International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
Volume 6, Issue 12, December-2017, ISSN: 2278 – 1323

Improved DHT Based Task Allocation Strategy with Least Migration and Adaptive Load Balancing Approach

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

The author declare that there is no conflict of interest regarding the publication of this paper

***Abstract*—Solutions proposed for load balancing using virtual server assignment consider physically and contextually aware decisions. None of these solutions considered the influence of heterogeneous environment and the proximity factor for load balancing. We are proposing a novel scheme for task assignment based on proximity aware load balancing mechanisms. A self-organized k-ary tree constructed over DHT and load balancing is achieved by aligning two skews in load distribution such that nodes with higher capacity will carry heavier loads. Proximity aware information will be used to guide virtual server assignment among nodes for load balancing such that virtual server assignment will be physically close to heavy nodes among heavy and light loaded nodes.**

***Index Terms*— Contextual resource allocation, Load Balancing, Distributed hash table.**

I. INTRODUCTION

In **complex software systems,**interactions are involved among different parts to execute a task. All those interactions are carried out and managed by a collection of agents called software agents. These agents are responsible for two aspects: task allocations and load balancing.During task allocations, they will allocate tasks to some software agent to maximize performance of the system and to fulfill the resource requirement of the respective task.Earlier it was stated that in complex software, there are software agents which execute tasks by themselves and donot involve contextual agents.[1][2]. Number of tasks allocated to an agent is directly proportional to self owned resourecs.Task allocation was based on self owned resource distribution.But this approach is inefficient as itdoesnot take conetxtual agents into account.

This approach calculates only its own capability for task execution and doesnot involve contextual agents to take part in it. This problem is addressed by many theories based on contextual resource negotiation for task allocation.These approaches estimate resource capacity to execute task by involving contextual agents as well as the node itself to whom the task is submitted.Resource owned by agent within the contexts of agent are called **contextual resources [3][4]**.Therefore number of tasks allocated to an agent is directly proportional to self owned resources[4] as well as resource involved by contextual agents.Now**task allocation[6][7]** is based on self owned **resource distribution[8][9]** as well as resource owned by contextual agents.

Consider Fig. 1, task t which require resource $\{r_1,r_2,r_3\}$ and node a_1 with resource $\{r_1,r_2\}$ and node a_2 with resource $\{r_1\}$.Task will be allocated to node a1 according to approach in which contextual agents are not considered.But if node a_2 can share $\{r_2,r_3\}$ with other agents then task t will be allocated to a_2

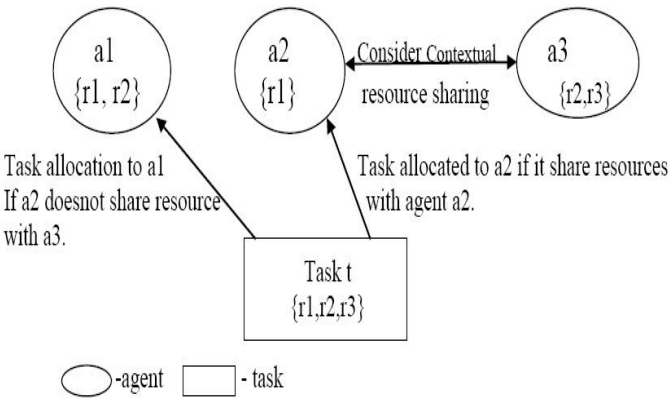


Fig. 1 Context Aware Task Allocation.

There are two contexts in contextual **resource negotiation[10]** based task allocation approach: physical context and social context,wherein the former approach considers physical parameters like network distance for allocating the task, the later considers the matching index factor of resources required for executing the task completely.There are two types of agents in physical and social context. Initiator Agent(IA)and Response Agent(RA).The former are the agents resposible for task implemetation,first one to which task is allocated.The later are the agents with resource set responsible to implement task with intiator agent or lend resources to initiator agent.For each task there will be different IA as well as RA.

DHT based system such as **chord [11], tapestry [12]** offer DHT for object storage and retrieval.It was assumed that nodes are homogenous in nature with respect to its bandwidth and storage capacity. But it leads to some limitation such that **Hash**

function[13] which produces ids of object is not able to provide perfect load balancing which result into *O (logN)* imbalance factor at peer node. Secondly,**homogenous overlay structure [14]** ignores the heterogeneity of the system as it was found recently that node properties are much skewed in system such as gnutella because it is usedto harness all resources with nodes efficiently.

Most of the solutions proposed for load balancing strategy are not aware of proximity factor. Therefore in this paper we have used proximity aware load balancing scheme which will be used to guide **virtual server [15][16]** assignment for load balancing. By using proximity aware load balancing scheme leads to the following advantage: it will efficiently utilize all resources of nodes. It will avoid bandwidth consumption dedicated to load movement. It will avoid high latency link while transferring loads between heavy loaded and lightly loaded nodes.

II. LITERATURE SURVEY:

A. Load Balancing

Why load balancing is required:

Consider the tasks $\{t_1,t_2,t_3,t_4\}$ require r_1 to execute and FIFS policy is considered.Enrichment factor for agents $\{a_1,a_2,a_3\}$ are *3.1,2.4,2.2* respectively.Since enrichment factor for resource $\{r_1\}$ is more for agent a_1 so agent a_1 will be initiator agent for task t_1.But due to allocation of task to agent a_1 there will be decrease in enrichment factor for resource r_1 in agent a_1 and in its contextual agent.But enrichment factor will be same for task t_2. It remains as *3.1,2.4,2.2*.After allocation of task t_2 ,it will be *3.1,2.4,2.2*.After allocation of task t_3 it will be *3.1,2.4,2.2*.For task t_4 it will be alloacted to agent a_1 if requirement for resource is same but waiting time as well as size of task team is increasing continuously which will create high load on agent a_1 whereas other contextual agent a_2 and a_3 will be without any task in this scenario.

Therefore a load balancing strategy should be incorporated which will resolve this issue.Therefore if anyhow enrichment factor can be reduced after allocation of task which proportionally reduce enrichment factor impact on agent as well as contextual agent with different amounts.

Load balancing is mainly determined by waiting time of the task or size or number of task team on agents. If the above example we found that there is more task waiting in queue of agent a_1 for resource r_1.In order to address such scenarios it is required to perform modification in comphrhensive enrichment factor. If a particular node have high contextual enrichment factor, task will be allocated for particular resource but at the same time its contextual enrichment factor for that particular resource will be reduced. Analysis shows that how enrichment factor helps in selecting agents and how it varies after modification in comphrehensive enrichment factor.

