Improved Proximity Aware Load Balancing for Heterogeneous Nodes

Pooja Gandodhar
ME (Computer)
Department of Computer Engineering
PCCOE, Pune

Sudarshan Deshmukh
Asst. Professor
Department of Computer Engineering
PCCOE, Pune

ABSTRACT
Conventional load balancing schemes are efficient at increasing the utilization of CPU, memory, and disk I/O resources in a Distributed environment. Most of the existing load-balancing schemes ignore network proximity and heterogeneity of nodes. Those schemes follow the technique of self owned resource based load balancing. But load balancing involves more challenges due to lack of sources for some of the tasks.

Hence it is not always possible the task get all required resources on a single node. Consider node conditions i. e. the other nodes situated in its context. The task which lack with some of the resources will try to negotiate those resources with these contextual nodes. Then a task can complete and the resources utilized will be released. Doing this we are able to reduce load migrations.

This proposed scheme not only works well with heterogeneous nodes but also considers proximity in them. Our result shows more than 20% reduction in communication costs than the previous methods based on self-owned resource distribution of nodes.

Keywords
Distributed system, context based load balancing, and proximity aware load balancing.

1. INTRODUCTION
Load balancing on multi computers is a challenge due to the autonomy of the processors and the interprocessor communication overhead incurred in the collection of state information, communication delays, redistribution of load etc. 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 a system of multiple hosts the probability of one of the hosts being idle while other host has multiple jobs queued up can be very high. Here load balancing is likely to improve performance. Such imbalances in system load suggest that performance can be improved by either transferring jobs from the currently heavily loaded hosts to the lightly loaded ones or distributing load evenly/fairly among the hosts .The algorithms known as load balancing algorithms, helps to achieve the above said goal(s).

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). The transfer of a job may be initiated by the originating host (source-initiative algorithm), or by the target host (server-initiative algorithm). The unit of execution that is to transferred/redistributed may range from complete jobs submitted by the users, or individual processes, or even smaller program modules. Finally, the transfer of a job may be restricted to be done prior to the start of its execution (initial job placement), or may also be allowed during its execution (process migration).

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. Some others workload indicators (also known as load index) have been suggested by researchers. But none of them is found to be an idle one.

Distributed systems possess multiple resources with transparency in system operations. There is variety of differing techniques and methodologies for scheduling processes of a distributed system have been proposed. All these techniques are classified into three types: task allocation method, load balancing among nodes, load sharing in between different nodes. The main goal of load balancing is to balance the workload among the nodes by reducing execution time, minimizing communication delays and efficient resource utilization and maximizing throughput [1].

2. RELATED WORK
In contextual environment, the performance of Load balancing may comprise of the selection of a node based on nodes social or physical context. Generally, the task allocation is always done considering the node`s self-owned resource distribution; the number of tasks allocated on a node is directly proportional to its self-owned resources, i.e., an node may be assigned with more tasks if it owns more resources by itself. On the other hand in context based systems, if a node does not own required resources by it, but it can get remaining resources for that task from other connected nodes easily. The number of allocated tasks on a node is not only proportional to its own resources but also the resources of other node`s interacting with it. [2].

Distributed hash table (DHT) systems are a new class of peer-to-peer networks which provide routing infrastructures for scalable information storage and retrieval. Such systems comprise a collection of nodes organized in an overlay network. As peers participating in a DHT are often heterogeneous, virtual servers are used to cope with the heterogeneity of peers. Participating peers in a DHT can host different numbers of virtual servers, thus taking advantage of

peer heterogeneity. Let VS be the set of virtual servers in the system and ND be the set of participating peers [3]. By reallocating virtual servers in VS to peers in ND suggest that each peer i $\in$ ND deal with the load balancing problem by minimizing the following:

$$\left| \frac{\sum_{VS \in VS_i} L_{VS}}{C_i} - A \right| \quad (1)$$

Where $VS \in VS_i$ set of virtual server allocated to peer I, L_{vs} load value of the virtual server $VS \in VS_i$, Ci is the capacity value of peer i.

Using Distributed hash table's (DHT's) virtual server can be designed to find suitable resources on a node for load assignments and proximity aware load balancing. If a given node has all the resources available for give task, execution minimum of task & load balancing overhead will be less otherwise the amount of load transfer and hence cost of load balancing will be more [4].

