Volume 4, Issue 2, February 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Improved Context Aware Load Sharing for Grid Environment

Dr. Sanjay Tejbahadur Singh
Professor, P.K. Technical Campus,
Chakan, Pune, India

Abstract*— In distributed system tasks can be scheduled on nodes based on number of resources present on them, Load balancing that is resource allocation on such environment is referred as Self owned resources based load balancing. Due to the deficiency of resources most of the times the nodes need to negotiate with other nodes in network for resources. Hence the capacity of a node not only depends on itself but also on the other contextual nodes. The number of tasks allocated on a node is directly proportional to its physical and social contextual resources. Hence load balancing becomes more challenging in such environment. In today's world since most of the task execute on heterogeneous environment resource negotiation becomes more complex. Most of the existing load-balancing schemes ignore network proximity and heterogeneity of nodes. Hence we have proposed a load balancing scheme which considers proximity of nodes under physical and social contextual resources on heterogeneous environment. Our results shows more than 20% reduction in communication costs between allocated nodes than the previous methods based on self-owned resource distribution of nodes.

***Keywords*— Context based load balancing, distributed system, proximity aware load balancing.**

I. INTRODUCTION

Advancement in computer networking technologies have led to increase interest in the use of large-scale parallel and distributed computing systems. Distributed systems are gaining popularity by one of its key feature: resource sharing.
A distributed system consists of independent workstations connected usually by a local area network. Users of the system put forward jobs to their computers at random. In such a system some computers are heavily loaded while others are available with processing capacity. A Load balancing algorithm tries to balance the total systems load by transparently transferring the workload from heavily loaded nodes to lightly loaded nodes in an attempt to ensure good overall performance relative to some specific metric of system performance [8]. Effective load balancing algorithms are used to distribute the processes/load of a parallel program on multiple hosts to achieve goal(s) such as minimization of execution time and communication delays with efficient resource utilization and maximized throughputs [1].

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. Load balancing focuses on 1) improving system performance at a reasonable cost by reducing task response considering various delays. 2) Fault tolerance 3) System stability 4) Resource multiplicity etc.

Load Balancing: - The computational capacity of any distributed system is a combination of allowing its underlying computational elements (CPEs), or nodes, to work cooperatively so that large loads are allocated among them in a fair and effective manner. Any technique for load distribution among CPEs is called load balancing (LB). A successful LB technique ensures best possible use of the distributed resources whereby no CPE remains in an inactive state while any other CPE is being used for a task execution. Valuable utilization of parallel computer architecture requires the computational load to be circulated, equally over the available CPEs. The process of distribution of computational load across variable resources is called to as the load balancing problem.

The rest of the paper is organized as follows. Section 2 summarizes related work based on DHT, LMS, and virtual server load balancing. In Section 3, we identified problem by studying different load balancing methods and found assumptions and constraints of the problem. In Section 4, we presented the research methodology with detail analysis of the problem. In Section 5, we discussed mathematical model for the proposed system. In Section 6, we provide the proposed algorithm and the architecture of the system. Section 7 represents the simulation results to validate our proposed model. Finally we conclude this paper in section 8.

II. 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 $\in$ VSi set of virtual server allocated to peer I, Lvs load value of the virtual server VS $\in$ VSi, 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 [10]. 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.

III. PROPOSED WORK

A. Problem Statement

An improved and efficient load balancing scheme in cluster networks considering contextual information of nodes in heterogeneous environment.

- Context based load balancing: - The context of a node can be simply described as the surroundings in which it is situated which includes the physical context and the social context (organizational one). The physical context is formed by the node's physical surroundings, which can comprise of the node's material location, and the physically nearby nodes within the network; the resources owned by the node inside its physical context are called the physically contextual resources. On the other hand, nodes in the network system should be controlled within some social organizations, so the counterpart nodes in the social organizations can be regarded as the node's social context, and the resources of the nodes in the social context are called socially contextual resource. Therefore, each node in the complex system may have two kinds of contexts, one is the physical context, and the other is the social context [20], [21], [22].
- Physical context:-If a node lacks the necessary resources to implement the allocated task (we call such node as initiator), it may negotiate with its physically contextual node; if the physically contextual node have the required resources (we call those nodes that lend resources to the initiator as response nodes), then the initiator and the

response node will cooperate together to implement such task. The negotiation relation from node to other node within its physical context form a directed acyclic graph with single source ‘a’, which is called the physically contextual resource negotiation topology (PCRS-NT) of node a.

