A New Fuzzy Algorithm for Dynamic Load Balancing In Distributed Environment

Nidhi Kataria Chawla
Assistant Professor (Babu Banarsi Das University, Luck now)
U.P, India
ernidhikataria@gmail.com

Abstract - **Load Balancing means to distribute the load among the processors to improve the performance of a parallel and distributed system. This project presents a new fuzzy algorithm for dynamic load balancing in distributed environment. This approach characterizes the uncertainty in a distributed system by using the fuzzy set theory , fuzzy control methods and presents a fuzzy based distributed load balancing algorithm that explicitly reflects the effect of uncertainty in the decision making process. The notion of the linguistic variables is used to model the sender and receiver node. In this we have taken the two input variables, 'load' and 'number of heavy load nodes' and one output variable 'status of load balance node'. On the basis of decision rules we have concluded the status of load balance node i.e. whether the node is receiver or sender. The decision rules have implemented on the MATLAB. Then the system model is created on network simulation software. Simulation results show the response time.**

Keywords: SLB, DLB, Load Balancing, Fuzzy Logic

I INTRODUCTION

Distributed computer systems are becoming increasingly available because of the rapid decrease in hardware cost and the advances in computer networking technologies. It is frequently observed that in a computing environment with a number of hosts connected by a network, the hosts are often loaded differently. In typical distributed system task arrive at the different nodes in random fashion .This causes a situation of non-uniform load across the different nodes .Load imbalance is observed by existence of nodes that are highly loaded while the others are lightly loaded or even idle. Such situation is harmful to the system performance in terms of response time and resource utilization. To overcome such problem by increasing the overall throughput of the system by allowing heavily loaded nodes to redistribute tasks to lightly loaded nodes. To achieve such a goal, system nodes must exchange state information and make control decisions.

A Load Balancing Load balancing is a technique to spread work between two or more computers, network links, CPUs, hard drives, or other resources, in order to get the optimal resource utilization, throughput, or response time [5]. Load balancing is done by migrating tasks from the overloaded nodes to other lightly loaded nodes to improve the overall system performance. 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).
There are two types load balancing algorithm:

Static Load Balancing (SLB): In this method performance of the nodes is determined at the beginning of the execution. Then depending upon their performance the workload is distributed in the start by master node. The slave processors calculate their allocated work and submit the result to the master [1].

Dynamic Load Balancing (DLB): DLB algorithms that consider the current load conditions (i.e. at execution time) in making job transfer decisions. In this method the workload is distributed among the nodes at runtime. The master assigns a new process to the slave based on the new information collected. Dynamic load balancing does the process while job are in execution. In dynamic load balancing the load at each host is calculated (as number of process, structure of node) and redistributes the load from heavily loaded processor to the lightly loaded processor. Dynamic load balancers continually monitor the load on all the processors, and when the load imbalance reaches some predefined level, this redistribution of work takes place. This redistribution does incur extra overhead at execution time [1].

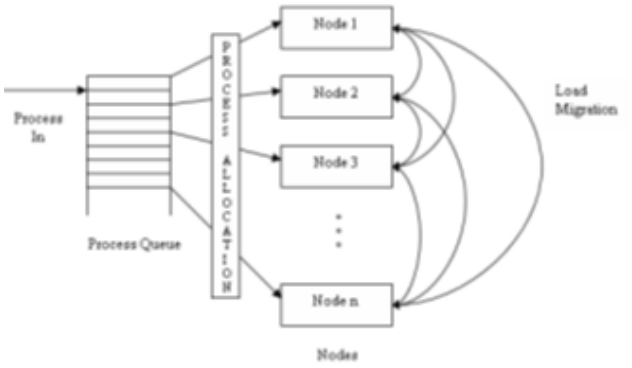

Figure 1: Simple dynamic Load balancing to avoid overload on heavily loaded node by transferring process to light weighted node.

Problem Definition: My work is to do propose an improved algorithm for dynamic load balancing in the distributed environment using fuzzy logic concept.

II. REVIEW OF LITREATURE

1. As an area of research, load balancing has received considerable attention since the early days of distributed systems in 1970s.
2. A Fuzzy based distributed load balancing algorithm, Chulhye Park and Jon G.Kuhl, 1995 IEEE
3. A global plan policy for coherent co-operation in distributed dynamic load balancing algorithm, Mourad Kara, 1995,
4. A Fuzzy based Dynamic Load balancing Algorithm, Kun Ming V.Yu, Chih –Husun Chou, 1998
5. A Fuzzy Approach to Load Balancing in a Distributed Object Computing Network, Brisbane, Australia , ccgrid, pp.694, First IEEE International Symposium on Cluster Computing and the Grid (CCGrid'01), 2001.