Considering the P2P system perspective, "efficiently" means getting fair load distribution among all peer nodes. Many solutions have been proposed to tackle the load balancing issue in DHT-based P2P systems [4]. However, existing load balancing approaches have some limitations; they either ignore the heterogeneity of node capabilities, or transfer loads between nodes without considering proximity relationships, or both.

In distributed load balancing if all the resources are located at the same site; there will not be any load transfers. However, for large-scale applications, where the resources may be distributed at different heterogeneous nodes, the load transfers may no longer be neglected. As a result, when any node fall out of resources its load status to be declared and load migration decisions should be made accordingly . As the nodes are heterogeneous the load on each node may change continuously. This will affect on the performance related to the load balancing algorithms. All the existing methods neglect the heterogeneity of nodes and load assignments considering the proximity of nodes on the accuracy of the load balancing solutions. Hence a comparative study of different load balancing algorithms considering context aware is discussed [5].

Also an overlay network scheme with Multitier Heterogeneity-Aware Topologies, discovered. The power of node heterogeneity is utilized by making the selection of super nodes and the construction of the topology more generic for increase in network utilization, minimizing the search latency, and getting better fault tolerance of the P2P system. The three classes of overlay topologies (Hierarchical topology, Layered Sparse topology, Layered Dense topology) that can be used to develop a Probabilistic-Selective routing technique employ a heterogeneity-aware routing algorithm [6].

A Local Minima Search (LMS) protocol designed for unstructured topologies inherits some properties of random walks, which is known to be the best way for improving searches on unstructured networks. Using Namespace virtualization LMS both peers and objects to identifiers in a single large space. In LMS, the replica of each object is placed on several other nodes. Similar to DHT, LMS places replicas onto nodes which have ID s "close" to the object but DHT, there is no mechanism to by which node routed to the closest item [7].

Many researchers have shown load balancing can be done with either using DHT or virtual servers. The type of environment in which the node is located should also be considered. While making the load balancing or load movement decision the proximity in between node helps us to make competent choice for selection of new node.

3. PROPOSED WORK

3.1 Problem definition

An Improved Load Balancing Scheme Using Node conditions considering contextual information of nodes in heterogeneous environment using the concept of load movement from in the form of virtual machine.

3.2 Highlights

Load Balancing: - The computing power of any distributed system can be realized by allowing its constituent computational elements (CEs), or nodes, to work cooperatively so that large loads are allocated among them in a fair and effective manner. Any strategy for load distribution among CEs is called load balancing (LB). An effective LB policy ensures optimal use of the distributed resources whereby no CE remains in an idle state while any other CE is being utilized. Effective utilization of parallel computer architecture requires the computational load to be distributed, evenly over the available CEs. Distribution of computational load across available resources is referred to as the *load balancing* problem.

Node Conditions: - The conditions of an agent can be simply regarded as the environment it is situated which includes the physical and the social environment (organizational one) [3]. The physical condition is produced by the agent's physical environment, which can be regarded as the agent's physical location, and the physically nearby agents within the subsystem; the resources owned by the agents within its physical environment are called the physically contextual resources [3]. On the other hand, agents in the complex system should be organized within some social organizations, so the counterpart agents in the social organizations can be regarded as the agent's social context, and the resources of the agents in the social context are called socially contextual resource [3].

Physical Context: - If an agent lacks the necessary resources to implement the allocated task (such agent are referred as initiator agent), it may negotiate with its physically contextual agents [3]; if the physically contextual agents have the required resources (call such agents that lend resources to the initiator agent as response agents), then the initiator agent and the response agents will cooperate together to implement such task [3]. The negotiation relations from agent to other agents within its physical context form a directed acyclic graph with single source a, which is called the physically contextual resource negotiation topology (PCR-NT) of agent a [3].

Social Context: - In the social organizations, it is more likely that the near individuals may have more similarities and, being closer together in the organizational hierarchy, share more common interests than the remote individuals [3] Therefore, in the social organizations of complex systems, each agent will negotiate with other agents for the requested resources gradually from near places to remote places [3]. Let 'a' be an agent that will negotiate with other agents within its

social context and the agents in the nth round of negotiation of agent 'a' be called the social contexts with gradation n [3].

