A Propound Method For Agent Based Dynamic Load Balancing Algorithm For Heterogeneous P2P Systems

Rupali Bhardwaj
Research Scholar, AIMACT
Banastali Vidyapeeth
Rajasthan, India
rupalibhardwaj09@gmail.com

V.S. Dixit
Dept of CS & IT
ARSD College
Delhi, India
veersaindixit@rediffmail.com

Anil Kr. Upadhyay
Dept of IT
IMR
Ghaziabad, India
anilupadhyay2005@gmail.com

***Abstract*- In peer to peer (P2P) systems agent based load balancing is one of the most important problem. P2P systems are characterized by decentralization, scalability and dynamicity, such that they can been seen as instances of complex adaptive systems (CAS). In this paper we present ant-based load balancing algorithm, which effectively balances loads of peers distributed among P2P systems with the help of autonomous agents called Ants. Ants search a pair of overloaded and underloaded nodes through wandering on network and transfer tasks from different overloaded nodes to different underloaded nodes simultaneously. It is assumed that time break that ants spend on searching a pair of overloaded, underloaded node and transfer of virtual servers between them is negligible. The algorithm developed increases response time of submitted jobs and decreases communication overhead by load transfer in terms of virtual servers between overloaded and underloaded nodes simultaneously.**

Keywords- agent, underloaded node, overloaded node , Ant

I. INTRODUCTION

A P2P system in which every participating node acts both as a client and as a sever (servent) and share a part of their own hardware resources such as processing power, storage capacity or network bandwidth. These resources accessible by other peers directly without passing intermediate entities. The participants of such a network are thus resource providers as well as resource requestors. Peer to Peer systems have a no of advantages over client server system such as scalability, fault tolerance, dynamicity, and performance. Scientists/ Researchers conducted a large amount of research in some challenging areas such as security, Reliability, Flexibility, Load balancing, Searching etc. A p2p system will have a number of peers (nodes) working independently with each other. Some of them are connected by link while some are not. Each node has an initial load, in terms of virtual server, and has a different processing capacity. To minimize the time needed to perform all tasks, load has to be uniformly distributed over all resources according to their capacity [3]. A load balancing algorithm attempts to improve the response time of user's submitted jobs by ensuring maximum utilization of available resources. Generally load balancing algorithms can be classified into static and dynamic categories. In static load balancing algorithm a job is assigned to a fixed resource when it is generated to the system, it is easier to implement and placement of task on a resource is static so that estimation of computation cost can be made in advance but movement of task is not allowed in the system so that system remained in an imbalanced state. In dynamic load balancing algorithm jobs allocated/ reallocated to resources at run time so that imbalanced state of system can be resolved by redistributing jobs in real time. A load balancing algorithm attempts to balance the load by transferring the virtual servers from heavily loaded nodes to lightly loaded nodes in an attempt to ensure better system throughput in terms of response time. A load balancing algorithm is designed by taking into consideration the following issues: Load estimation policy, process transfer policy, state information exchange policy, location policy, priority assignment policy and migration limiting policy [4, 5, 14]. Complex Adaptive Systems (CAS) can be a new programming paradigm for P2P applications. In the CAS framework, a system consists of a large number of relatively simple autonomous computing units, or agents. From a P2P perspective, CAS offers several attractive properties, including total lack of centralized control [13, 18, 21]. In this paper we present a load balancing algorithm, which effectively balances the loads distributed among interconnected nests with the help of autonomous agents called Ants. Autonomous agents (Ants) search a pair of overloaded and underloaded nodes through wandering on network and transfer tasks from different overloaded nodes to different underloaded nodes simultaneously. It is assumed that time break that ants spend on searching overloaded, underloaded node and transfer of virtual servers between them is negligible.

The rest of the paper is organized as follows. Section 2 discusses other work that relates to the development of load balancing algorithm. Section 3 introduces some basic

definitions related to paper. Section 4 presents a propound method for agent based dynamic load balancing algorithm for heterogeneous P2P systems. Finally, Section 5 concludes the paper and indicates directions for future work.