- 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[19]. Therefore, in the social organizations of complex systems, each node will negotiate with other node for the requested resources gradually from near places to remote places. If ‘a’ is a node that will negotiate with other nodes within its social context and the nodes in the nth round of negotiation of node ‘a’ be called the social contexts with gradation n.
- Existing System: - The existing context based algorithms are based on homogeneity of network and they ignore distance between nodes.
- 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. Using virtual machine we 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.

B. Objective and Scope

Generally, the performance of LB in contextual environment depends upon the selection of a node based on nodes social or physical context. In a practical scenario if a given node has all the resources available for task execution minimum load balancing overhead will incur otherwise the amount of load transfer will be more.

In distributed load balancing if all the resources are located at the same site; the load transfers may be negligible. However, for large-scale, where the resources may be distributed across 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 capacity of each node will differ. This will affect the accurateness, and the performance, of the load balancing algorithms.

In any network number of tasks exists. All the nodes will have their own tasks, which they alone can execute or need someone`s help. All these network nodes differ in number and type of resources present on them. For some tasks to be executed the resources need to be transferred from one node to another i.e. negotiation takes place.

The proposed algorithm helps to select idle candidate node for resource migration for load balancing. This algorithm can reside on server and can cover number of clients under that server. Any application can have tasks to be executed. These tasks are send to scheduler first, scheduler will find cluster which is available on time migrate task to that cluster. Cluster will gather load information, status, and resource information from the nodes under it. Once it gathers all these information cluster coordinator will allocate active jobs to respective node. Once the tasks executed on nodes results will be sent back to application.

C. Constraints

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 site.

D. Comparative Analysis of Different Approaches

Table 1 list down comparison of a variety of load balancing algorithms based on certain attributes. We present assessment of three IEEE transaction papers.

- The environment of nodes is either homogeneous or heterogeneous.
- The criterion for load balance is Node based as well as DHT based. DHT based techniques uses the concept of virtual which are lightweight and helps in fair load distribution considering the amount of load on nodes (heavy/ light weighted).

Table 1. Comparison of different load balancing techniques based on a few attributes.

- The criterion for load balance is Node based as well as DHT based. DHT based techniques uses the concept of virtual which are lightweight and helps in fair load distribution considering the amount of load on nodes (heavy/ light weighted).
- A DHT technique points the proximity of nodes while using node’s technique load movement is considering the context of nodes. Who so ever possesses more similar resources will be chosen as candidate for sharing that load.

- The nodes which participate in load balancing process should be cooperative not competitive and someone should be the controller on the top of others.
- A fair load balancing technique has to be devised, which will handle all the above issues.

Table1: Comparative Analysis

Attributes	"Contextual Resource Negotiation based Task Allocation and Load balancing in complex software systems"	"Efficient Proximity Aware Load Balancing For DHT-Based P2P Systems"	"Large Scaling Unstructured Peer-to-Peer Networks with Heterogeneity-Aware Topology and Routing"
Environment	Unstructured networks and homogeneous	Unstructured networks and homogeneous	Unstructured networks and heterogeneous
Approach	Node based	DHT based	DHT based
Technique	Node negotiation based on number of resources owned by it.	Creation of virtual server using k-ary tree structure.	Heterogeneity class aware topology and routing.
Performance	Reduces communication and execution time than self owned resource based load balancing.	Fair load distribution increases efficiency of system.	Higher efficiency than homogeneous networks with reduction in bandwidth consumption
Points to be considered for future work	All nodes are assumed of same computational power.	Proximity information is considered only for VSR.	Construction and maintenance of topology.
	What will be the nature of an node for negotiation? (competitive / cooperative)	How and who will allocate the load to a virtual server (K-ary root).	State will change number of times.
	Is there any controller on top of all nodes for defining physical and social context?	The VST unavoidably causes the k-ary tree to restructure each time load movement from heavy to light node.	