Proposed System: - The proposed algorithm will take care of the social context of nodes in the network at the time of making load balancing decision. All the nodes in the network are heterogeneous in the sense of their functionality and (or) resources. For this purpose, virtual server concept is used which is not actually present on nodes it is just a notion of load transfer between nodes in network. Virtual server is used in cluster environment. Virtual server is the notion of load transfer and it can transfer active jobs from one node to another. In case of agent based approaches administrative control is required which will guide load transfer. Using virtual machine can handle security boundaries for job transfer. Rearranging the topology will reduce the load migration cost and bound the contextual similar nodes in one closure

3.3 Constraints and assumptions

1. We are assuming that the nature of the nodes in the network is cooperative (not competitive), which can support resource negotiation among nodes.

2. We are not considering the network delay parameter while selecting a particular node for load balancing; it is assumed that there is same delay for every connection in the network.

3. For synchronization of nodes we are assuming that the nodes are following the same time clock. This will effect to balance the load in network at particular time interval.

4. As the distance between contextually similar nodes can be anything we have to reconfiguration the network for socially contextual node and consider the cost for reconfigurations.

5. Presence of similar contextual information at more than one siteRoman in which these guidelines have been set.

4. RESEARCH METHODOLOGY

4.1 Steps to be carried out for project work.

1. Formation of network i. e. cluster formation.
2. Client and server to be assigned with some computation function and resources.
3. Authentication for the clients with detail configuration information.
4. Initialization of tasks at various server sites.
5. Discovering resource requirement for the tasks.
6. Defining context of tasks which can have type and number of resources required for execution.
7. Find the ideal node for task execution if no node satisfy task requirement, relocate task as per context.
8. While relocating tasks as per context priority of resources need to consider hence allocate tasks according to priority of resources..
9. Load transfer in the form of virtual server(containing required resources for that task)
10. Execution of tasks with time required for execution and comparison of results.

4.2 Proposed Algorithm

/* let a be the initiator agent, and the set of agents in a's social context be A, Tx: the subtree whose root is agent x in the hierarchical structure; Px: the parent node of x in the hierarchical structure. */

1) Set the tags for all agents in A to 0 initially;
2) b = 0;
3) At = {a}; /*the allocated agent set for task t*/
4) $R^t_{a\ =}$ Rt - Ra; /*The lacking resources of agent a to implement task t*/
5) If R^t_a = = {}, then b= 1; /*Agent a can provide all resources to implement task t*/
6) If (b == 0), then:
6.1) Negotiation (a, a); /*a negotiates with the agents within Ta using following steps
6.1.1. Gather load balancing information (LBI) in the form of < $L_{i,}$ C_i , $L_{i,\ min}$ > <total load on virtual server, capacity of a node i, minimum load of virtual server on node i>
6.1.2. Classify nodes as
A heavy node if $L_i > T_i$
A light node if $(T_i - L_i) >= L_{min}$
A neutral node if 0 <= if $(T_i - L_i) >= L_{min}$
6.1.3 For Virtual server assignment a heavy node chooses a (say i) chooses a subset of its virtual servers {$v_{i,1}$; . . . ; $v_{i,\ m}$} (m >= 1)
6.1.4 Upon receiving the paired VSA information the heavy node i will transfer the virtual server $v_{i,r}$ to the light node j.
7) If (b ==1), then Return (A_t) /*All resources for implementing t are satisfied*/
else return (False);
8) End.

In the social organizations, it is more likely that the near individuals may have more similarities and, being closer together in the organizational hierarchy, share more common interests than the remote individuals [3] Therefore, in the social organizations of complex systems, each agent will consult with other agents for the requested resources gradually from near places to remote places [3]. Consider an agent that will negotiate with other agents within its social context and the agents in the nth round of negotiation of agent a be called the social contexts with gradation n [3].In the hierarchical structures, each agent can interact directly only to its parents and children; thus, each agent will first negotiate with its parent or children for resources. However, in the hierarchical organizations, resource negotiation happens between pairs of agents that share the same immediate parents; and agents will always negotiate resources through the lowest common ancestor. Therefore, let there be an agent that can negotiate with other agents within the hierarchical structure according to the following orders [3]:

1. The subordinates of agent a in the hierarchical structure;
2. The immediate superior of agent a;
3. The sibling agents with the lowest common superiors

When CRM (context based task allocation model) is used, the allocated agents will always incline to be located within the near physical contexts or social contexts, so the communication time will be reduced; [3] however, the allocated agents in SRM (self owned resource based task allocation model) will always be distributed through the system, so the communication time among agents will incline to be more than the one of CRM model [3]. Therefore, the task execution time of CRM model is always less than the one of the SRM model and while the number of tasks increases, the communication time will also increase [3].

