-6)
Member Function 4: heavy (5-9)
Member Function 5: very heavy (8 -10)

Second Input Parameter: No. of heavy load node (0-5)
Member Function 1: more (0-2.5)
Member Function 2: less (2.5 – 5)

Output Parameter: Status of load balance node (0-10)
Member Function 1: receiver (0-5)
Member Function 2: sender (6-10)

In our work here we have taken the Gaussian distribution function for all the different linguistic variables for the input "load". This is shown in figure 3.

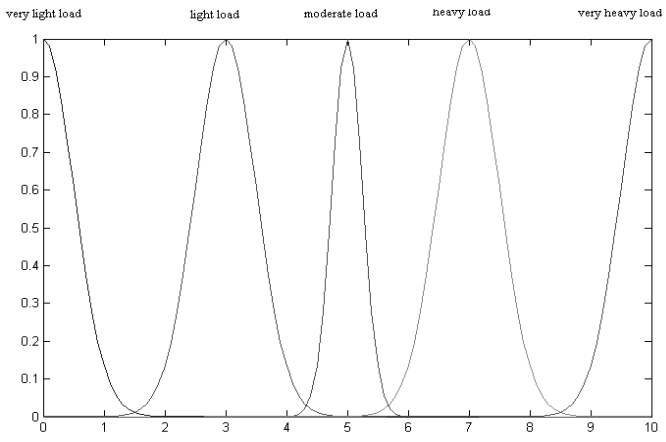

Figure 3: Input variable load of the node under consideration and it's membership function

The membership function used for the number of heavy load nodes is shown in figure 4.

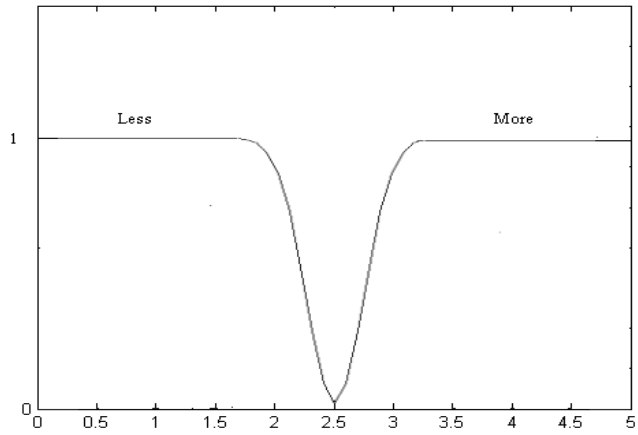

Figure 4: Input variable No. of Heavy Load Node and it's membership function.

The membership function for the output variable status of load balance node is shown in figure 5. From this figure we can see that there are two linguistic variables sender and receiver here and the load on a node determines it's value based upon the membership function.

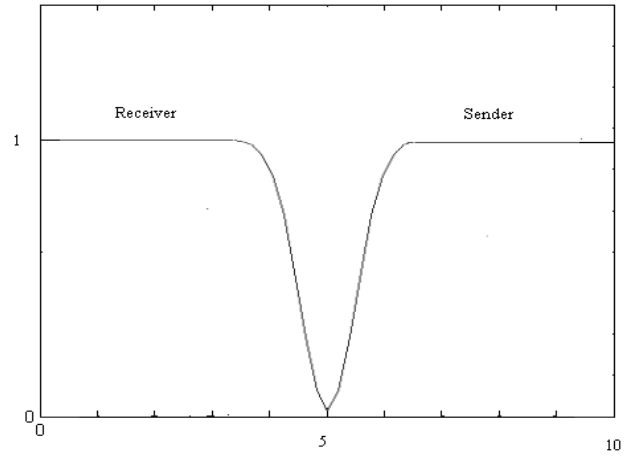

Figure 5: Membership function for the output variable Status of Load Balance Node

3.2 Decision Making

The Fuzzy rules that have been used in this work are given below:

Rule [1]. If (load is very light) then (node is receiver)

Rule [2]. If (load is very heavy) then (node is sender)

Rule [3]. If (load is heavy) and (no. of heavy load nodes is less) then (node is sender)

Rule [4]. If (load is heavy) and (no. of heavy load nodes is more) then (node is receiver)

Rule[5]. If (load is light) and (no. of heavy load nodes is more) then (node is receiver)

Rule[6]. If (load is light) and (no. of heavy load nodes is less) then (node is sender)

Rule [7]. If (load is moderate) and (no. of heavy load nodes is more) then (node is receiver)

Rule [8]. If (load is moderate) and (no. of heavy load nodes is less) then (node is sender)

This rule base is used to find out the value of the output variable using the fuzzy inference method.