IV. RESEARCH METHODOLOGY

Let Pi be the initiator node, the resources owned by Pi is RSpi. Now consider task t is scheduled on Pi and number of resources required for implementing t is RSt. Therefore the node lacks in number of resources for task t as:

$$*RS^{t}_{pi} = RS_{t} - RS_{pi} \quad (2)$$

Node Pj is negotiated by Pi. Node Pj having the set of resources as RSpj. During negotiation the set of resources node Pj can donate to Pi is:

$$RS_{tPj \to Pi} = \{ r \mid r \in RSP_{i} \wedge r \in *R_{tPi} \} \quad (3)$$

Hence the set of lacking resources of Pi to implement task t is reduced to:

$$*RS^{t}_{pi} = *RS^{t}_{pi} - RS_{t\,Pj \to Pi} = *RS_{t_{pi}} - RS_{pj} \quad (4)$$

The initiator node Pi negotiates according to the PCRS-NT/SCRS-NT, until all requested resources are satisfied with the physically/social contextual nodes.

The hierarchical structures are in the form of a tree where each node can have parents and children with which a node can directly interact for negotiation. Let there be a node p in hierarchical structure as shown in fig. 2 that can negotiate with other nodes according to the following orders:

1. The children of node p in the hierarchical structure;
2. The immediate parent of node p;
3. The sibling nodes with the lowest common superiors.

Let node p21 be the initiator node, in fig. 3 an SCR-NT according to the above social negotiation orders in the hierarchical structure is represented.

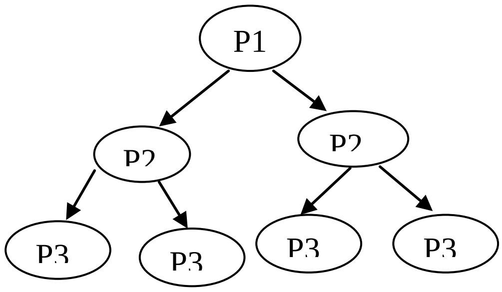

Fig. 1 Topology in hierarchical structure

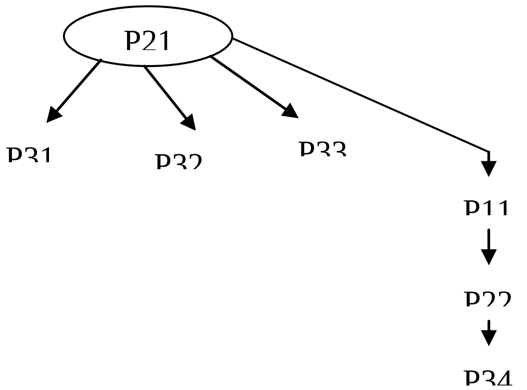

Fig. 2 SRC-NT of node P_{21}

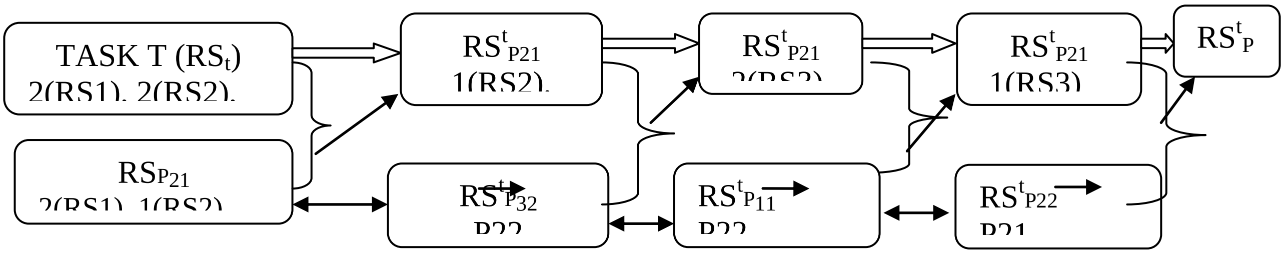

Fig. 3. SRC-NT of node P21

Resources → / Nodes ↓	RS-1	RS-2	RS-3
P11	2	0	1
P21	2	1	2
P22	1	2	1
P31	1	0	0
P32	2	1	0
P33	2	0	0
P34	1	2	1

Fig. 4 Number of resources owned by nodes

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

V. MATHEMATICAL DESIGN

Set N= {N1, N2, N3 …..Nn}
Total T task present on system for execution on various nodes,
Set T= {T1, T2, T3…Tn}
Various nodes present on system represented by,
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 nodes.
S is denoted for social context of a group of nodes

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

Let R be set of resources on A,
Set R= {R1, R2, R3….Rn}

$$\{R \in A / Ri \in Aj\}$$ Ri $\in$ R and Aj $\in$ A

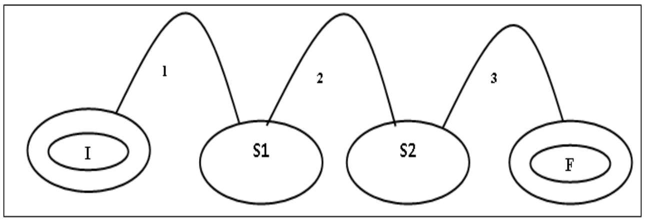

Fig.5. State diagram

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)

VI. PROPOSED ALGORITHM

Let p be the initiator node, and the set of nodes in p's social context be A, now the socially contextual resource negotiation process of node a in hierarchical structures is as shown in following algorithm.