Cosider the tasks $\{t_1,t_2,t_3,t_4\}$ require r_1 to execute and FIFS policy is considered. Enrichment factor for agents are given task enrichment factor for agents $\{a_1,a_2,a_3\}$ are *3.1,2.4,2.2.* Since enrichment factor for resource r_1 is more for agent a_1 so agent a_1 will be initiator agent for task t_1.But due to allocation of task to agent a_1 there will be decrease in enrichment factor for resource r_1 in agent a_1 which will effect enrichment factor for resource r_1 in agent a_2 as well as a_3 because of considering contextual agent if there is decrease in contextual enrichment factor for particular resource with an agent then it will also effect contextual enrichment factor of its contextual agent in some factor for particular resource.Since effect will be more with agent a_1 and less with agent a_2 and a_3.Its because of the distance factor present in comphrehensive enrichment factor which is in denominator.

From this we can state that as the distance will be more contribution of resource from that contextual agent will be less.Therefore when task allocated to a_1 it will decrease enrichment factor of a_1 with more faster than a_2 and a_3.Cosider it becomes 2.6,2.2,2. After allocation of task t_2 it will be 2,2,1.5.After allocation of task t_3 1.7,1.9,1.4.for allocation of task t_4 it wont be alloacted to task agent a1 it will be alloacted to agent a_2. After analysing we found that task allocation is more for towards agent a_1 for task only require resource r_1 for their implementation.But if processing capability of agent is less then it is difficult for the agent to handle such task.

B. Task Allocation

There are different criteriafor task allocations of resource requirement which are like shortest path, Context aware nodes and proximity aware DHT approach

1. On the Basis of Physically Context Shortest Path

Most of the existing mechanisms concentrate on utilizing actual parameter of the network specifically attributed by physical characteristic outputs of the systems. But these utilizations are not contributing much in the performance and efficiency improvements. Majority of physically contextual algorithms considers distance as a parameter to calculate the proximity of nodes with each other. Distance from source to destination node should be minimum. Many more parameter like no. of nodes in between, type of interconnection network, history analysis, and other parameters are also considered for evaluation.

Let there will be different paths from source to destination and having different cost. Suppose *a1,…. a_n* are such paths for source to destination with cost *c1,……,c_n*respectively. Let b is the path of minimum cost means a shortest path or path containing less number of nodes. Then

$$B = \min cost \rightarrow 0\{ a_1,a_2 \ldots a_n\} \qquad (1)$$

In this approach all path from source to destination is searched and its cost is calculated. Path with minimum cost selected as a physically shortest path. Drawback of this approach is that this approach gives a shortest path but without assured efficiency i.e. selected path is shortest but may be having more time requirement, more number of hopes, high latency, and more probability to be in congestion. Hence

relying on such criteria will diffuse path towards quality of service.

2. On the Basis of Contextually Similarity Nodes

Since whenever task is allocated it will not be allocated to a node which does not contain any resource which is required by task. So whenever task is allocated it should be allocated to those nodes which contain that type of resource. In shortest path approach, system will search all nearer nodes for resources which are required by task and if it found those resources with some near node, it will allocate task to that specific node. Since in this approach time for searching such nodes will be more .As it has to search all nodes. It can get those resources in 1st nodes or it has to search till last node. But in contextual similarity approach this won't be happen.

According to this approach path from source to destination will be a path which can be shortest or not but maximum number of nodes among all nodes which is required to make this path from source to destination will be similar in properties such as resource type, capacity, instruction set etc. It can be mapped to real world in such a way that path may be short or it may be long but time used from source to destination for this path will be always less as compare to some other paths.

Let $p1....pn$ are the paths from source $s_i$to destination d_i B is the path from source to destination which contains maximum number of contextual similar nodes. Then

$$B = \max\{No.of\ contextual\ similar\ nodes \rightarrow infinity\}\{p_1,p_{2.....}p_n\} \qquad (2)$$

Here to perform task allocation,it is mandatory to look for all similar nodes and search all similar nodes for **resource requirement [17]**. Since it will take less time as compare to 1st approach because searching time will be less. The reason behind reduced time requirement is subset of nodes required to search which falls under the specific resource requirement category.

Contextual resource Based Allocation:

Earlier task will be allocated to node which is having high number of that resource. Chances of task allocated to node are directly proportional to number of resource required by task present in that node. This approach never considers including all other similar nodes which also contain those resource to include while calculating node enrichment factor. To calculate node enrichment factor, suppose a task T_i require resource R_i and $N_i(T_i)$ is node enrichment factor for task T_i will be,

$$N_t = Count\ (R_i\ present\ in\ node\ N) \qquad (3)$$

III. PROPOSED ALGORITHM:

A. Physical Contextual Resource Allocation

Finding shortest distance $src \rightarrow dest$ path should contain shortest number of nodes. This approach considers agents whose physical location are nearby in physical environment or physically nearby agents in physical systems. Resource held by this agent is known as physical contextual resource.When an IA require resources which are not present with it,it will communicate with physical contextual agents(PCA) to lend those resources and the agents which will lend those resources will be considered as response agent.

In physical enrichment factor, all physical contextual agents are involved for calculating resource predominance of an agent. Resource involvement from each agent will depend on distance factor. As the distance factor increases, involvement of resource from that agent will be less. Therefore resource present with agent itself will contribute more and resource present with its physical contextual agent will be depending upon distance factor.

This approach is used to model the resources in topology forms which are shown in fig. 2.Therefore task t which requires resources $\{r_1,r_3\}$ will be allocated to agent a_{22} and its physical gradation is shown in fig.2.Since if it requires other resources which are not present with agent a_{22} it will start communicating with physical contextual agent for required resources according to physical gradation .It will communicate with agents present in gradation 1 and so on till it gets all resources. When initiator agent will interact with agents at n^{th} round, it will consider as nth Physical negotiation gradation. Physical gradation for each agent will be different because there will be different physically contextual agents of each agent.

