International Journal of Communication Networks and Security

Volume 2 | Issue 1 Article 8

January 2013

EFFICIENT LOAD BALANCING IN PEER-TO-PEER SYSTEMS USING VIRTUAL SERVERS

SUBY MARIA JACOB
Computer Science Department, Calicut University, SUBYMARIAACOB@gmail.com

EFFICIENT LOAD BALANCING IN PEER-TO-PEER SYSTEMS USING VIRTUAL SERVERS

SUBY MARIA JACOB

Computer Science Department, Calicut University

Abstract- Load balancing is a critical issue for the efficient operation of peer-to- peer networks. With the notion of virtual servers, peers participating in a heterogeneous, structured peer-to-peer (P2P) network may host different numbers of virtual servers, and by migrating virtual servers, peers can balance their loads proportional to their capacities. Peers participating in a Distributed Hash Table (DHT) are often heterogeneous. The existing and decentralized load balance algorithms designed for the heterogeneous, structured P2P networks either explicitly construct auxiliary networks to manipulate global information or implicitly demand the P2P substrates organized in a hierarchical fashion. Without relying on any auxiliary networks and independent of the geometry of the P2P substrates, this paper present ,a novel efficient, proximity-aware load balancing algorithm by using the concept of common virtual servers, that is unique in that each participating peer is based on the partial knowledge of the system to estimate the probability distributions of the capacities of peers and the loads of virtual servers. The movement cost can be reduced by using common virtual server.

Keywords- *decentralized load balance algorithms, DHTs, Load balancing, Peer-to-Peer Systems, Virtual Server*

I. INTRODUCTION

The Peer-To-Peer(P2P) computer network is one in which each computer in the network can act as a client or server for the other computers in the network, allowing shared access to various resources such as files, peripherals, and sensors without the need for a central server. P2P networks are application-level networks built on top of end systems, which provide message routing and delivery. This network tries to address the limitations of client/server architecture. Second generation P2P overlay networks are self-organizing, load balanced, fault-tolerant, and scalable guarantees on numbers of hops to answer a query. Load balancing is a critical issue for the efficient operation of P2P networks. This paper focus on the virtual server(VS) or migration based approach to load balancing. The most common type of structured P2P networks implement a distributed hash table (DHT). A virtual server represents a peer in the DHT; that is, the storage of data items and routing happen at the virtual server level rather than at the physical node level. A physical node hosts one or more virtual servers. Load balancing physical nodes to lightly loaded physical nodes. In this paper, the reallocation of a virtual server from a source peer to a destination peer can be simply done by simulating the leave and join operations offered by a typical DHT. The load value of any virtual server at a particular time is the sum of the loads of the objects (data items) stored in the virtual server at that time. Possible metrics for measuring the load of an object may include the storage size of the object and the mean bandwidth required for serving the object. The maximum load that a peer is willing to hold denote the available disk space, processor speed, and the bandwidth of that peer. This paper solve the load balancing problem in a fully decentralized manner. While pioneer studies presenting load balancing algorithms for distributed systems can be found in the literature, these studies either propose centralized load balancing algorithms or often aim at static, small-scale systems and/or homogeneous environments. In this paper, the reallocation of a virtual server from a source peer to a destination peer can be simply done by simulating the leave and join operations offered by a typical DHT. This technique introduces a concept of common virtual server to which the data are migrated from the super peers to reduce the movement cost of the data. It provides an efficient method compared to the prior studies.

II. RELATED WORK

As peers participating in a DHT are often heterogeneous, the work in [1] introduces the notion of virtual servers 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. The DHT is based on the Chord protocol. The Chord protocol supports just one operation: given a key, it maps the key onto a node. Depending on the application using Chord, that node might be responsible for storing a value associated with the key. Chord uses consistent hashing to assign keys to Chord nodes.