II. RELATED WORK

Most structured P2P system [1], [11], [19] provide a DHT abstraction for data storage and retrieval. Under this assumption the number of objects per node varies within a factor of O (log N), where N is the no of nodes in the system. [19] improves this factor by considering a subset of existing nodes instead of a single node when deciding what portion of the ID space to allocate to a new node. [11] was firstly use the concept of virtual server for load balancing by allocating log N virtual servers per node. They assume that node capacities and loads are homogeneous and object Ids are uniformly distributed. [7] accounts for node heterogeneity by allocating to each node some number of virtual servers proportional to node capacity. [6] for load balancing dynamically balance load among peers without using multiple virtual servers by reassigning light nodes to be neighbor of heavy nodes. However they do not fully handle the case of heterogeneous node capacities and while they prove bounds on maximum node utilization and load movement, it is unclear whether their techniques would be efficient in practice. [1] proposed three load balancing scheme one to one. One to many and many to many based on virtual server. [15] combines elements of many to many scheme (for periodic load balancing results in this emergency load balancing of one particularly overloaded node) simulation results in this have shown that this approach can achieve a good load balance in dynamic environment. However it reduces load balancing problem to a centralized problem, which may cause a single point of failure problem and make the directory nodes more vulnerable. On the other hand, virtual serves may be transferred between two nodes with large link latency because it does not account for proximity relationships of nodes when transferring virtual servers. [22] achieves load balance by constructing a k-ary tree on top of structured overlay. Load balancing information is aggregated and disseminated along the k-ary tree. The nodes of the k-ary tree are responsible for scheduling the reassignment of virtual servers. [12] have proposed the use of the " power of two choices" paradigm to achieve better load balance. Each object is hashed to d>=2 different Ids, and is placed in the least loaded node w of the nodes responsible for those Ids. The other nodes are given a redirection pointer to W so that searching is not slowed significantly. [23] did not use the concept of virtual server, proposed an fully distributed mechanism to maintain the history of file access information, used to predict the future file access frequencies and support the load distribution and redistribution operations. They designs a novel load balancing algorithm which takes the file access history and peer heterogeneity properties into account to determine the load distribution. [17] presented two new location policies that by adapting to system load, capture the advantages of sender initiated, receiver initiated and symmetrically initiated algorithms. [10] constructed simulation for perfect balancing algorithm in which each node had immediate and free access to the states of all other nodes in the system. The reason why this perfect load balancing failed is that even if a node is able to locate least loaded node in the system there is no guarantee that other nodes in the system are not acting on the same information and sending their jobs to the least loaded node. In [20] an important part of a distributed system design is the choice of a load sharing algorithm. A performance metric called the Q-factor (quality of load sharing) is defined which summarizes both overall efficiency and fairness of an algorithm and allows algorithms to be ranked by performance. [8] given a possibly suboptimal assignment of jobs to processors relocate a set of the jobs so as to decrease the makespan. The goal is to achieve best possible makespan under the constraint that no more than k jobs are relocated.

This paper [13] presented a load balancing framework which effectively balances the workloads of the jobs distributed among interconnected nests with the help of information carrying autonomous agents called Ants. The ants helps to effectively balance the loads as it wanders via the interconnection network to find a pair of under loaded and over loaded nests (collection of nodes)[9]. In [16] the given work presents the "Module migration algorithm" that performs load balancing by sort the node them in decreasing order of their load imbalance (i.e. the deviation of a node's load from its reference load (w-ref)). This gives a better pairing and a more optimized solution and by pairing nodes which is having maximum and minimum values of load imbalance. Module migration initiate within the pair of the nodes only.

III. DEFINITIONS

A. Autonomous Agents (Ants) – Ants are generated by nests, autonomously moving across nest network until it fulfill its tasks or with TTL terminated. Ants communicate with each other by modifying their environment (modifying information stored in nests).

B. Nest - Each nest is a peer entity capable of performing computations and hosting resources, has a unique identifier. Any machine connected to the internet and running anthill can act as a nest. In response to a user request one or more ants are generated by nest and assigned them a particular task.

C. Virtual Server - A virtual server looks like a single peer in underlying DHT, responsible for a contiguous region of DHT's identifier space. A physical node can host multiple virtual servers by accompany multiple noncontiguous region of DHT's identifier space. When a node become overloaded, it can transfer one or more virtual servers to a light one. Basic unit of load movement is virtual server. Movement of virtual

servers can be vision as a leave operation followed by a join op, both operations which are supported by all DHT's.