III. CONCEPT OF FUZZY LOGIC IN LOAD BALANCING

In this section, a fuzzy logic model that performs dynamic load balancing is proposed. The model is based on fuzzy logic that takes imprecise information for its input. This information is passed to the fuzzy inference engine which performs the analysis and assigns different ranks to the services. Clients can then choose the most appropriate service based on the rank of service. In our implementation, one of the service attributes is used as the service rank which characterized the availability of service. [2]

The fuzzy expert system reflects the impression in state information and makes scheduling decision based on a fuzzy logic. We believe that the performance of any distributed computing system cannot be improved beyond a limit which is determined by the degree of uncertainty in global state. Therefore we propose a new dynamic load balancing algorithm that characterizes uncertainty in the decision making process. In this mechanism, each distributed node dynamically adjusts its thresholds denoting the amount of consistency relaxation depending on the degree of uncertainty in the system state. A fuzzy based consistency model provides a mechanism that allows each node to make flexible scheduling and state update decisions based on its threshold.

A Fuzzy Logic

Fuzzy logic was develop by Zadeh and represents a form of mathematical logic. Values between 0 and 1 represent uncertainty in decision-making. 0 indicates a false value while 1 indicates a true value. So within a fuzzy set a value x is not restricted by the values 0 or 1, but from the real interval {0; 1}

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 un-sharp boundaries in which membership is a matter of degree. [3]

B. Fuzzy Logic Controller

To tackle the load balancing problem, conventional control theory can be applied to restore system equilibrium. Fuzzy logic control attempts to capture intuition in the form of IF-THEN rules, and conclusions are drawn from these rules.

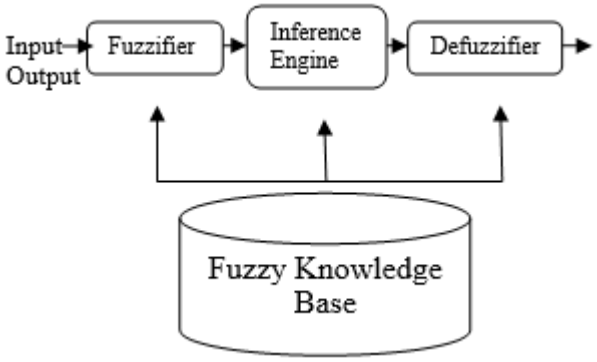

Figure 2: Fuzzy Inference Engine

The fuzzifier is the input interface which maps a numeric input to a fuzzy set so that it can be matched with the premises of the fuzzy rules defined in the application-specific rule base. The rule base contains a set of fuzzy if-then rules which defines the actions of the controller in terms of linguistic variables and membership functions of linguistic terms. The fuzzy inference engine applies the inference mechanism to the set of rules in the fuzzy rule

base to produce a fuzzy set output. This involves matching the input fuzzy set with the premises of the rules, activation of the rules to deduce the conclusion of each rule that is fired, and combination of all activated conclusions using fuzzy set union to generate fuzzy set output. The defuzzifier is an output mapping which converts fuzzy set output to a crisp output. Based on the crisp output, the fuzzy logic controller can drive the system under control.

IV. IMPLEMENTATION AND RESULT

The Algorithm:

We present the fuzzy based distributed load balancing algorithm.

The Algorithm basically divided into two parts:

I. The system Model
II. The Scheduler
   a) Threshold Estimation
   b) Decision Making

The System Model

A distributed system is assumed to be c collection of autonomous nodes connected by a communication network. Message passing is the only form of communication among nodes. The system model with d dimensions has 2^d nodes. If we number the nodes from 0 thru (2^d)-1 and look at the numbers as d-digit binary numbers, then each node will be connected to d other nodes which differ only in one bit. That is, suppose d=4 for example, then 0010 (node 2) will be connected to 1010 (node 10), 0110 (node 6), 0000 (0) and 0011 (node 3). This is how the hypercube is built in hcubened; the gate number is the number of the bit that differs, that is, node 0010's gate #2 goes to node 0110.