4.3 Mathematical Model

Let there be a network system.Total **N** nodes present in the system, set N= {N1, N2, N3, …..Nn}.Total **T** task present on system for execution on various nodes, set T= {T1, T2, T3,…Tn}. Various agents present on system, set **A**= {A1, A2, A3,….An}.The current state of the system is $\forall A(\exists T \in N)$.

Let, P is denoted for physical context of a group of agents.
S is denoted for social context of a group of agents.

$$\forall A\{\exists Ax / \forall A \in Ax, Ax \in P\} \text{ and}$$
$$\forall A\{\exists Ay / \forall A \in Ay, Ay \in S\}$$

Let R be set of resources on A, set R={ R1, R2, R3,….Rn}
Then, $\{R \in A / Ri \in Aj\}$ Ri ∈R and Aj ∈ A

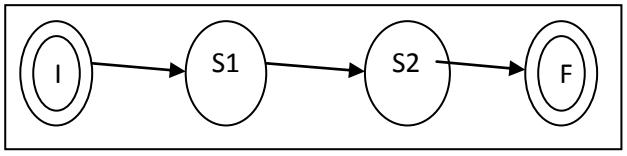

Fig. 1 State Diagram of System

I Initial state (network with client server)
S1 System with task allocated on various network nodes.
S2 System state with node resource negotiation process and virtual server operations.
F Output state (execution of tasks)

5. SYSTEM ARCHITECTURE

In the state of the system there are number of nodes present on network. Each node belongs to some of the network. On each node there are some agents present which initiate load balancing process. From Fig. 2 we can see different agents are present on nodes in the network. The proximity related information for all agents on nodes is considered. All the nodes are allocated with certain resources and computations. At the time of task execution each node will verify the required resources present on it. If they are not present the nodes need to negotiate with other nodes considering their context physical and social.

Each node in turn is provided with virtual machine (VM). A VM, like a PM (physical machine), has associated resource levels of CPU, memory, and input/output (I/O) devices. While instantiating VMs, each machine needs to be provisioned/allocated these resources. These resources can be overcommitted, and multiplexing them across VMs is the hypervisor's responsibility. Typically, a "sizing" process, based on resource-usage profiles of applications or estimations to meet load requirement and so on, is used to determine initial provisioning-levels of a VM. Changes in workload conditions of VMs can lead to "hot spots" — not enough resources provisioned to meet demand or "cold spots" — provisioned resources are not utilized efficiently. In each case, resource allocations are varied dynamically to address the situation under consideration.

Each virtual machine can be transferred from one node to other to satisfy resource requirement of the nodes. Before transfer one need to identify certain parameters of nodes like

A heavy node if $L_i > T_i$.

A light node if $(T_i - L_i) >= L_{min}$.

A neutral node if 0 <= if $(T_i - L_i) >= L_{min}$.

Where Li denote the sum of the loads of all virtual servers on a DHT node i, and C_i represent the capacity of a DHT node i. The target load proportional to its capacity can be calculated as

$$T_i = (L/C + E)\, C_i. \quad (5)$$

Virtual server assignment and transfer process continues till task execution.

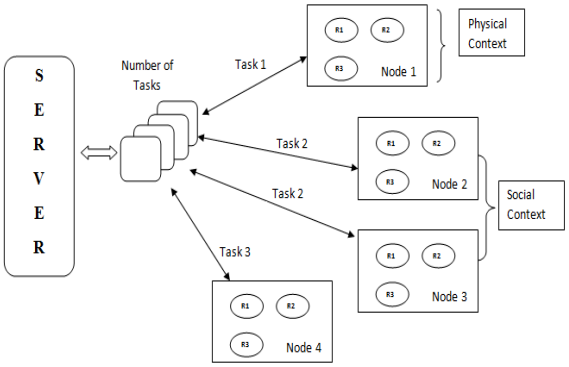

Fig. 2 System Architecture

6. EXPERIMENTAL RESULTS

This section, will validate the correctness of proposed algorithm, so a comparison of the performances of the following two models is presented.: 1) physical contextual resource-based load balancing mode (PCR-LB). 2) Social contextual resource based load balancing model (SCR-LB). In the (PCR-LB), the task allocation is implemented according to the physical contextual resources of nodes. In the (SCR-LB), the task allocation is implemented according to the social contextual resources of nodes.