4. Interpretation of Results

We have done the implementation of scheduler on MATLAB. We have taken two input parameters and one output parameter for fuzzy implementation of our logic. The first input parameter is 'load' and the second one is 'Number of heavy Load Node' and one output i.e. 'status of load balance node'. We measure the input parameters load and Number of heavy load node on a scale of 0 to 10 and 0 to 5 respectively and the output parameter status of load balancing node on a scale of 0 to 10

Based upon the crisp values that are obtained the nodes are categorized either as sender or as receiver. We have calculated this crisp value using the five defuzzification methods described above.

The surface plots that we obtain for the results are shown in Figures 6 to 10 and the input and output values obtained for 20 sets of data is shown in Table 1.

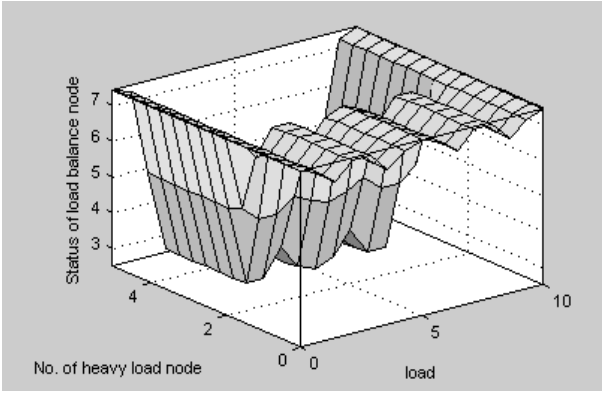

Figure 6 : Centroid Method

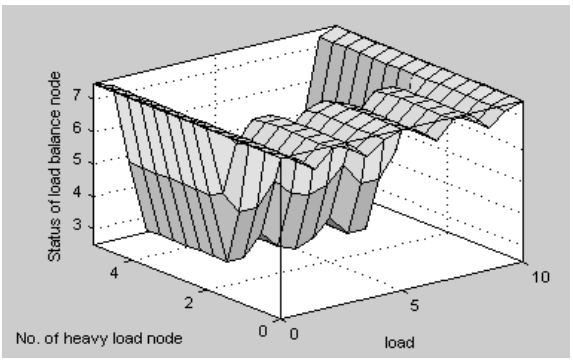

Figure 7: Bisector method

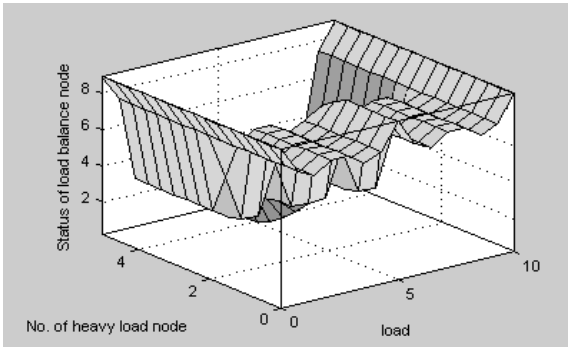

Figure 8: Mean of Maximum

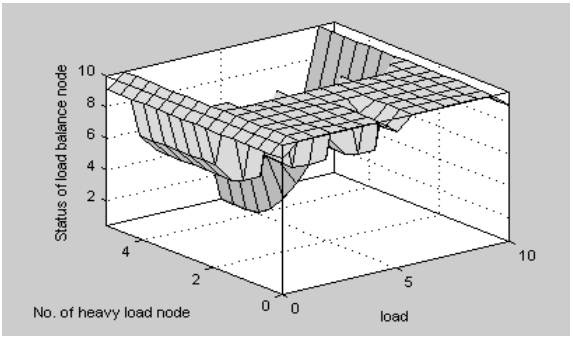

Figure 9: Largest of Maximum

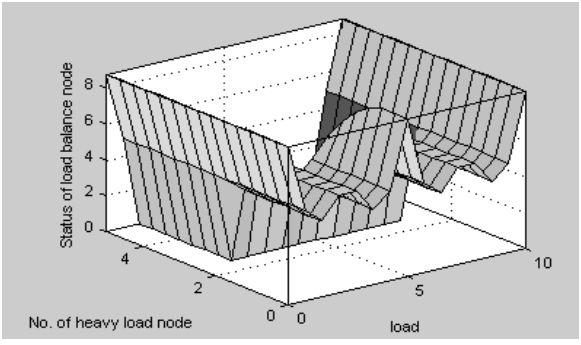

Figure 10: Smallest of Maximum

Table 1: Output values obtained for different defuzzification methods.