This paper aims to solve the load balancing problem in a fully decentralized manner. The pioneer study in [2] propose a centralized algorithm. As the centralized algorithm may introduce performance bottleneck or single point of failure. While [3], [4] propose the centralized algorithms that rely on some rendezvous nodes to balance the loads of peers in a DHT. The proposal in [5] organized virtual servers in a DHT network into an auxiliary tree-shaped overlay network on top of the DHT. Such tree-shaped overlay is used to compute and disseminate the global aggregate, namely, the average allowing each peer in the system to be based on global knowledge to compute exactly its target load threshold and then to identify whether it is heavy or light. Additionally, the tree overlay facilitates the reallocation of virtual servers. The reallocation is performed in a bottom-up fashion toward the root of the tree. The nodes in the tree experience skew the workload for coordinating the reallocation of virtual servers, thus introducing another load imbalance issue.

Application-layer peer-to-peer (P2P) networks are considered to be the most important development for next-generation Internet infrastructure. For these systems to be effective, load balancing among the peers is critical. Most structured P2P systems rely on ID-space partitioning schemes to solve the load imbalance problem and have been known to result in an imbalance factor of in the zone sizes. Chen et al. makes two contributions [6]. First, propose addressing the virtual-server-based load balancing problem systematically using an optimization-based approach and derive an effective algorithm to rearrange loads among the peers It also explore other important issues vital to the performance in the virtual server framework, such as the effect of the number of directories employed in the system and the performance ramification of user registration strategies.

The work in [7] present a method to reduced the load imbalance factor to a constant. It exploit the heterogeneity of peers, such that the number of objects allocated to a peer is proportional to the peer's capacity. This paper does not aim to balance loads among peers in terms of objects allocated to peers. Based on the concept of virtual servers, the many-to-many framework[8] is used to cope with the load imbalance in a DHT. In the many-to-many framework, light and heavy nodes register their loads with some dedicated nodes, namely, the directories. The directories compute matches between heavy and light nodes and then, respectively, request the heavy and light nodes to transfer and to receive designated virtual servers. The many-to-many framework essentially reduces the load balancing problem to a centralized algorithmic problem. As the entire system heavily depends on the directory nodes, the directory nodes may thus become the performance bottleneck and single point of failure.

The work in [5] also present a fully decentralized method for migrating objects stored in a DHT.[5] organizing a DHT into a two-level hierarchical network, where the higher level of the network consists of local clusters. Objects with excess loads are moved in a local cluster to balance the loads of the peers located in the same cluster. For those objects that cannot be moved in a local cluster, they can be transferred to peers in some foreign clusters. The proposed does not assume any specific geometry of DHTs and is thus applicable to any DHT network.

III.PROPOSED SYSTEM

This paper, present a novel load balancing algorithm to minimize both the load imbalance and the amount of load moved. The unique feature of this proposal is that each participating peer estimates and represents the "system state" as the probability distributions for the capacities of nodes and the loads of virtual servers. The approximated probability distributions not only help estimate the expected load a peer should perceive but also provide hints for each peer in the system to schedule the transfers of virtual servers. The participating peers in our proposal operate independently, and they need not rely on dedicated nodes to pair virtual servers and participating peers, eliminating the performance bottleneck and single point of failure. On the other hand, the decentralized algorithm not only depends on global knowledge but also requires coordination among the participating peers to transfer their virtual servers; this may introduce extra workload to the peers responsible for the coordination. In contrast, each peer in our proposal independently and solely manipulates partial information of the system and then reassigns its virtual servers to other peers based on the approximated system state. This proposal is independent of the geometry of DHTs. Unlike [5] this does not require the global knowledge of the entire system to compute the expected load per unit peer capacity. The load balancing scheme we present in this paper is not restricted to a particular type of resource (e.g., storage, bandwidth, or CPU). Here introduce the concept of common virtual server. The super peers share a virtual server and upload the files into it. Whenever this common virtual server exceeds its capacity, it will transfer the load to the virtual server assigned for that super peer. By introducing the concept of common virtual server it is possible to reduce the optimum movement cost

This paper make the following contributions:

- Propose an algorithm which provide dynamic load balancing in heterogeneous, structured P2P systems.
- Solving the problem of load balancing in a completely decentralized manner.