/* consider node p as the initiator node, and the set of nodes in p's social context be A, Tx: the subtree whose root is node x in the hierarchical structure; Rx: the parent node of x in the hierarchical structure. */

1) Set the tags for all nodes in A to 0 initially;
2) b = 0;
3) At = {a};
/*the allocated node set for task t*/
4) Rtp = Rt - Rp;
/*The lacking resources of node p to implement task t*/
5) If Rtp = = {}, then b= 1;
/*Node p can provide all resources to implement task t*/
6) If (b == 0), then:
6.1) Negotiation (p, p);
/*a negotiates with the nodes within Ta using following steps*/
6.1.1. Gather load balancing information (LBI) in the form of < Li, Ci , Li, 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 Li > Ti
A light node if (Ti- Li)>= Lmin
A neutral node if 0 <= if (Ti- Li)>= Lmin
6.1.3 For Virtual server assignment a heavy

node chooses a (say i) chooses a
subset of its virtual servers {vi,1; . ; vi,m}
(m >= 1)
6.1.4 Upon receiving the paired VSA
information the heavy node i will transfer the virtual server
vi,r to the light node j.
7) If (b ==1), then Return (At)
/*All resources for implementing t are satisfied*/
else return (False);
8) End.

VII. SYSTEM ARCHITECTURE

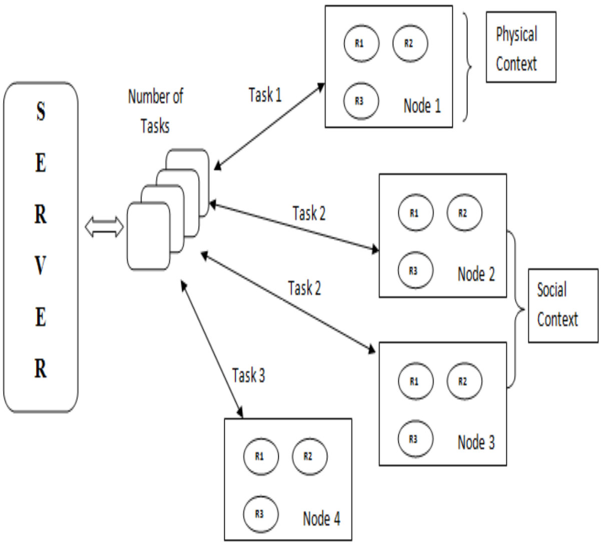

Fig.7. System Architecture

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 nodes present which initiate load balancing process. We can see from the figure 7 System architecture different nodes are present on nodes in the network. We have assumed proximity related information for all nodes. 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 physical and social context.

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 we have to identify certain parameters of nodes like

- A heavy node if Li > Ti.
- A light node if (Ti- Li)>= Lmin.
- A neutral node if 0 <= if (Ti- Li)>= Lmin.

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

$$T_i = \left(L / C + E\right) C_{i.} \qquad (5)$$

Virtual server assignment and transfer process continues till task execution.

VIII. SIMULATION

In this section, we will validate the correctness of proposed algorithm, so we will compare the performances of the following two models: 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 Rt { N1R1; N2R2; NnRn} Nji denotes the number of Ri owned by node Aj, Ri denotes the set of resources owned by node Ai. Then, we will make a series of case simulations to test the two models. For example, let task t1 be allocated to "p44 (3r1, 1r3), p22 (3r2), p43 (1r2)," so the execution of t1 is gi given as follows:
At first, t1 obtains"3r1, 1r3" from p44, then task is sent to p22, then resources "3r1, 1r3" are released; 2) now, p22 receives the task and donate its resources. Task t1 obtains "3r2" from p22, the task is sent to p43, then resources "3r2" are released; 3) now, p43 receives the task when t1 obtains "1r2" from p43 now the task is finished, and resources "1r2" re released. Therefore, let GTpi denote the time of "getting required resources" from node pi, $\mathrm{ctp_i} \rightarrow p_j$ denote the communication time from pi to pj then the execution time of t1 is

$$\mathrm{Et}_1 = \mathrm{GTp}_{44} + ctp_{44} \rightarrow p_{22} + \mathrm{GTp}_{22} + cta_{22} \rightarrow a_{23} + \mathrm{GTp}_{43} \quad (6)$$

From above equation, we 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.