Algorithm 1: Physically Contextual Resource Negotiation of Agent a:

1) *Set the Tag (all_agents in A) = 0;*
2) *Queuecreate(Q);*
3) Queueinsert (Q, a);
4) Tag (a) = 1;
5) resourceFlag = 0;
6) A_t *= {a};*
7)$R_{lackresource} = R_t - R_a$*;*
8) If $R_{lackresource}$ *== {} then resource_flag = 1;*
9) While (!Queueempty(Q) and (resource_flag == 0)) do:
9.1) aOut = Queueout(Q);
9.2) $R_{remaining} = R_{lackresource} - R_{aOut}$*;*
9.3) If $R_{remaining} \neq R_{lackresource}$*, then:*
9.3.1) $R_{lackresource} = R_{lackresource} - R_{aOut}$*;*
9.3.2) $A_t = A_t\ U\ \{aOut\}$*;*
9.4) If $R_{lackresource}$ *== {}, then resource_flag = 1;*
9.5) For $\forall$ *alocal* $\in L_{aOut}$ *:*
If the Tag (alocal) = 0, then:
9.5.1) Queueinsert(Q, alocal);
9.5.2) Set the Tag (alocal) = 1;
10) If (resource_flag == 1), then Return (A_t*)*
else Return (False);
11) End.

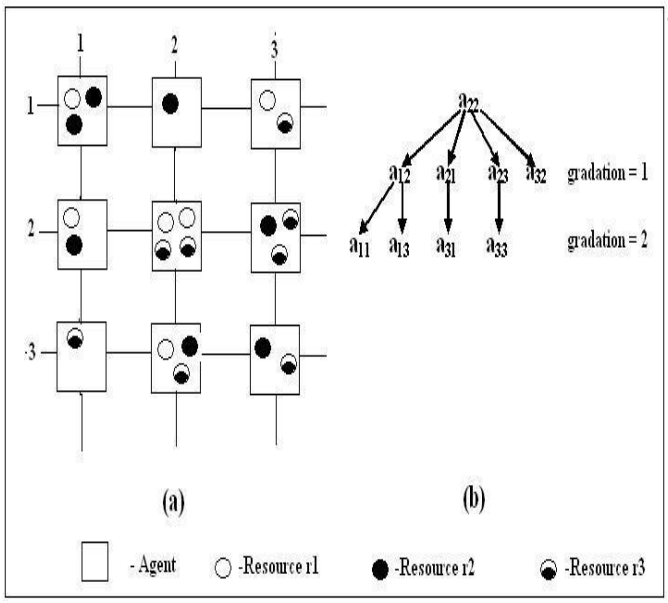


Fig.2 Physical Contextual Resource Allocation

Let *a* be initiator agent and *A* is the physical context of *a*. Consider tag function which is used to assign particular value *0* or *1* to agents present in *A*. Tag value *1* implies that an agent has provided the resource to task for its execution. Consider a task t_i which requires resource R_t for its implementation. Task t_i is allocated to agent who owns resourcesR_a.

Therefore,

$$R_{lackresource} = R_t - R_a \qquad (4)$$

Where$R_{lackresource}$ is as set of lacking resources.If $R_{lackresource}$ is empty it will set resource flag to *1* implying thatrequirement of task t_i completed. Otherwise it will start negotiating with other agent. Agent *a* will start negotiating with other agents present in PCR topology for lacking resources. Consider an agent a_{local} which is physical neighbor or present in PCR topology in next gradation to agent *a,*owns resource R_{alocal} which is required by task t_i to implement. So Agent a_{local} will lend its resources to agent *a* for task implementation.

$$R^t_{alocal} \rightarrow a = \{ r \mid r \epsilon R_{alocal} \} \qquad (5)$$

After lending of resource, the reduced lacking resource will be,

$$R_{remainingResource} = R_{lackResource} - R^t_{alocal} \rightarrow a \qquad (6)$$

The initiator agent will continue negotiating with agent present in PCR topology until task requirement for resource is completed and keeping a set of allocated agents to task $t_i$inA_t. All existing approaches consider static parameters likedistance, no. of hopes to decide the physical proximity of the node with each other. In dynamic environments, weighted assessment of these attributes is very small and contributes nothing to the variable change in the environment. Hence it is
required to consider attributes which can adhere to the change frequency of the environment ,load balancing aware dynamic communication among multiple resources in a timesharing fashion environment is proposed. Each agent servesresources which can dynamically share by the task. To facilitate our proposed load-balancing mechanism, it is required to measure the communication load imposed.

Let a resource be represented by $t_0,t_1,t_2....t_{p-1}$and where n_i is the agent node to which t_i is assigned. Without loss of generality, Let $L_{j,p,COM}$ denote the communication load induced by t_j which can be computed with the following formula, where $T_{ij,COM}$ is the same as the one used in (7):

$$T_p = S_p + \sum_{i=1}^{N} \{ MAX^p_{j=1} (T^i_{j,CPU} + T^i_{j,COM} + T^i_{j,DISK}) \} \qquad (7)$$

The first term on the right-hand side of the equation corresponds to the communication load of resource t_j when the resource t_j and resource t_0 are allocated to different nodes. The second term represents a case where all the resources are present on the same node, and therefore, exhibits no communication load. Similarly, the third term on the right-hand side, measures the network traffic in and out of the agent node, which is the summation of the communication, so $L_{j,p,COM}$ is given by,

$$\begin{cases} \sum_{i-1}^{N} T^i_{j,COM} & j \neq 0, n_j \neq n_j \neq n_0 \\ 0 & \text{j}\neq 0 \ , n_j = n_0 \\ \sum_{1<k<p, n_k \neq n_j} , \sum_{i=1}^{N} T^i_{k,COM} & \text{j=0} \end{cases} \qquad (8)$$

Load of the resources that are assigned to agent's other than the IA. Intuitively, the communication load on agent i,L_i, COM, can be calculated as the cumulative load of all the resources currently running on the agent. Thus, L_i, COM can be estimated as follows:

$$L_{i,COM} = \sum_{\forall j:n\, j=i} L_{j,p.COM} \qquad (9)$$

Given a task arriving at a node, the load-balancing scheme attempts to reduce the communication load by forming the gradation of agents with a lower utilization of network resources. Before dispatching the active task to the remote agents, the following two criterions must be satisfied to avoid useless migrations:

Criterion 1: Let nodes i be the home node and, j, k be the candidate remote nodes for task t, the communication load discrepancy between nodes (i, j) and (i, k) for task t is compared. Criterion 1 guarantees that the optimal node with minimum communication cost is selected for task migration. We can formally express this criterion as:

$$R_{talocal} \epsilon (j_{i.....n}) \qquad (10)$$

Gradation Index GI to decide the level in the gradation tree

$$Gradation\ Index\ GI < \sum_{j=0}^{n} Min_{com}(L_j) \qquad (5)$$

Criterion 2: Let nodes i and j be candidate remote agents for task t with IA as k, then if the estimated response time of t onnode j is less than node i. Hence, we have the following inequality, where R_t^i and R_t^j are the estimated response timeof t on nodes i and j, respectively. $R_{t,mig}^{k,j}$, $R_{t,mig}^{k,i}$ is the migration cost for process t.

$$R_t^j + R_{t,mig}^{k,j} < R_t^i + R_t^{k,i} \qquad (6)$$

The migration cost $R_{t,mig}^{k,j}$ is the time interval between theinitiation and the completion of ts migration, which is comprised of a fixed cost for running *t* at the remote node *j* and the task transfer time that largely depends on the task size and the available network bandwidth measured by the resource monitor.