Algorithm Sketch

In this paper, we assume that the entire hash space provided by a DHT and each virtual server in the DHT has a unique ID selected independently and uniformly at random from the space. Let N be the set of participating peers, and V be the set of virtual servers hosted by the peers in N in the DHT. Denote the set of virtual servers in peer i by Vi. Each peer in our proposal estimates the load, which is denoted by

Ti, that it should perceive, where A is an estimation for the expected load per unit capacity.

If the current total load of i is greater than Ti (i.e., i is overloaded), then i migrates some of its virtual servers to other peers. Otherwise, i is underloaded, which does nothing but waits to receive the migrated virtual servers. For an overloaded peer (e.g., peer i), i picks those virtual servers for migration, such that 1) i becomes underloaded, and 2) the total movement cost, MC, in (3) is minimized due to the reallocation. If i is an underloaded peer, then i may be requested to receive a migrated virtual server, and i accepts such a virtual server if the added load due to the virtual server will not overload itself; otherwise, i rejects such virtual server. Algorithm 1 (REALLOCATION), which given as follows illustrates the idea.

Algorithm 1(REALLOCATION), the challenges of implementing the algorithm are 1) how a peer precisely and timely estimates A and 2) how an overloaded peer seeks the peers to receive its migrated virtual servers for balancing the loads among peers. To deal with these issues, our idea is to represent the capacities of participating peers and the loads of virtual servers as the probability distributions, which are denoted by $Pr(X < x)$ and $Pr(Y < y)$, respectively. Both $Pr(X < x)$ and $Pr(Y < y)$ provide valuable information to help the participating peers estimate A, and the overloaded peers discover the underloaded peers to share their excess loads.

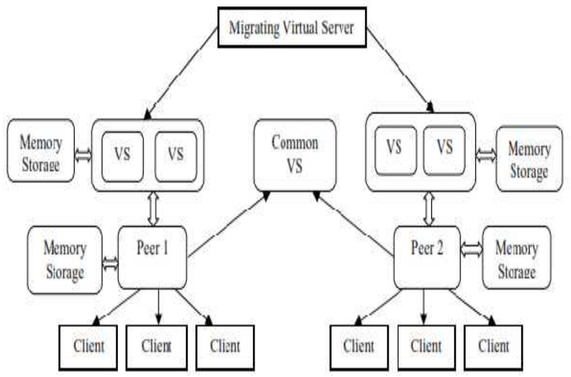

Fig. 1 Architecture diagram Vs-virtual server

IV. SYSTEM DESIGN

The load balancing in the peer to peer networks and by migrating virtual servers the peer can balance the load proportional to their capacity; the load balancing is carried out by 4 modules in this method. They are

- Network Creation.
- Implementation of the Client.
- Super Peer Implementation.
- Establishment of the Virtual Server.
- Migration of the Work Load.

Network Creation

This module designs the windows for the peer page (fig 3.2).These windows are used to send a message from one peer to another. Here we use the Swing package available in Java to design the User Interface. Swing is a widget toolkit for Java. It is part of Sun Microsystems' Java Foundation Classes (JFC) — an API for providing a graphical user interface (GUI) for Java programs. This module mainly focusing the login design page with the Partial knowledge information. Every user should know about their neighbour details so that they can share their information in the network. In this situation we mainly focus the DHT table in the network. Distribute Hash Table maintain a neighbour information so we can get the partial information about the clients. Like Client name, Ip Address, Port No etc. With this information client can communicate with other clients in the network and share their data.

Implementation of the Client.

This module develops a client home page design. In this first the client is going to login. While entering into a client home page client should connect with super peer for that we have to provide a Super Peer Ip & Port No of Super Peer which is running already in the network. Super Peer will accept the request from the client and send the acknowledgement to the client. Now client can communicate with super peer.Client can Browse the file from the system and upload the data to the super peer.