8.1 Task Execution Based on Contextual Enrichment Factor
Let there is a node ai, the set of nodes within its physical context is PCi and the set of nodes within its social context is SCi. Obviously, every node within PCi or SCi will contribute differently to the resource predominance of ai; the contribution of an node within PCi or SCi to node ai is determined by the physical or social distance between such node and ai. Now, we have the concepts of physically contextual enrichment factor and socially contextual enrichment factor.

The physically contextual enrichment factor of node ai for resource rk is

$$\alpha_{\mathrm{i}}(\mathrm{k}) = \sum_{a_j \in PCi} (n_j(k) \bullet \frac{1/dp_{ij}}{\sum_{a_j \in PCi} 1/dp_{ij}}) \quad (7)$$

The social contextual enrichment factor of node ai for resource rk is

$$\beta_{\mathrm{i}}(\mathrm{k}) = \sum_{a_j \in SCi} (n_j(k) \bullet \frac{1/ds_{ij}}{\sum_{a_j \in SCi} 1/dps_{ij}}) \quad (8)$$

For 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.

8.2 Comparison between Self-Owned Resource Negotiation Based Load Balancing (SONRM) and Contextual Resource Negotiation Based Load Balancing (CNRM)

In this section we compare SONRM and CNRM. In the SONRM, the task allocation is implemented according to the self-owned resources of nodes, which is always used in the previous related work. At the moment, we introduce the SONRM model briefly. When a task requires more than one resource, we should
assign a node to act as the initiator (node within the negotiation), which will initially negotiate with other nodes in its proximity for the resources required by the task; then, the set of all nodes that provide resources for the task is the allocated nodes. To determine the initiator node, we use some criterions as follows
First Required Resource-First Satisfy (FRFS).

When a task requires more than one resource, it may call them one by one that is sequentially. In FRFS, the node that has the highest enrichment factor for the first called resource should be allocated as the initiator one of that task. Let t be a task, the set of resources called by task t orderly is {N1R1, N2R2… NnRn} where Ni denotes the number of required resources Ri. Hence we will allocate task t to the node Pj, which has the highest enrichment factor for Ri and so on.
8.1.1.2 Most Important Resource-First Satisfy (MIFS).
When a task requires many resources, there may be one resource that is the most important for implementing that task. If t is a task, the set of resources called by task t orderly is {N1R1, N2R2… NnRn} if R1 is the most important resource for implementing task t, we will allocate t to node Pj, which has the highest enrichment factor for resource R1.
All Resources-Averagely Satisfy (AAS).
When a task t requires resources Rt, we can allocate the task to

the node, which has the highest average contextual enrichment factor for all resources in Rt.

