A Comparative Study of Different Load Balancing Techniques for Heterogeneous Nodes

[1] Pooja Gandodhar , [2]Prof Sudarshan Deshmukh,
Department of Computer Engineering
Pimpri Chinchwad College of Engineering, Akurdi, Pune
{gandodhar.pooja, deshmukh.sudarshan}@gmail.com

***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. Load balancing becomes more challenging as load variation is very large and the load on each server may change dramatically over time, by the time when a server is to make the load migration decision, the collected load status from other servers may no longer be valid. This will affect the accuracy, and hence the performance, of the load balancing algorithms. All the existing methods neglect the heterogeneity of nodes and contextual load balancing. In this paper, context based load balancing and task allocation with network proximity of heterogeneous nodes will be discussed.**

***Index Terms*— distributed system, context based load balancing, 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 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. Effective load balancing algorithms are used to distribute the processes/load of a parallel program on multiple hosts to achieve goal(s) such as minimizing execution time, minimizing communication delays, maximizing resource utilization and maximizing throughputs[5].

Distributed systems are characterized by resource multiplicity and system transparency. A variety of widely differing techniques and methodologies for scheduling processes of a distributed system have been proposed. These techniques are broadly classified into three types: task allocation approach, load balancing, load sharing. The main goal of load balancing is to equalize the workload among the nodes by minimizing execution time, minimizing communication delays, maximizing resource utilization and maximizing throughput.

II. Literature Survey

Generally, the performance of Load Balancing in contextual environment depends upon the selection of an agent based on nodes social or physical context [1]. On the other hand, using Distributed hash tables (DHT's) virtual server can be designed to find suitable resources on a node for load assignments and proximity aware load balancing. 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.From the P2P system perspective, "efficiently" is interpreted as striving to ensure fair load distribution among all peer nodes. Many solutions have been proposed to tackle the load balancing issue in DHT-based P2P systems [6]. 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; 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 load on each node may change continuously. This will affect the accuracy, and hence the performance, of 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 [4]. Hence a comparative study of different load balancing algorithms considering context aware is discussed.

II. Different Load Balancing Techniques

A. Contextual Resource Negotiation based Task Allocation and Load Balancing in Complex Software Systems.

In the complex software systems, software agents always need to negotiate with other agents within their physical and social contexts when they execute tasks. Obviously, the capacity of a software agent to execute tasks is determined by not only itself but also its contextual agents; thus, the number of tasks allocated on an agent should be directly proportional to its self-owned resources as well as its contextual agents' resources.When a multiagent complex system wants to execute a task, the first step is to allocate the task to some software an agent, which is called task allocation. The main goal of task allocation is to maximize the overall performance of the system and to fulfill the tasks as soon as

DOI: 03.LSCS.2013.1.540

possible.An idea of task allocation: if an agent does not own plentiful resources by it, but it can obtain enough resources from other interacting agents easily, it may also be allocated tasks; the number of allocated tasks on an agent is directly proportional to not only its own resources but also the resources of its interacting agents.

The context of an agent can be simply regarded as the environment it is situated which includes the physical context and the social context (organizational one). The physical context 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 context are called the physically contextual resources. 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.

Physical context If an agent lacks the necessary resources to implement the allocated task (we call such agent as initiator agent), it may negotiate with its physically contextual agents; if the physically contextual agents have the required resources (we call those 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. The negotiation relations from agent a 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.

Let a_i be the initiator agent, and the resources owned by a_i are R_{ai}; now, task t is allocated to a_i, and the set of requested resources for implementing t is R_t. Therefore, the set of lacking resources of a_i to implement t is R_t is

$$R^t_{ai} = R_t - R_{ai} \quad (1)$$

Now, it is assumed that agent a_j is negotiated by a_i the set of resources owned by a_j is R_{aj}. If a_j has any resources that resources that a_j can lend to a_i for implementing task t is

$$R^t_{aj->ai} = \{ r / r \in R_{ai} \wedge r \in R^t_{ai} \} \quad (2)$$

The set of lacking resources of a_i to implement t will be reduced as

$$R^t_{ai} = R^t_{ai} - R^t_{aj->ai} = R^t_{ai} - R_{aj} \quad (3)$$

The initiator agent a_i will negotiate with the physically contextual agents according to the PCR-NT, until all requested resources are satisfied.

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. 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. 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.Therefore, let there be an agent a, can negotiate with other agents within the hierarchical structure according to the following orders:

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.

Let there be an agent a_i, the set of agents within its physical context is PC_i and the set of agents within its social context is SC_i. Obviously, every agent within PC_i or SC_i will contribute differently to the resource predominance of ai; the contribution of an agent within PC_i or SC_i to agent a_i is determined by the physical or social distance between such agent and ai. Now, we have the concepts of physically contextual enrichment factor and socially contextual enrichment factor.