Now, let there be a task t, the set of resources called by task t is R_t { N_1R_1; N_2R_2; NnRn} N^j_i denotes the number of R_i owned by node Aj, R_i denotes the set of resources owned by node A_i. Then, after a series of case simulations to test the two models. For example, let task t1 be allocated to "p_{44} ($3r_1$, $1r_3$), p_{22} ($3r_2$), p_{43} ($1r_2$)," so the execution of t_1 is gi given as follows:

At first, t1 obtains"3r1, 1r3" from p_{44}, then task is sent to p_{22}, then resources "3r1, 1r3" are released; 2) now, p_{22} receives the task and donate its resources. Task t1 obtains "3r2" from p_{22}, the task is sent to p_{43}, then resources "3r2" are released; 3)

now, p_{43} receives the task when t1 obtains "1r2" from p_{43} now the task is finished, and resources "1r2" are released. Therefore, let GTp_i denote the time of "getting required resources" from node p_i,

$ctp_i \rightarrow p_j$ denote the communication time from p_i to p_j then the execution time of t1 is

$$Et_1 = GTp_{44} + ctp_{44} \rightarrow p_{22} + GTp_{22} + cta_{22} \rightarrow a_{23} + GTp_{43} \quad (6)$$

From above equation, one can distinguish that the execution time of a task is a combination of the resource access time (i.e., the waiting time of the task) and the communication time among allocated nodes. Therefore, the task time of "getting resource" is more which makes the execution time of task longer. Other hand, if the distance among nodes is more, the time of "communication time" will increase, and that also results in increase execution time of task.

6.1 Task Execution Based on Contextual Enrichment Factor

Let there is a node a_i, the set of nodes within its physical context is PC_i and the set of nodes within its social context is SC_i. Obviously, every node within PC_i or SC_i will contribute differently to the resource predominance of ai; the contribution of an node within PC_i or SC_i to node a_i is determined by the physical or social distance between such node and ai. Considering the hysically contextual enrichment factor and socially contextual enrichment factor of a node, the superior its contextual resource enrichment factor to one sort of resource is, the superior is its chance to gain an adequate amount of such type of tasks in the system.

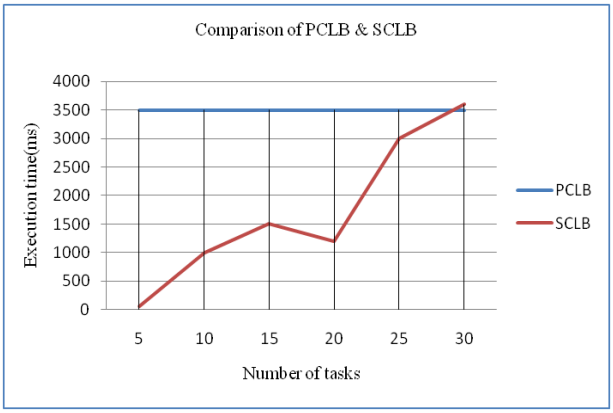

Fig.3 Comparison of PCLB & SCLB

From the results, as shown in fig. 3 the findings are as follows: 1) when CNRM model is used, the allocated nodes will always bring round to be located within the near physical contexts or social contexts, so the communication time will be reduced; however, the allocated nodes in SONRM model will always be distributed through the system, so the communication time among nodes will incline to be more than the one of CNRM model.

Therefore, the task execution time of CNRM model is always less than the one of the SONRM model and 2) while the number of tasks increases, the communication time will also increase; so the difference between the two models will increase accordingly. Therefore, the conclusion is that the CNRM model can reduce the total communication time among allocated nodes so as to reduce the total task execution time compared to SONRM, especially while the number of tasks is large.

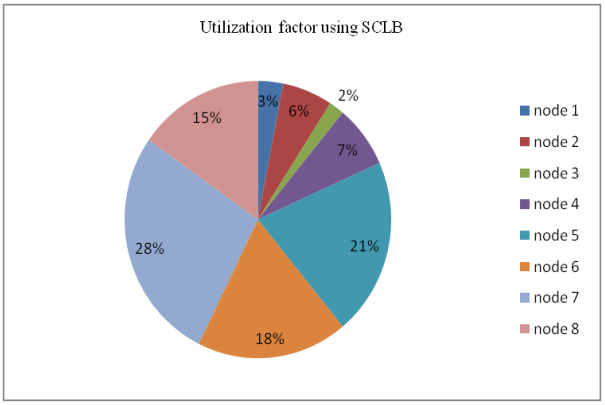

Fig.4 Utilization factor using SCLB

Our algorithm works well with the utilization of nodes. If the SCLB technique is used at some instance of time we are able to present the status of each node in terms of how many resources are in use. Fig.4 shows the results for the utilization factor of the nodes which are currently engaged in executing tasks. Utilization factor is unstable all the time and will be updated at every request. As shown in fig. 5 the utilization factor can be presented at any time but the status of all connected nodes will not be there. At every request what so ever be the nodes involved for task execution, they utilize the resources in some quantity and hence that will the utilization of these nodes.