The experimental results show that the communication aware approach can improve the performance by up to *20* to *35* percent, in terms of slowdown and turn-around time, respectively, under workloads with high communication demands. If the workload is memory-intensive or I/O-intensive in nature, this strategy dynamically and adaptively considers the memory or disks as first-class resources, thereby sustaining the same level of performance as the existing memory-aware and I/O-aware schemes

B. Social Contextual Resource Allocation

Finding shortest distance *(src → dest)* but it contain large number of nodes which is similar in characteristics i.e. here we took nodes characteristics as high, low, medium and finding the path which contain maximum number of nodes containing high characteristic.

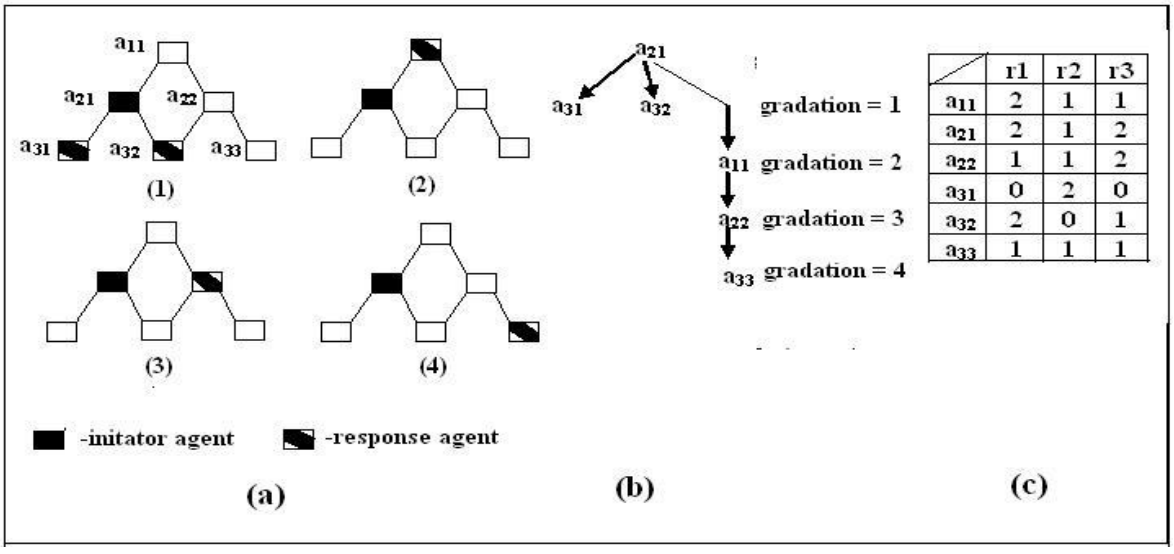
It consider agents which are socially similar and resources in the social context are called socially contextual resource.Since for social context ,multiagents are organised in certain social structure which is used to defined their structure.In Social structure agents which are more near will have more similarities as compared to remote agents.When an intiator agent require resources which are not present with it,it will communicate with social contextual agents to lend those resources and the agents which will lend those resources will be considered as response agent.When an initiator agent will interact with social context agent for resources it will follow SCR-NT topology.

SCR-NT topology contain hierarchical structure which is used by initiator agent to coomunicate with social context agent for required resources.In this herarchical structure agents can communicate with its immediate supervisor,sibling and subordinate in following order:

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.

An agent is more resource predominant if there are more contextual resources. For execution of task,it has resource requirement which need to be satisfied.If task require one kind of resource, it is easy to make task allocation with contextual enrichment factor.But if multiple resource are required to implemnt task then there are different ploicy has to be adopted for initator agentFRFS is a first required resource first satisfy.In this policy agent whose comphrehensive enrichment factor will be more for first resource required by task will be selected as initiator agent.Suppose *{a1,a2,……,an}* E agents in the system and R E *{r1,2r3,r4}* is a set of resources which is required by task to execute.So the agents which will be having high enrichment factor for r_1 will receive chance of becoming an initiator agent,since r_1 is the first required resource by task.

MIFS is a most important resource First satisfy.In this policy,agent whose comphrehensive enrichment factor will be more for most important resource required by task will be considered as an initiator agent.Suppose *{a1,a2,……,an}* E agents in the system and R E *{r1,2r3,r4}* is a set of resources which is required by task to execute and r_3 is an important resource for task *t.* So the agents which will be having high enrichment factor for resource r_3 will be considered an initiator agent,since r_3 is the most important resource required by task t.



	r1	r2	r3
a_{11}	2	1	1
a_{21}	2	1	2
a_{22}	1	1	2
a_{31}	0	2	0
a_{32}	2	0	1
a_{33}	1	1	1

The environment that is considered is with mobile host serviced with virtual server assignment.For heterogeneous environment ,migration of task from one agent to other is a complex task since active tasks required to bypass the different execution boundries when the agents are allocated on different architectureal environments. The main aim of using virtual server is to move the active tasks from one agent to other without any interuptions. In the proposed strategy by Yichuan et.al,Contextual agents are grouped together by their degree of closure for resources held required for execution of particular task. The point which is still unaddressed by most of the papers is the distance parameter between all contextually aware nodes that should participate in task execution.

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 agents, the load transfers may no longer be neglected. As a result, when any agent fall out of resources its load status to be declared and load migration decisions should be made accordingly. Existing approaches considers migrating the load to suitable contextual node without considering the distance between them. Load migration cost is worse affected by the distance. As the agents are heterogeneous the load on each agent may change continuously. This will affect the accuracy, and hence the performance, of the load balancing algorithms.