The system model module in hcubened takes the d dimension as a parameter. It also takes another parameter called nodetype: a string that names the module type to be used as node in the hypercube. In hc_net.ned, when a hypercube network is described, nodetype is give the "HypercubeNode" value. HypercubeNode is a compound module type which consists of a sender (source) and receiver (sink) and a router module.

The Scheduler

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.

Implementation of the Scheduler

In fuzzy logic toolbox we have taken two input parameters. 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 load and Number of heavy load node on a 0 to 10 scale and status of load balancing node on 0 to 20 scale. (This is just to keep the example simple obviously any scale could have been used).

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.’

FIRST INPUT PARAMETER: Load (0-10)
Mf1 : Very lightly
Mf2 : lightly
Mf3: moderate
Mf4: heavy
Mf5: very heavy
SECOND INPUT PARAMETER: No. of heavy load node (0-10)
Mf1: more
Mf2: less

OUTPUT PARAMETER: Status of load balance node (0-20)
Mf1 : sender
Mf2: receiver

Define Decision Rules

Assuming sender initiated load balance algorithm proposed knowledge base is as follows:

Rule [1]. If (load is very_lightly_load) then (status__loadbalance__node is receiver)

Rule [2]. If (load is very_heavy_load) then (status__loadbalance__node is sender)

Rule [3]. If (load is heavy_load) and no.__heavy__load___nodes is less) then (status__loadbalance__node is sender)

Rule [4]. If (load is heavy_load) and (no.__heavyload___nodes is more) then (status__loadbalance__node is receiver)

Rule[5]. If (load is lightly_load) and (no.__heavy__load___nodes is more) then (status__loadbalance__node is receiver)

Rule[6]. If (load is lightly_load) and (no.__heavy__load___nodes is less) then (status__loadbalance__node is sender)

Rule [7]. If (load is moderate_load) and (no.__heavy__load___nodes is more) then (status__loadbalance__node is receiver)

Rule [8]. If (load is moderate_load) and (no.__heavy__load___nodes is less) then (status__loadbalance__node is sender). [4]

Implementation of System Model

A distributed system is assumed to be c collection of autonomous nodes connected by a communication network. Message passing is the only form of communication among nodes. We have implemented system model on a Network Simulation Tool. The system model with d dimensions has 2^d nodes. If we number the nodes from 0 thru (2^d)-1 and look at the numbers as d-digit binary numbers, then each node will be connected to d other nodes which differ only in one bit.When a network model is described, nodetype is give the "HypercubeNode" value. The system model is a compound module type which consists of a sender (source) and receiver (sink) and a router module.