The physically contextual enrichment factor of agent a_i for resource r_k is

$$A_i(k) = \sum_{a_j \in PC_i} \left(n_j(k) \bullet \frac{1/dp_{ij}}{\sum_{a_j \in PC_i} 1/dp_{ij}} \right) \quad (4)$$

The social contextual enrichment factor of agent a_i for resource r_k is

$$B_i(k) = \sum_{a_j \in SC_i} \left(n_j(k) \bullet \frac{1/ds_{ij}}{\sum_{a_j \in SC_i} 1/dps_{ij}} \right) \quad (5)$$

B. Efficient Proximity Aware Load Balancing For DHT-Based P2P Systems

Many solutions have been proposed to tackle the load balancing issue in DHT-based P2P systems. However, all these solutions either ignore the heterogeneous nature of the system, or reassign loads among nodes without considering proximity relationships, or both. In this section, we will discuss an efficient, proximity-aware load balancing scheme by using the concept of virtual servers[2].

There are two main advantages of a proximity-aware load balancing scheme. First, from the system perspective, a load balancing scheme bearing network proximity in mind can reduce the bandwidth consumption dedicated to load movement. Second, it can avoid transferring loads across high-latency wide area links, thereby enabling fast convergence on load balance and quick response to load imbalance. Like a physical peer node, a virtual server is responsible for a contiguous portion of the DHT's identifier space.

A proposed load balancing scheme consists of four phases:

1. *Load balancing information (LBI) aggregation*. Aggregate load and capacity information in the whole system.
2. *Node classification*. Classify nodes into overloaded (heavy) nodes, under loaded (light) nodes, or neutral nodes according to their loads and capacities.
3. *Virtual server assignment (VSA)* Determine virtual server assignment from heavy nodes to light nodes in order to have heavy nodes becomes light. The VSA process is a critical

DOI: 03.LSCS.2013.1.540

phase because it is in this phase that the proximity information is used to make our load balancing scheme proximity-aware. 4. *Virtual server transferring (VST)* Transfer assigned virtual servers from heavy nodes to light nodes. We allow VSA and VST to partly overlap for fast load balancing.This load balancing scheme guarantees fair load distribution —that is, have nodes carry loads proportional to their capacities.

C. Heterogeneity-Aware Topology and Routing in P2P Networks

We classify nodes into Heterogeneity Classes (HCs) based on their capabilities, with class zero used to denote the least powerful set of nodes. Peers at the same heterogeneity class are assumed to be almost homogeneous in terms of their capability [3]. The key idea is to ensure that two nodes whose HCs vary significantly are not directly connected in the overlay network. So, we do not allow a connection between two nodes i and j if |HC (i) - HC (j)| > 1, where HC(x) denotes the heterogeneity class of node x. One fundamental question that first arises is how to classify nodes into heterogeneity classes.

Classifying nodes into capability classes is vital for the performance of the P2P system for at least the following reasons: 1) Weak nodes are prevented from becoming hot-spots or bottlenecks that throttle the performance of the system, 2) avoiding bottlenecks decreases the average query response time as perceived by an end user, and 3) from the perspective of the P2P system designer, load balanced architectures improve the overall throughput of the system.

Three Classes of Multitier Topologies

Here we define three classes of overlay topologies: Hierarchical Topology, Layered Sparse Topology, and Layered Dense Topology. We characterize an overlay topology by the type of connections maintained by each node in the system.

Let HC (p) denote the heterogeneity class of peer p. Let maxHC denote the highest heterogeneity class. A multitier topology is defined as a weighted, undirected graph G <V, E> where V is the set of nodes and E is the set of edges such that the following

Connectivity Preserving Rule (CPR) and Connection Restriction Rule (CRR) are satisfied:

$$\text{CPR } \forall p \in V, \text{ if } HC(p) < \text{maxHC then } \exists q \in V \text{ such that } ((HC(q) = HC(p) + 1) \wedge ((p, q) \in E)) \quad (6)$$

$$\text{CRR: } \forall (p, q) \in E \; |HC(p) - HC(q)| <= 1 \quad (7)$$

The key idea behind our probabilistic selective routing (psr) algorithm is to promote a focused search through selective broadcast. The selection takes into account the peer heterogeneity and the huge variances in the number of documents shared by different peers. A main distinction of algorithm is that, in a given iteration, when the query is required to be forwarded to, say, n neighbours, conscious effort is put forth to ensure that the query is sent to the best subset of n neighbours rather than any or all of the n neighbours. We use a heterogeneity-aware ranking algorithm to select the best B neighbours from a set of neighbour peers. The ranking algorithm also takes into account the performance of peers in the recent past and dynamically updates the rankings as peers join or leave the system. The probabilistic selective routing algorithm comprises three major components: ranking neighbour peers, processing a query using the ranking information, and maintaining the rankings up-to-date.