S.No	INPUTS		OUTPUT				
	Load	No. of Heavy Load Node	Centroid	Bisector	MOM	LOM	SOM
1	7	2	7.2647	7.3000	7.5500	10.0000	5.1000
2	4	5	2.8181	2.8000	2.6000	5.2000	0
3	9	3	7.0142	7.1000	7.3500	10.0000	4.7000
4	6	1	7.1095	7.1000	7.3500	10.0000	4.7000
5	4	5	2.8181	2.8000	2.6000	5.2000	0
6	10	5	7.4348	7.5000	8.9500	9.1000	8.8000
7	9	3	7.0142	7.1000	7.3500	10.0000	4.7000
8	6	1	7.1095	7.1000	7.3500	10.0000	4.7000
9	7	2	7.2647	7.3000	7.5500	10.0000	5.1000
10	6	3	2.8415	2.8000	2.6000	5.2000	0
11	3	3	2.6763	2.7000	2.4000	4.8000	0
12	4	2	7.0955	7.1000	7.3500	10.0000	4.7000
13	5	4	2.5057	2.5000	0.2000	0.4000	0
14	3	4	2.6635	2.6000	2.4000	4.8000	0
15	9	2	7.0955	7.1000	7.3500	10.0000	4.7000
16	2	1	7.2724	7.3000	7.5500	10.0000	5.1000
17	3	2	7.2647	7.3000	7.5500	10.0000	5.1000
18	2	2	7.2647	7.3000	7.5500	10.0000	5.1000
19	3	3	2.6763	2.7000	2.4000	4.8000	0
20	5	3	2.5143	2.5000	1.4000	2.8000	0

5. Conclusion and Future Work

The results obtained using the five defuzzification methods have been shown in table1. From this table we find that centroid method, bisector method and mean of maximum method are giving us approximately the same results in the load balancing application that we have taken. Where as for the smallest of maximum and largest of maximum approaches there is wide variations in the results that are obtained. The reason for this is that these two methods use the two extremes i.e smallest or largest values for calculation of the crisp value.

The results obtained in the tables above are graphically shown in figures 6 to 10 and from these figures also we infer the same results. Hence we conclude that centroid, bisector and MOM methods are better as compared to the LOM, SOM, as there is more consistency in the results.

In future this work has to be extended by using these methods in actual simulation for load balancing to find out the effect on response time.

References

[1]. P. V. McGregor and R. R. Boorstyn, “Optimal load balancing in a computer network,” in Proc. 1975 Int. Conf: on Commun., vol. 3, pp. 41.14-41.19.

[2]. E. D. S. Silva and M. Gerla, “Load balancing in distributed systems with multiple classes and site constraints,” in Proc. Performance’84, pp. 17-33.

[3]. A. N. Tantawi and D. Towsley, “A general model for optimal static load balancing in star network configurations, ” in Proc. Performance’84, pp. 277-291.

[4]. A. N. Tantawi and D. Towsley, “Optimal static load balancing in distributed computer systems,” J. ACM, vol. 32, no. 2, pp. 445465, Apr. 1985.

[5]. J. F. Kurose and S. Singh, “A distributed algorithm for optimum static load balancing in distributed computer systems,” in Proc. IEEE INFOCOM’86, pp. 458-467.
[6]. F. Bonomi and A. Kumar, “Adaptive optimal load balancing in a heterogeneous multiserver system with a central job scheduler,” IEEE Trans. Computer, vol. 39, pp. 1232-1250, Oct. 1990.
{7]. T. C. K. Chow and J. A. Abraham, “Load balancing in distributed systems,” IEEE Trans. Software Eng., vol. SE-8, pp. 401-412, July 1982.
[8]. Chulhye Park and Jon G.Kuhl, “A Fuzzy based distributed load balancing algorithm”, Proceedings of the Second International Symposium on Autonomous Decentralized Systems (ISADS'95) IEEE, 1995.
[9]. Lap-Sun Cheung, “A Fuzzy Approach to Load Balancing in a Distributed Object Computing Network”, First IEEE International Symposium on ClusterComputing and the Grid (CCGrid'01) risbane, Australia May 15-May 18, 2001.
[10]. Yu-Kwong Kwok And Lap-Sun Cheung, “A New Fuzzy-Decision Based Load Balancing System For Distributed Object Computing”, Journal Of Parallel And Distributed Computing, Volume 64 Issue 2, February 2004
[11]. Abbas Karimi, Faraneh Zarafshan, Adznan b. Jantan, A.R. Ramli and M. Iqbal b. Saripan, “A New Fuzzy Approach for Dynamic Load Balancing Algorithm” International Journal of Computer Science and Information Security(IJCSIS).

AUTHORS PROFILE

Sameena Naaz received the degree of B.Sc Engg. in computers from Aligarh Muslim University, in 1998 and the M..Tech Degree in Electronics from Aligarh Muslim University, in 2000. She is pursuing her Ph. D from Hamdard University. Sameena Naaz has worked as a lecturer at Amity College of Engg. And Tech. Delhi, Inti College Malaysia and is currently working as an Assistant Professor at Jamia Hamdard University, New Delhi India in the Department of Computer Science. She is a member of International Association of Computer Science and Information Technology (IACSIT). Her research interests include soft computing and load balancing and scheduling in distributed systems.

Professor Afshar Alam has an MCA and Ph. D degree and is working as a Professor at Jamia Hamdard University in the Department of Computer Science

Professor Ranjit Biswas has an M.tech , Ph. D and is currently working with Manav Rachna International University Faridabad, Haryana, India.