Super Peer Implementation

This module develops a super peer home page design. Super peer will get the request from the client and add to the Distributed Hash Table. After adding to the Hash table client can store the data in the super peer. Super peer can request the virtual server from other network if it is overloaded and can migrate some of the virtual server to the other network if it is under loaded. Super peer will have some capacity so it will check the memory first and if memory is available it will get the data from the client and store it in the memory if memory is not available in the Super peer it will check is there any virtual server is there so that it will upload the data to the virtual server.

Establishment of the Virtual Server

This module develops a virtual server in java swing concept. It is possible to run as many virtual servers needed in the simulation manner. The virtual server will handle the request and response to the super peer according to its capacity. First give a common virtual server to the super peers.Super peer can have N no of virtual server as its need. Virtual server will have some capacity. If super peer capacity is over it will store the data into common virtual server. If common virtual server capacity is no more then it will upload data in to other virtual servers. When those virtual servers also exceed its capacity it will response to the super peer to request another super peer to migrate some of it's the virtual server.

Migration of the Work Load
The load balancing in the peer network is accomplished in this module . It manages the work load in the network when the excess load occurs in the network and the node is heavily-loaded the loads are transferred between virtual servers. If the peer node has no sufficient virtual server for balancing the load then the peer request for the other super peer in the network and ask for the virtual server and then balance its load.

V. CONCLUSIONS

This technique presenting a novel load balancing algorithm for DHTs with virtual servers. The proposal is unique in that it represents the system state with probability distributions. Unlike prior solutions that often rely on global knowledge of the system, each peer in the proposal independently estimates the probability distributions for the capacities of participating peers and the loads of virtual servers based on partial knowledge of the system. With the approximated probability distributions, each peer identifies whether it is under-loaded and then reallocates its loads if it is overloaded. The concept of common virtual server helps to reduce the movement cost. This technique completely overcomes the limitations of directory approach. This method is an efficient load balancing technique.

REFERENCES

[1] I. Stoica, R. Morris, D. Liben-Nowell, D.R. Karger, M.F. Kaashoek, F. Dabek, and H. Balakrishnan, "Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications," IEEE/ACM Trans. Networking, vol. 11, no. 1, pp. 17-21. Feb. 2003

[2] L.M. Ni and K. Hwang, "Optimal Load Balancing in a Multiple Processor System with Many Job Classes," IEEE Trans. Software Eng., vol. 11, no. 5, pp. 491-496, May 1985.

[3] S. Surana, B. Godfrey, K. Lakshminarayanan, R. Karp, and I. Stoica, "Load Balancing in Dynamic Structured P2P Systems," Performance Evaluation, vol. 63, no. 6, pp. 217-240, Mar. 2006

[4] .Chen and K.-C. Tsai, "The Server Reassignment Problem for Load Balancing in Structured P2P Systems," IEEE Trans. Parallel and Distributed Systems, vol. 12, no. 2, pp. 234-246, Feb.2008.

[5] Y. Zhu and Y. Hu, "Efficient, Proximity-Aware Load Balancing for DHT-Based P2P Systems," IEEE Trans. Parallel and Distributed Systems, vol. 16, no. 4, pp. 349-361, Apr. 2005.

[6] Y.Zhu, "Load Balancing in Structured P2P Networks," Handbook of Peer-to-Peer Networking, Springer, July 2009.

[7] D. Karger and M. Ruhl, "Simple Efficient Load Balancing Algorithms for Peer-to-Peer Systems," Proc. 16th ACM Symp. Parallel Algorithms and Architectures (SPAA '04), pp. 36-43, June 2004.

[8] A. Rao, K. Lakshminarayanan, S. Surana, R. Karp, and I. Stoica, "Load Balancing in Structured P2P Systems," Proc. Second Int'l Workshop Peer-toPeer Systems (IPTPS '02), pp. 68-79, Feb. 2003

◈◈◈