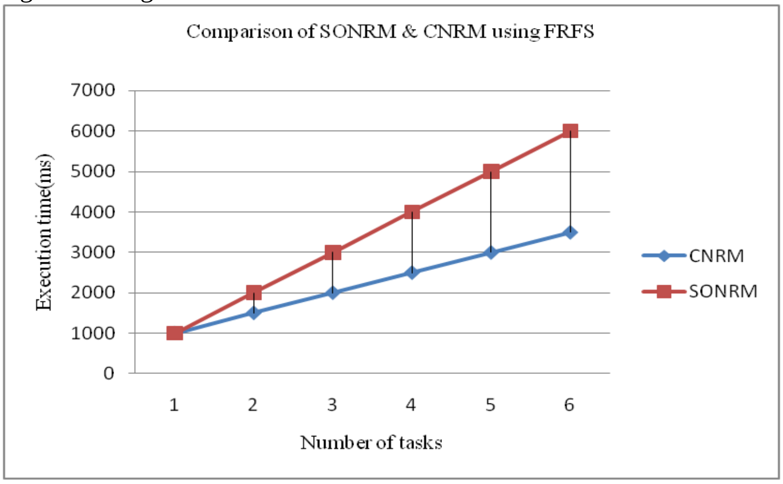

Fig. 8. Comparison of SONRM & CNRM using FRFS

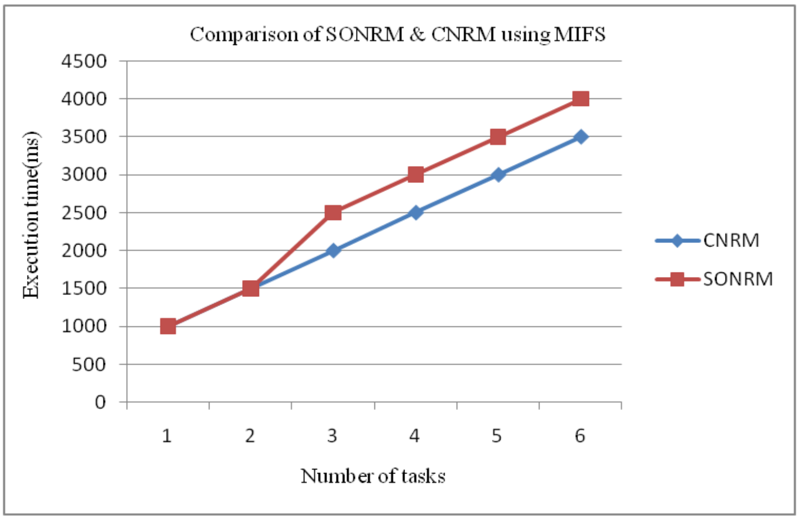

Fig. 9. Comparison of SONRM & CNRM using MIFS

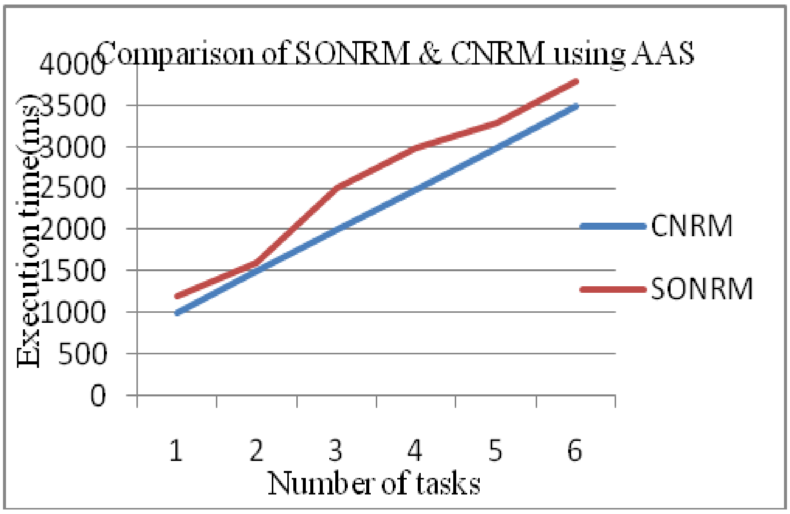

Fig. 10. Comparison of SONRM & CNRM using AAS

Then, we make a series of experiments to test the above two models. The node number is 50, the physical distribution and social structure can be set randomly. We can use SONRM and CNRM models to make task allocation; the allocated nodes will execute the tasks, then the total execution time of tasks is tested. The results are shown in Fig. 8, 9 and 10. From all results we can see that CNRM outperforms SONRM; as the task number increases, the CNRM will do better than SONRM. Therefore, our load balancing model can make an effective load balancing and reduce the task waiting time for resources while there are multiple tasks in the systems.

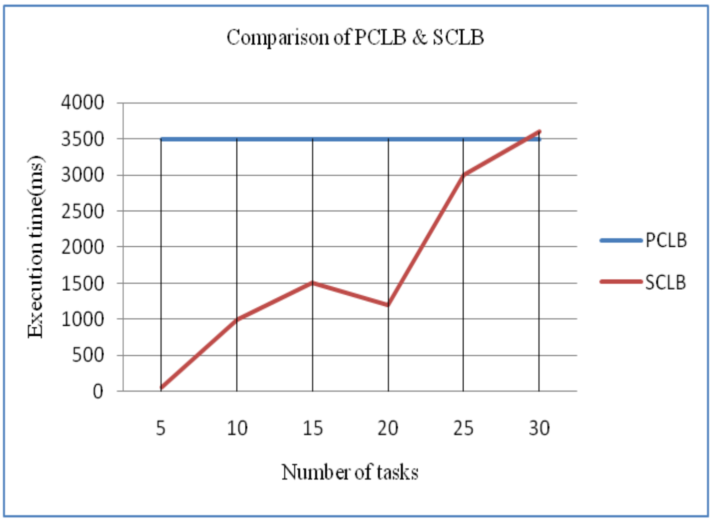

Fig. 11 Comparison of PCLB & SCLB

From the results, as shown in fig. 11 we can obtain the following: 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, we can conclude 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.