a) Ranking Neighbour Peers

The query processing using the selective routing search technique consists of three steps:

1) The job done by the query originator to control the iteration and the breadth parameter,

2) The job done by the peer p that receives a query Q from the query originator or another peer, which involves computing the ranking metric for each candidate neighbor peer of p,

3) The job of p upon receiving the results from its B neighbors for the query Q.

b) Selective Routing

The query processing using the selective routing search technique consists of three steps:

1) The job done by the query originator to control the iteration and the breadth parameter,

2) The job done by the peer p that receives a query Q from the query originator or another peer, which involves computing the ranking metric for each candidate neighbour peer of p,

3) The job of p upon receiving the results from its B neighbours for the query Q.

c) Maintaining Rankings Up-to-Date

A peer, once having ranked its neighbours, gets stuck with the same ranking order simply because it does not explore the rest of its neighbours to see if they can return better results.

Comparison of all above discussed in table I listing the features of techniques with reference to the mentioned attributes.

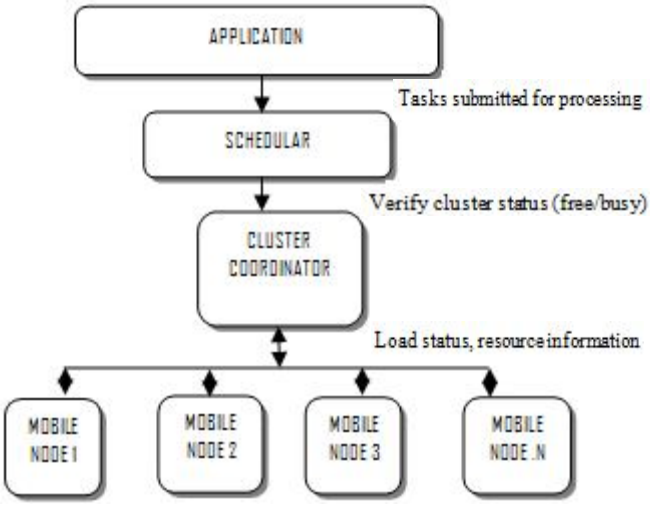

Fig. 1. Project Scope

DOI: 03.LSCS.2013.1.540

TABLE I. COMPARISON OF LOAD BALANCING TECHNIQUES

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	Agent based	DHT based	DHT based
Technique	Agent negotiation based on number of resources owned by it.	Creation of virtual server using k-ary tree structure.	Heterogeneity class aware topology and routing.
Technique	Agent 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 agent 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.	

D. Proposed Work

An improved and efficient load balancing scheme in cluster networks considering contextual information of nodes in heterogeneous environment using the concept of load movement from in the form of virtual machine.Generally, the performance of LB in contextual environment depends upon the selection of an agent 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. The Scope diagram of the proposed work is as Figure 1.

CONCLUSION AND FUTURE SCOPE

In this paper we studied different load balancing techniques of distributed systems.We learned that 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 agents.

We studied an efficient, proximity-aware load balancing scheme to tackle the issue of load balancing in DHT-based P2P systems. The first goal of our load balancing scheme is to align those two skews in load distribution and node capacity inherent in P2P systems to ensure fair load distribution among nodes—that is, have nodes carry loads proportional to their capacities. The second goal is to use the proximity information to guide load reassignment and transferring, thereby minimizing the cost of load balancing and making load balancing fast and efficient.

The next focus of my study will be efficient load balancing and task allocation in P2P networks considering contextual information of heterogeneous nodes.

REFERENCES

[1] "Contextual Resource Negotiation based Task Allocation and Load balancing in complex software systems" Yichuan Jiang and Jiuchuan Jiang, Senior Member, IEEE IEEE Transactions Parallel And Distributed Systems, Vol. 20, No. 5,March 2009.

[2] "Efficient Proximity Aware Load Balancing For DHT-Based P2p Systems" Yingwa Zhu and Yiming Hu,. Senior Member, IEEE IEEE Transactions On Parallel And Distributed Systems, Vol. 16, No. 4, April 2005

[3] "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

[4] "Efficient Lookup on Unstructured Topologies" Ruggero Morselli, Bobby Bhattacharjee, Michael A. Marsh, and Aravind Srinivasan IEEE Journal On Selected Areas In Communications, Vol. 15, No. 1, January 2007

[5] "Distributed system Concepts and Design" By George Colouris, Iean dollimore an tim Kinderberg Fourth edition, addison Wesley, 2010.

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

DOI: 03.LSCS.2013.1.540