Our proposed approach forms a closure set for the agents who are contextually linked with each other for a task. Closure set is formed by relocating the nodes within the distance limits set by closure set. Virtual server allows agents to migrate the active tasks. These active tasks can be efficiently migrated from one agent to other within the closure only. Once closure is calculated, nodes are ranked with the order of usage so that numbers of repetitive migrations are reduced and also dynamic membership of closure allows agent member to leave from the group, ones its service to the task gets completed.

Due to this facility, blocking time of tasks waiting for the agent outside the closure gets reduced to some extent. Also the available set of agents for task admission increases significantly. Task admission grant is issued when the agents with enough number of resources are available. To increase this probability, agent should get relieved from closure set as soon as its need is over. To inculcate this requirement, indexed utilization of resource is proposed.

Algorithm 2: Socially contextual resource negotiation of agent a in a hierarchical social structure.

1) Set the Tags (all_agents in A) = 0;
2) Resource_flag = 0;
3) $A_t = \{a\}$*;*
4) $R_{lackresource} = R_t - R_a$*;*
5) If $R_{lackresource} == \{\}$*, then resource_flag = 1;*
6) If (resource_flag == 0), then:
6.1) Negotiation (a, a);
6.2) tempAgent = a;
6.3) While ((resource_flag == 0) and (ptempAgent <> Nil) do:
6.3.1) tempAgent = ptempAgent;
6.3.2) Negotiation (a, tempAgent);
7) If (resource_flag == 1), then Return (A_t*)*
else Return (False);
8) End.

Algorithm 3: Negotiation (a, tempAgent).

1) *Queuecreate (Q);*
2) QueueInsert (Q, tempAgent);
3) Set Tag (tempAgent) = 1;
4) While (! Queueempty (Q) and (resource_flag == 0)) do:
4.1) aOut = Queueout (Q);
4.2) $R_{remaining} = R_{lackresource} - R_{aOut}$*;*
4.3) If $R_{remaining}$ *!=*$R_{lackresource}$*, then:*
4.3.1) $R_{lackresource} = R_{lackresource} - R_{aOut}$*;*
4.3.2) $A_t = A_t \cup \{aOut\}$*;*
4.4) If $R_{lackresource} == \{\}$*, then resource_flag = 1;*
4.5) For $\forall$ *achild* $\in$ *child (aOut):*
If Tag (achild) = 0:
4.5.1) Queueinsert (Q, achild);
4.5.2) Set Tag (achild) = 1;
5) Return (resource_flag);
6) End.

Explanation:

Let a be initiator agent and set of agents in social context of a is A. Consider tag function which is used to assign particular value 0 or 1 to agents present in A. Tag value 1 implies that an agent has provided the resource to task for its execution.

Consider a task t_i which requires resource R_t for its implementation. Task t_i is allocated to agent who owns resources R_a . Therefore $R_{lackResource} = R_t - R_a$ where $R_{lackResource}$ is as set of lacking resources.If $R_{lackResource}$ is empty it will set resource flag to *1* implying that task t_i requirement has completed. Otherwise it will start negotiating with other agent present in SCR topology. Agent a will start negotiating with other agents present in SCR topology for lacking resources in following order as 1) subordinate agents of a 2) parents of agent a and 3) sibling of agent a.

For this it will follow a strategy such that agent *y* is initiator agent then T_y will be subtree whose root will be agent *y* and P_y will be its parent. After considering this, it will start negotiating with agents in the stated order. Consider an agent *TempAgent* which is parent of agent a in SCR topology and owns resource R_{aLocal} which is required by task t_i to implement. So agent *TempAgent* will lend its resources to agent a for task implementation.

$$R^t_{TempAgent} \rightarrow a = \{ r | r \in TempAgent \}$$

After lending of resource, the reduced lacking resource will be

$$R_{remainingResource} = R_{lackResource} - R^t_{TempAgent} \rightarrow a$$

The initiator agent will continue negotiating with agent present in SCR topology until task requirement for resource is completed and keeping a set of allocated agents to task t_i in At.

C. Improved DHT Aware Resource Allocation

IN THIRD ALGORITHM:

Contributions:

1) Relying on a self-organized, fully distributed k-ary tree structure constructed on top of a DHT, load balance is achieved by aligning those two skews in load distribution and node capacity inherent in P2P systems—that is, have higher capacity nodes carry more loads.

2) Proximity information is used to guide virtual server reassignments such that virtual servers are reassigned and transferred between physically close heavily loaded nodes and lightly loaded nodes, thereby minimizing the load movement cost and allowing load balancing to perform efficiently.

3) Their simulations show that our proximity-aware load balancing scheme reduces the load movement cost by *60-65* percent for all the combinations of two representative network topologies, two node capacity profiles, and two load distributions of virtual servers. Moreover, we achieve virtual server reassignments in *O (logN)*time. Load balancing scheme consist of four phases:

1) Load information aggregation
2) Node classification
3) Virtual server assignment
4) Virtual server transfer

1. Load balancing aggregation: Load capacity information of whole system is calculated.
2. Node classification: Under node classification each node is classified as heavy, neutral and light weighted node. Each node is having some capacity. Let a node is having Load as L_i and capacity as C_i . Therefore Ui = (L_i/C_i), If U_i > 1, node is considered as heavily loaded node. If U_i = 1 node is considered as neutral node. If U_i< 1 node is considered as lightly loaded node.
3. Virtual server assignment: The function of virtual server is to reduce load from heavily load to light node. Therefore there is need to find virtual server assignment form heavy node to light node with the help of proximity aware Information.
4. Virtual server transferring: Here virtual server will be transferred from heavily loaded node to light node for load balancing.

The concept of virtual server was proposed to improve load balancing. In this system, a DHT address space for whole system is used and it is divided into different identifier spaces. Each virtual server is responsible for an identifier space and a node can host multiple virtual servers in it. The function of virtual server is to serve data request of object whose id falls into its region. Each virtual server represents some load so whenever any node is heavily loaded it just transfer virtual server from heavily loaded node to lightly loaded node.

Hence load balancing can be achieved by moving virtual server from heavily loaded node to lightly loaded node. And movement of virtual server can be performed by join and leave operation supported by DHT. Load balancing scheme followed here does not concentrated on particular type of resources but it has some assumptions that there can be only one bottleneck resource in the system and multi resource load balancing will be their previous work and load on virtual server will stable for some timescale for load balancing algorithm to perform.