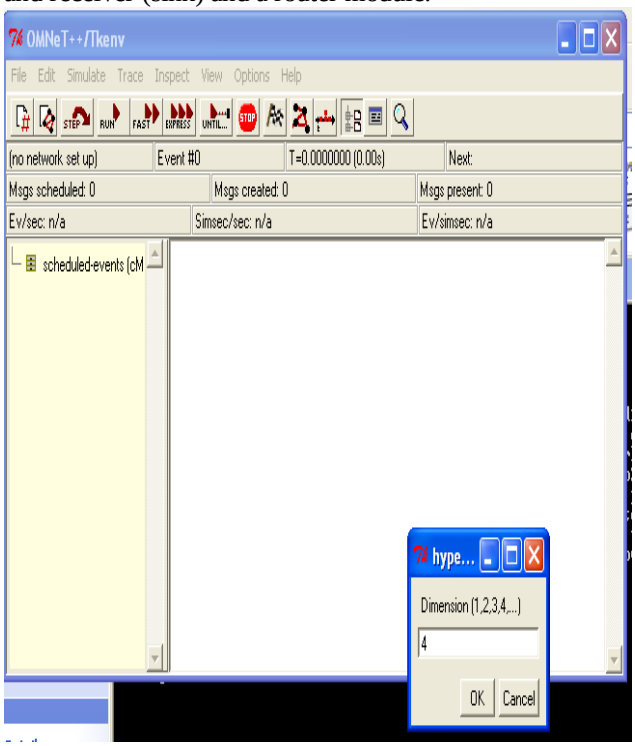

Figure 3: Giving the Dimension Omnet++

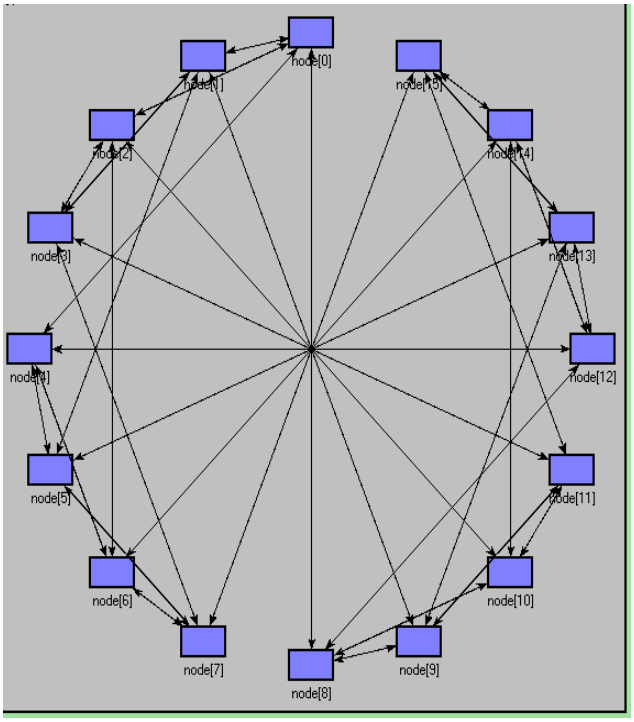

Figure 4: Hypercube Network

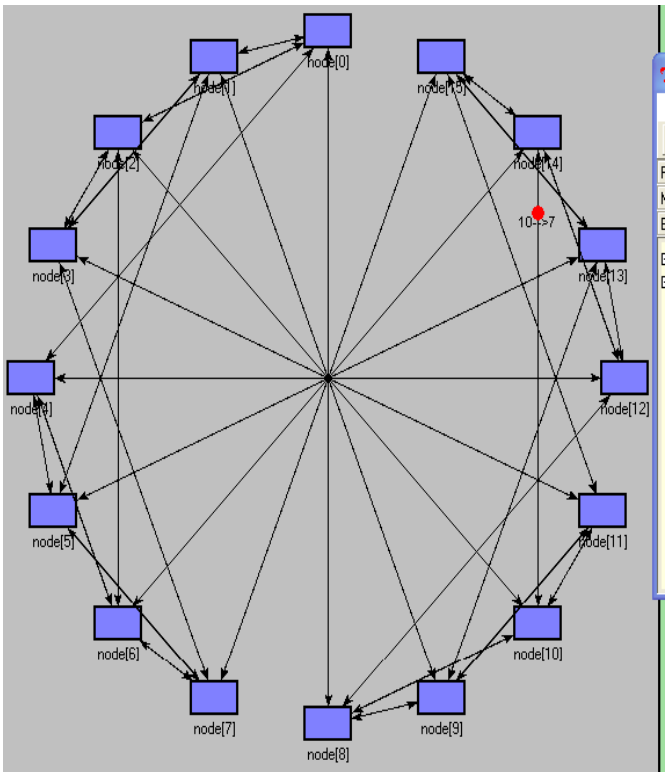

Figure 5: Load Is Transferring from One Node to Another

V. RESULTS

Sender	Receiver	Response Time
0	8	1.99
1	9	1.99
9	11	1.99
11	9	1.99
13	9	1.99
15	7	1.99
15	11	1.16392
3	10	2.99
2	11	2.99
5	0	2.99
4	2	2.88
7	14	2.03683
10	3	2.99
12	10	2.99

VI. CONCLUSION

In this project we present a new algorithm for dynamic load balancing in distributed environment using fuzzy logic. Preliminary results show the response time for each sender and receiver pair.

VII. FUTURE WORK

In this work I have compared the results with the existing algorithms simulated on a different platform. In future it can be compared with the existing algorithms simulated on the same platform as mine. This will be a better estimate of our comparison.

REFERENCES

[1] http://waset.org/publications/5537/performnce-analysis-of-load-balancing-algorithms
[2] http://citeseerx.ist.psu.edu/viewdoc/download?
[3] doi=10.1.1.75.7260&rep=rep1&type=pdf
[4] http://www.mathworks.in/help/fuzzy/foundations-of-fuzzy-logic.html
[5] http://arxiv.org/ftp/arxiv/papers/0910/0910.0317.pdf
[6] http://arxiv.org/ftp/arxiv/papers/0910/0910.0317.pdf