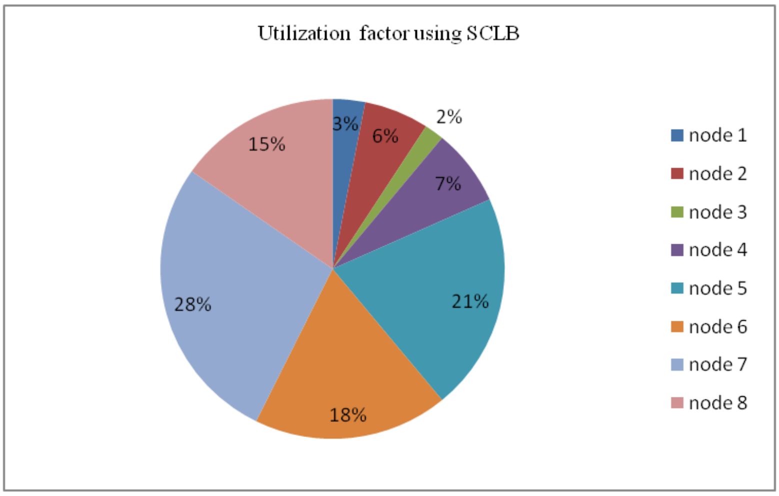

Fig. 12 Utilization factor using SCLB

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. 12 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. 12 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.

IX. 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 we can see 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.

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, Addison Wesley.

[9] Y. Chawathe, S. Ratnaswamy, L. Breslau, N. Lanham, and S. Shenker, "Making Gnutella- Like P2P Systems Scalable," Pr oc. ACM SIGCOMM, 2003.

[10] N.R. Jennings, "An Node-Based Approach For Building Complex Software Systems," Comm ACM, vol. 44, no. 4, pp. 35-41, 2001.

[11] Chiara Francalanci, Paolo Giacomazzi, and Alessandro Poli "Cost–Performance Optimization of Application- and Context-Aware Distributed Infrastructures" IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART A: SYSTEMS AND HUMANS, VOL. 39, NO. 6, NOVEMBER 2009.

[12] Jesse S.A. Bridgewater, P. Oscar Boykin, Member, IEEE, and Vwani P. Roychowdhury "Balanced Overlay Networks (BON): An Overlay Technology for Decentralized Load Balancing "IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 18, NO. 8, AUGUST 2007.

[13] Haiying Shen, Member, IEEE, and Kai Hwang, Fellow, IEEE "Locality-Preserving Clustering and Discovery of Resources in Wide-Area Distributed Computational Grids" IEEE TRANSACTIONS ON COMPUTERS, VOL. 61, NO. 4, APRIL 2012.

[14] Adrian Holzer, Patrick Eugster, and Benoˆıt Garbinato "Evaluating Implementation Strategies for Location-based Multicast Addressing" IEEE TRANSACTIONS ON MOBILE COMPUTING.

[15] RAMACHANDRA KOTA, NICHOLAS GIBBINS, and NICHOLAS R. JENNINGS, "Decentralized Approaches for Self-Adaptation in Node Organizations" ACM Transactions on Autonomous and Adaptive Systems, Vol. 7, No. 1, Article 1, Publication date: April 2012.

[16] Niranjan G. Shivaratri, Phillip Krueger, and Mukesh Singhal Ohio State University "Load Distributing for Locally Distributed Systems" Reprinted from *Computer,* vol. 25, no. 12, Dec. 1992, pp. 33-44.

[17] HAROLD S. STONE, MEMBER, IEEE "Multiprocessor Scheduling with the Aid of Network Flow Algorithms" Reprinted from *IEEE Trans, on Software Eng.,* vol. SE-3, no. 1, Jan. 1977, pp. 85-93.

[18] THOMAS L. CASAVANT, MEMBER, IEEE, AND JON G. KUHL, MEMBER, IEEE, "A Taxonomy of Scheduling in General-Purpose Distributed Computing Systems" Reprinted from *IEEE Trans, on Software*