Presently if a task has to be allocated to all nodes suppose {1,2,3,4,5} then a message will be prepared and shortest path for each nodes is found and message containing all details will be forwarded to all nodes but when node will receive message it will be able to get details what it requires no other details.

Consider a message has to be sent to all nodes {1,2,3,4,5} then a message will be prepared containing all message identifier and a shortest path among all nodes will be created whose information will be with super node (which usually contain all information about all node) .After creation of path which will contain all destination node, a single message will be transferred on the path and at each destination as it received the message it will send acknowledgement to source node that message is received to it. Similarly, the same message will be travelled along the path and provide particular details to particular destination. After receiving the message, particular destination node will acknowledge.

For task migration, at each state, calculation of proximity analysis in a node, at the end socially contextual node and physically closeness is calculated. In this case, the path calculation containing set of contextual node is identified and hence the calculation of context at each step is alleviated. At present, major concern is to what factor required defining the contextual proximity. Here it is set of processes required for execution.

Existing methodology analyzes the set of processes at each step from non-local system and contextually closure is calculated. But our algorithm makes this history analysis at once in the initial state. Here original node {1, 2, 3, 4, 5} as per capacity factor of the contextual node. Due to this scenario the complexity and processing of original node is decreased.

Procedure Improved_Dht ( )

Begin
Consider task t and network N
Network_formation(t, N)
Resource_flag = 0
Assign task t to initiator node i present in the logical group say j=1 having high % of resource requirement matching after network rearrangement.
$node_t$ = {i}
$R_{lackresource} = Resource_t - Resource_i$
if($R_{lackresource}$==*{}) then resource_flag = 1;*
*j=1 /*group having group Index = 1 having high resource matching and decreases as* $j \rightarrow k | k \in$ *no of logical network formed*/*
*while(eachLogicalGroup in network is Considered || resource_flag == 1)/*consider k different logical group forms after network rearrangement so j<= k */*
new Node a = searchwithinGroup(j, $R_{lackresource}$*);/* it will return node which will match next resource requirement after searching for particular resource in logical group index j */*
$R_{lackresource} = R_{lackresource} - Resource_a$

$node_t = node_t$ *U {*
if($R_{lackresource}$*=={}) then resource_flag = 1*
j++;
End while
if(resource_flag == 1)
return true;
else
return false;

Procedure network_formation (task t, Network N)

1) *Begin*
2) Let p [1…k] be resource required for previous task ≅ resourceRequire (t)
3) Rearrange Network (N) on the basis of principal index p[1..k]./ socially context node will be closer to each other */*
4) If (IsNetworkRearrange == true)
4.1) reconstruct Dht;
5) End

Explanation:

Consider a task t which requires resource R_t for its implementation. Let p[1…k] be principal resource for particular task t.This algorithm will consider history of resources utilized by task t to decide its principle resources. Since according to that principle resource, it will rearrange the logical network and reconstruct the DHT for logical network. After rearrangement of logical network there will be different $group_{[1.....k]}$ will be formed $\{group_{[1.....k]} \mid group_i > group_j$ in terms of enrichment of principle resource where i <j<=k} Let i be initiator agent in $group_j$ to whom task is allocated.

Task t is allocated to agent i owns resources R_{ij} implies that it present in agent i at group,therefore

$$R_{[lackResources]} = R_t - R_{ij}$$

Where $R_{[lackResources]}$ is as set of lacking resources. If $R_{[lackResources]}$ is empty it will set resource flag to 1 implying that task t requirement has completed. Otherwise it will start negotiating with other agent present in group having group index j. Consider an agent which is present in group having group index j owns resource R_{aj} which is required by task t to implement. So agent a will lend its resources to agent i for task implementation

$$R^t_{aj \to ij} = \{r \mid r \in R_{aj}\}$$

After lending of resource the reduced lacking resource will be

$$R_{[lackResources]} = R_{[lackResources]} - R^t_{aj \to ij}$$

The initiator agent will continue negotiating with agent present in $\{group_j \mid$ j varies from 1 to k} until task requirement for resource is completed and keeping a set of allocated agents to task t inA_t.

IV. EXPERIMENTS:

Validations and Analyses:

In this section, to validate the correctness of our task allocation model,we make a series of case simulations to test the three models. In the simulations, the agent number is 60, the physical distribution and social structure can be set randomlyso that we will compare the performances of the following three models: 1) Physical Context based task allocation model, PCM, 2) Social Context based task allocation model, SCM and 3) Improved DHT based task allocation model, IDM.

In the PCM, the task allocation is implemented according to the physical Context and in SCM task allocation is implemented according to social Context, which is always used in the previous related work. Now, we introduce the IDM model where before task allocation, it rearranges the logical network on the basis of task's resource requirement. Development of logical network for task resource requirement is always useful because it allocates task to agent which hold almost all the resources which is mostly required by task execution which result into less execution time. Since Development of logical network will consume time but with respect to task migration for each resource requirement which is stated in previous models it is negligible.

Comparison for Execution Time With Respect to No. of Resources:

PCM, SCM and IDM models are used to make task allocation; the allocated agents will execute the tasks, the total execution time of tasks with respect to increase in resource requirement of task is tested.The results are shown in Fig. (a)

From the results, we can obtain the following: 1) when PCM model is used, the allocated agents will always incline to be located within the near physical contexts and in SCM model, the allocated agents will always incline to be located within the near social contexts; however, theabove two models do not consider task's resource requirement, they have developed their own structure for task allocation which is rigid and use the different policies to allocate the task on their developed structure.

In PCM execution time can be more if nearby agent does not have resource which is required for task execution whereas in SCM, it develops social structure on the basis of social context which does not changes with task requirement but in IDM for each task requirement a logical network is developed which allocate task to agent which consist most of the required resources and for absent resource it can easily migrate task to near agents which will be capable of complete task requirement.