When the performances of our load balancing algorithms are checked against physical and social contexts we found that when there are 25 nodes connected with our host and random number of tasks are initiated by the host the performances of PCLB is constant while that of SCLB improves with more number of nodes. The simulation result is as shown in fig.5.

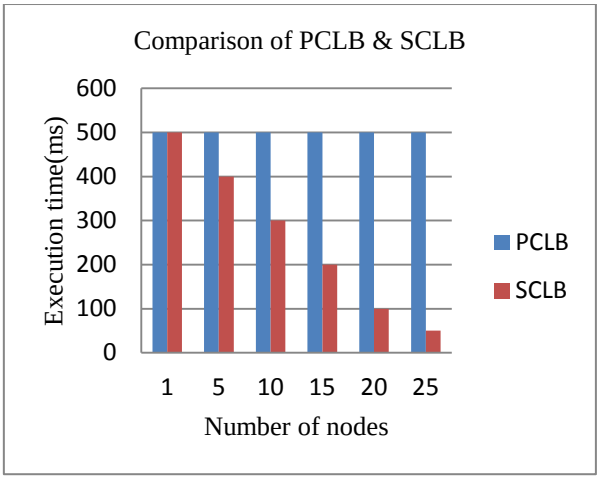

Fig.5 Comparison of PCLB & SCLB

7. CONCLUSION

In this paper we studied contextual load balancing techniques for distributed systems considering the homogeneous and heterogeneous environment of network. The task allocation and load balancing can be done based on contextual resource negotiation which outperforms the previous methods based on the self-owned resource distribution of nodes.

Relocation of nodes in network according to their social context is possible and hence it reduces the resource migration cost within the social context.

Through the experimental study it is observed that the model can work well especially while the task number is large. Therefore, the idea presented in this paper can be used to develop real distributed system. Moreover, we recognize that how to make trade-off between physical and social negotiations should be explored in our future work.

8. REFERENCES

[1] Ardhendu Mandal and Subhas Chandra Pal "An Empirical Study and Analysis of the Dynamic Load Balancing Techniques Used in Parallel Computing Systems" ICCS-2010, 19-20 Nov, 2010.

[2] Yichuan Jiang and Jiuchuan Jiang, Senior Member, IEEE "Contextual Resource Negotiation based Task Allocation and Load balancing in complex software systems" IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 20, NO. 5, MARCH 2009.

[3] Hung Chang Hsiao, Member, IEEE Computer Society, Hao Liao, Ssu-Ta Chen, and Kuo- Chan Huang "Load Balance with Imperfect Information in Structured Peer-to-Peer Systems" IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 22, NO. 4, APRIL 2011.

[4] Yingwa Zhu and Yiming Hu,. *Senior Member, IEEE* "Efficient Proximity Aware Load Balancing For DHT-Based P2p Systems" IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 16, NO. 4, APRIL 2005.

[5] Pooja Gandodhar, Sudarshan Deshmukh Dept. of Computer Engg. PCCOE, Pune "A Comparative Study of Different Load Balancing Techniques for Heterogeneous Nodes", *Proc. of Int. Conf. on Advances in Communication, Network, and Computing 2013*.

[6] "Large Scaling Unstructured Peer-to-Peer Networks with Heterogeneity-Aware Topology and Routing" Mudhakar Srivatsa, Student Member, IEEE, Bugra Gedik, Member, IEEE, and Ling Liu, Senior Member, IEEE, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 17, NO. 11, NOVEMBER 2006

[7] Ruggero Morselli, Bobby Bhattacharjee, Michael A. Marsh, and Aravind Srinivasan "Efficient Lookup on Unstructured Topologies" IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, VOL. 15, NO. 1, JANUARY 2007.

[8] George Colouris, Iean dollimore an tim Kinderberg "Distributed system Concepts and Design" Fourth edition.