D. Categorization of node – After collecting load information for each node, each node is categorized as either heavy or light or normal loaded node.

For each nest $_i$ of P2P system do
{
If ($L_i < T_{nesti}$)
Type $_i$ = underloaded
Else
{
If ($L_i > T_{nesti}$)
$Type_i$ = overloaded
Else
$Type_i$ = normalloaded
}
}

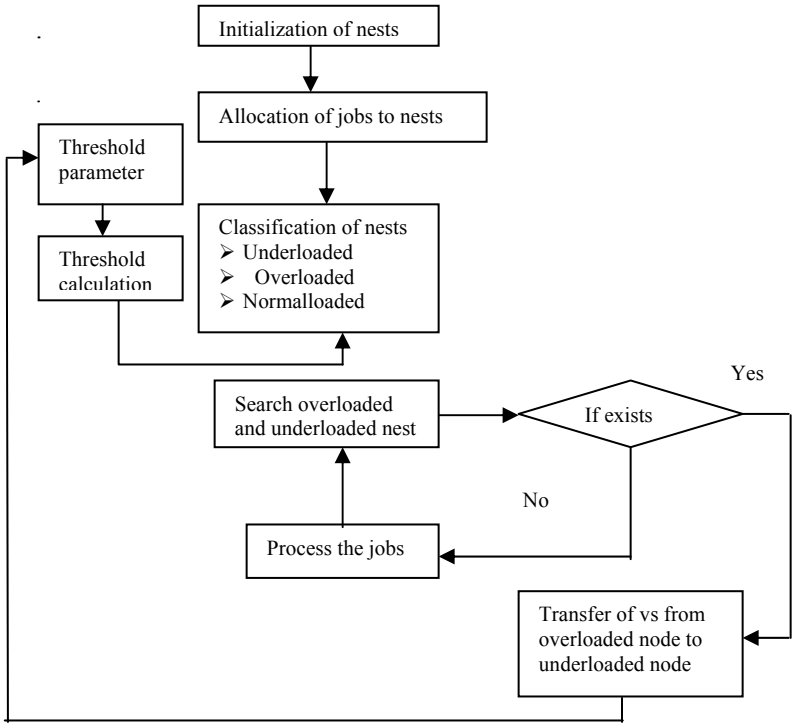
where $T_{nesti} = (L'/C') C_{nesti}$

$L' = \sum_{i=1}^{n} (L_i/n)$

$C' = \sum_{i=1}^{n} (C_i/n)$

IV. ALGORITHM

The load balancer performs load balancing of nodes based on the amount of load present in it. The nodes can be classified in to three categories e.g. overloaded, normalloaded and underloaded. The load of a nest is based on the number of jobs currently in the job queue of that nest. The control flow of the algorithm is shown in Fig 1.



Ants find a pair of overloaded and underloaded nest through wandering on network. Each pair has one overloaded and one underloaded node, former one is called source and later one is called destination. The ant based load balancing algorithm is described below-

```
Procedure loadbalancer ()
{
   Sort nodes according to their loads in ascending order such that most underloaded and most overloaded nest is on top and bottom of list
For each pair of (nd,ns) do
{
removed_vs=choose_best_vs_to_remove(ns)
Transfer removed_vs from ns to nd
}
}
Procedure choose_best_vs_to_remove (nest ns)
  {
  removed_vs = nil
  For each vs of nest ns do
  {
     If (( ns's load – vs's load) < Tns)
        {
          If ((removed_vs = nil)|| (vs's load < removed_vs's load))
                              removed_vs = vs
  }
  if(removed_vs = nil)
    Set removed_vs as the virtual server which has the heaviest load
   }
    return removed_vs
}
```

V. FUTURE WORK AND CONCLUSION

In future we implement agent based load balancing algorithm using java and compare it with other algorithms in terms of response time and communication overhead. Ants search a pair of overloaded and underloaded nodes through wandering on network and transfer tasks from different overloaded nodes to different underloaded nodes simultaneously. It is assumed that time break that ants spend on searching a pair of overloaded, underloaded node and transfer of virtual servers between them is negligible. The algorithm developed decreases response time of submitted jobs and also decreases communication overhead by load transfer in terms of virtual servers between overloaded and underloaded nodes simultaneously.