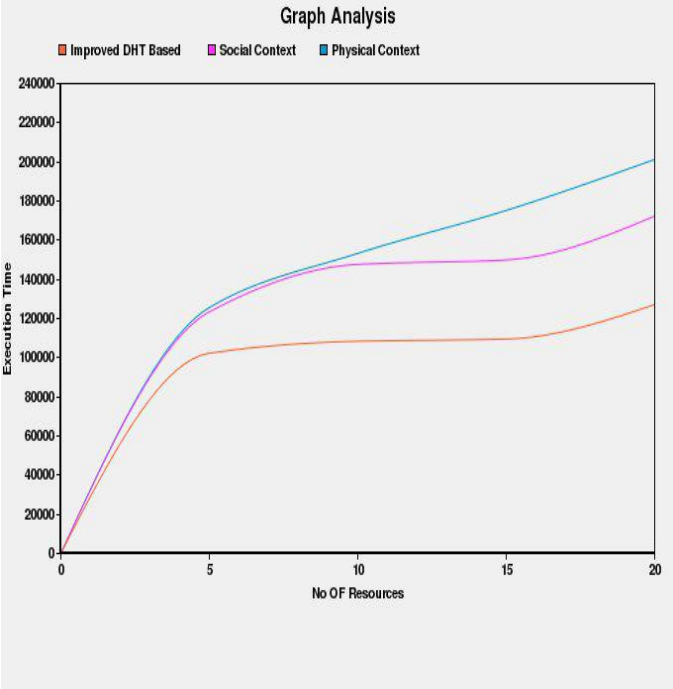


Figure (a)

So the execution time for constant resource requirement will be less in IDM as compare to SCM and PCM.As the no of resource will increase required resource searching will become more complex and lead to more execution time in SCM and PCM while it will be easily handled by IDM and varied its execution time with less factor which will keep its execution time as always less than SCM and PCM.

Comparison for Probability of Task Migration:

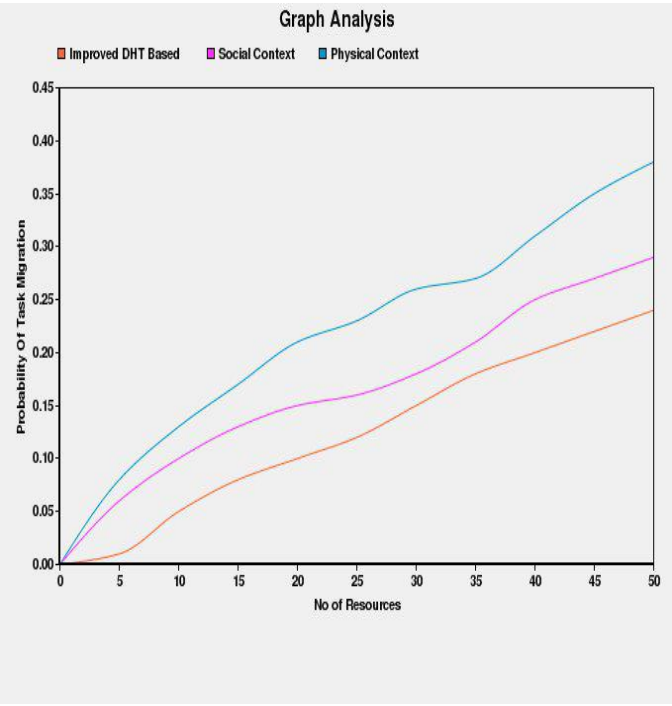


Figure (b)

Probability of task migration refers to the chances that task will not be completely executed at one agent and will be allocated to another agent for completing its requirement. Comparison for PCM, SCM and IDM is shown in fig (b). In PCM and SCM probability of task migration is more as it searches for resources on the basis of physical and social context while IDM allocates task to agents completing its most required resource requirement and developed the logical network such that if task has to be migrated it won't be migrate to far agents.

So probability of task migration will be less in IDM as compare to PCM and SCM. As we analyze the above figure we will observe that for lesser number of resources there is much difference which becomes slightly less as there is increase in number of resources which gives an idea that as resources increases in number ,capability of an agent to hold all the resource required by an task may be decreased. It's because agent will not have that much capability to hold such large number of resource with it.

And even if they are able to have such capability, probability of having all those resources which is required by task may be less which makes slightly change in above graph and leads to decrease the difference which was higher at the time of less number of resources.

Comparison for Probability of Task Allocation:

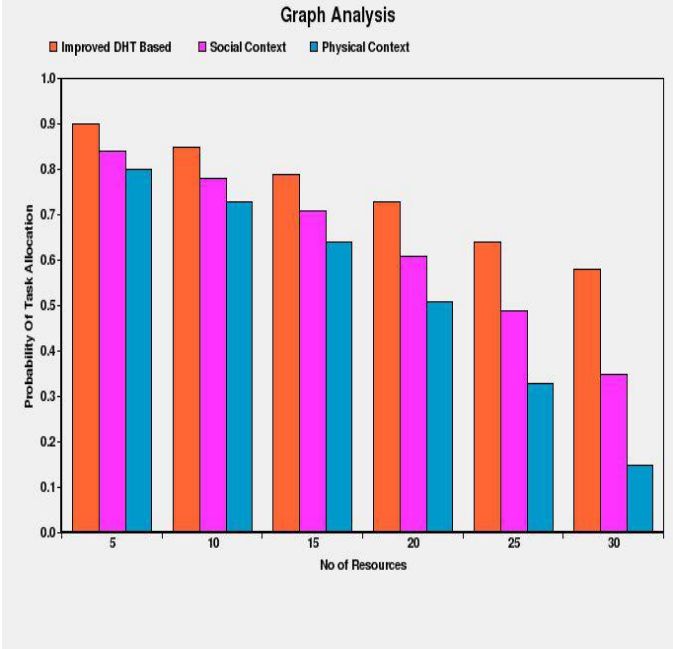


Figure (c)

Probability of task allocation refers to chances that task allocated to agent which can complete most of its requirement. Since if task allocated to agent who can complete its entire requirement then there will not be any need for task migration.

We can state that probability of task allocation is inversely proportional to probability of task migration because as task is allocated to an agent who can complete the requirement of task

then its chances of migrating to another agent would be less. It won't be required resource since its need will be already satisfied.

Comparison for PCM, SCM and IDM for probability of task allocation will compared in fig (c).Since in IDM rearrangement of logical network is done on the basis of principle resource agent (history of resource required by same type of task).So the probability of task allocation will be more as compare to PCM and SCM which uses three policies which is less effective than this. From the above graph we can state that probability of task allocation is decreasing as the resources in number increasing and difference among them is becoming more. IDM is shaping a logical network for the task which will be for completing the needs of the task which is not provided by the SCM and PCM. Due to which it makes such difference as the number of resources increased.

Comparison for No of Requests:

As shown in Fig. (d) , as the task migration will increase there will be increase in number of requests among agents. So it implies that task migration will be proportional to number of request. This can be seen in the result obtained from the simulation for SCM, PCM and IDM where in IDM number of request is less since task migration is less and if task is migrated it migrates to nearby agent which mainly holds principle resources. In PCM it becomes more which shows that if task unable to find nearby agent for resource it has to search all agents such that their physical distance will be less.

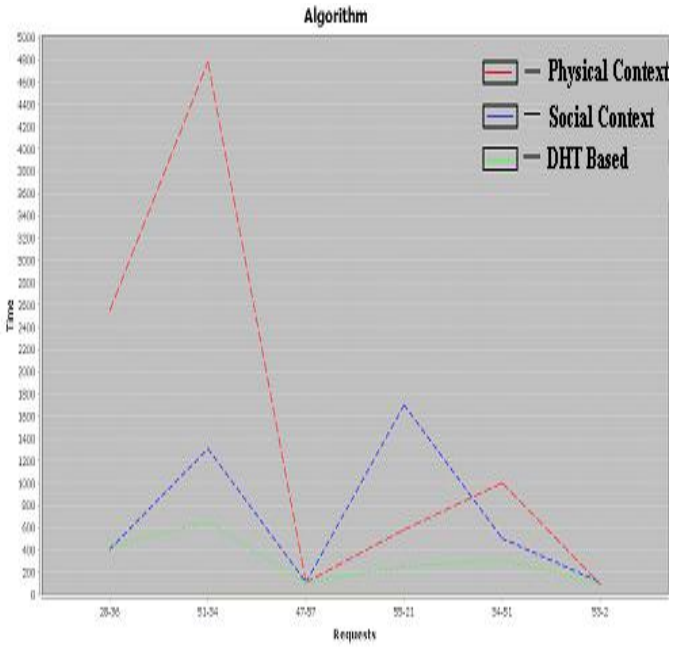


Figure (d)

In SCM if task unable to find resource with contextual agent it will be increase its number of request among agents for searching resources. But in IDM logical network is formed on the basis of principle resource so most of the time task will be able to find resource nearby but if there is change in resource requirement for task form principle resources then task will extend it search to another group. Still no of request among agents for searching resource is less as compare to SCM and PCM.

V. CONCLUSION

Physically and contextually aware decisions with heterogeneous environment outperforms most of the todays approaches under proximity factor and load balancing. Proposed novel scheme for task assignment based on proximity aware load balancing mechanisms reduces load balancing retransmissions by 6 %. A self-organized k-ary tree constructed over DHT allows load distribution such that nodes with higher capacity will carry heavier loads by 11%.

VI. REFERENCE

[1]Y.-C. Jiang and J.C. Jiang, "A Multi-Agent Coordination Model for the Variation of Underlying Network Topology," Expert Systemswith Applications, vol. 29, no. 2, pp. 372-382, 2005.
[2] S. Kraus and T. Plotkin, "Algorithms of Distributed Task Allocation for Cooperative Agents," Theoretical Computer Science, vol. 242, nos. 1-2, pp. 1-27, 2000.
[3]A. Dey and G. Abowd, "Towards a Better Understanding of Context and Context-Awareness," Proc. CHI Workshop What, Who, Where, When and How of Context-Awareness, Apr. 2000.
[4]Yichuan Jiang; Jiuchuan Jiang ,"Contextual Resource Negotiation-Based Task Allocation and Load Balancing in Complex Software Systems" Parallel and Distributed Systems, IEEE Transactions on Volume: 20 , Issue: 5,pp. 641 - 653 2009 .
[5]*Tada, H.* ," Nearest Neighbor Task Allocation for Large-Scale Distributed Systems", 2011 10th International Symposium on Autonomous Decentralized Systems (ISADS), pp: 227 – 232, 2011
[6] Yadav V.K. ,Yadav M.P. ,Yadav D.K.,"Reliable Task Allocation in Heterogeneous Distributed System with Random Node Failure: Load Sharing Approach ",2012 International Conference onComputing Sciences (ICCS),
pp. 187-192, 2012
[8]Wang Qiong; Yang Jie; Zhou Hui; Chen De,"A Heterogeneity-Based Strategy for Resource Distribution in P2P Network", 2011 International Conference on Network Computing and Information Security (NCIS), pp.179 – 182, 2011.
[9]Iijima, K. ; Fukumoto, T. ; Nakano, M. ,"Proposal of Symbiosis ADS concept and negotiation support methods for cooperative resource allocation" 7th International Conference on System of Systems Engineering (SoSE), 2012
pp. 503 – 508,2012.
[10] Xiaofeng Xu ,Guohua Wei,"A Resource Distribution Algorithm Based on Node Factors of Transmission for Ad Hoc Network " , 2012 Second international Conference on Business Computing and Global Informatization (BCGIN), pp. 607 – 610,2012
[11] Yi-Chun Wu ; Chuan-Ming Liu ; Jenq-Haur Wang "Enhancing the Performance of Locating Data in Chord-Based P2P Systems" 14th IEEE International Conference on Parallel and Distributed Systems, pp. 841 – 846, 2008.
[12]Hung Nguyen Chan ; Van, K.N. ; Giang Ngo Hoang ,"Characterizing Chord, Kelips and Tapestry algorithms in P2P streaming applications over wireless network", Second International Conference on Communications and Electronics, pp. 126 – 131, 2008
[13]Yiming Zhang, Lei Chen et.al." Enabling routing control in a DHT", IEEE Journal on Selected Areas in Communications, Volume: 28, Issue: 1, pp.28 – 38, 2010.
[14] Jing Qi, Jiguo Yu , "Scale-free Overlay Structures for Unstructured Peer-to-Peer Networks, "Seventh International Conference on Grid and Cooperative Computing, pp. 369 – 373, 2008.
[15] Chyouhnwa Chen ; Ching-Bang Yao et.al. , "Towards Practical Virtual Server-Based Load Balancing for Distributed Hash Tables", International

Symposium on Parallel and Distributed Processing with Applications, pp. 35 – 42,2008.
[16] Zhang Xingming ; Zhan Shaoxin ,"One Load Balancing Solution for Mobile Video Surveillance System ",International Conference on Computer Science & Service System (CSSS), pp.757 – 760,2012