REFERENCES

[1] Ananth Rao, Karthik Lakshminarayanan, Sonesh Surana, Richard Karp and Ion Stoica, " *Load Balancing in Structured P2P systems*.", In 2nd International Workshop on Peer to Peer Systems (IPTS) Feb 2003.

[2] Antony Rowstron and Peter Drushel, " *Pastry: Scalable, Distributed*

Object Location and Routing for Large-scale Peer to Peer Systems", in proc. Middleware, 2001.

[3] B. Yagoubi, Y. Slimani: Dynamic Load Balancing Strategy For Grid Computing. Proceedings of WASET, Vol. 13, page no 260-265, 2006.

[4] B. Yagoubi, Y. Slimani: Task Load Balancing Strategy For Grid Computing. Journal of Computer Science 3 (3), page no 186-194,2007.

[5] B. Yagoubi,H. Tayeb Lilia, H. Si Moussa: Load Balancing in Grid Computing. Asian Journal of Information Technology 5 (10): 1095-1103, 2006.

[6] David Karger and Matthias Ruhl, "*New Algorithms for Load Balancing in Peer to Peer Systems*", Tech. Rep. MIT-LCS-TR-911, MIT LCS, July 2003.

[7] Frank Dabek, Frans Kasshoek, David Karger, Robert Morris, Ion Stoica,"*Wide Area Cooperative Storage with CFS",* Proc. ACM SOSP, 2001

[8] G. Aggarwal, R. Motwani and A. Zhu, " *The Load Rebalancing Problem*", in Proc. ACM SPAA,2003.

[9] Hein Meling, Alberto Montresor , O zalp Babaog˘lu : Peer-to-Peer Document Sharing using the Ant Paradigm

[10] Ian R. Philip," *Dynamic Load Balancing in Distributed System*", IEEE 1990, page no 304-307

[11] Ion Stoica, Robert Morris, David Karger, M. Frans Kasshoek and Hari Balakrishnan, " *Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications*," in Proc. ACM SIGCOMM, San Diego, 2001, pp149-160.

[12] J. Byres, J. Considine , M. Mitzenmacher, " *Simple load balancing for distributed hash tables*", Proc. IPTPS, Feb. 2003.

[13] K. Saruladha, G. Santhi :Behavior of Agent Based Dynamic Load Balancing Algorithm for Heterogeneous P2P Systems. International Conference on Computational Intelligence and Multimedia Applications 2007, IEEE, 109-113

[14] Kabalan, K.Y., W.W. Smar and J. Y. Hakimian: Adaptive Load Sharing in Heterogeneous Systems: Policies, Modifications and Simulation. Intl. J. Simulation, Page No. 89-100, 2002.

[15] Karthik Lakshminarayanan, Sonesh Surana, Richard Karp and Ion Stoica, B. Godfrey " *Load Balancing in Dynamic Structured P2P Systems*.", In Proc IEEE INFOCOM Mar 2004.

[16] Neeraj Sharma, Girish Sharma: An Algorithm for Dynamic Load Balancing in Heterogeneous Distributed System using Module Migration Probabilities. IACC 2009, 2483-2486

[17] Niranjan G. Shivaratri, Philip Krueger, " *Two adaptive location policies for global scheduling algorithms*", IEEE 1990, page no 502-5098.

[18] O zalp Babaoglu, Hein Meling, Alberto Montresor: Anthill: A Framework for the Development of Agent-Based Peer-to-Peer Systems

[19] S. Ratnasamy, P. Francis, M. Handley, R. Karp and S. Shenker, " *A scalable Content- Addressable Network*", in Proc. ACM SIGCOMM, 2001

[20] Wang, Y.T., and Morris, R.J.T.," *Load sharing in distributed Systems*" , IEEE Transactions on Computers, 34:204-217,1985

[21] Y.Wang, J. Liu: Dynamics of Agent Based Load Balancing on Grids

[22] Y. Zhu, Y.Hu," *Efficient proximity Aware load balancing for DHT-based P2P systems*", IEEE Transactions on parallel and distributed systems, Vol. 16, No.4, April 2005.

[23] Z. Xu, Laxmi Bhuyan,"*Effective load balancing in P2p Systems*", Proc. CCGRID 06 IEEE.