INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY
WINGS TO YOUR THOUGHTS.....

DISTRIBUTED HASH TABLE IN PEER-TO-PEER (P2P) SYSTEM

Dhiraj M. Bochare[1], Dr. A. S. Alvi[2]

M.E.(I.T.) Scholar Departmet of I.T.
PRMIT&R, Badnera, Amravati, India
[1]dhiraj.bochare14@gmail.com,

HOD, Department of I.T.
PRMIT&R, Badnera, Amravati, India
[2]abrar_alvi@gmail.com

Abstract: *Peer-to-peer system can share the computing resources and services by communicating directly within a widely distributed network. A fundamental with peer-to-peer applications is the efficient location of the node that stores a desired data item. Peer-to-peer technology has become very popular for various applications such as file sharing or live streaming. Today, a large fraction of the Internet traffic is due to peer-to-peer applications. In this paper, we focus on P2P systems where data items are spread across distributed peer computers nodes and the location of each item is determined in a decentralized manner using a distributed hash table (DHT), such as Chord. Peer-peer systems present an interesting and inherently different resource allocation problem than traditional client/server models. Distributed Hash Tables are scalable, robust, and self-organizing peer-to-peer systems.*

Keywords: *P2P system, DHT, Chord*

1. INTRODUCTION

The peer-peer approach differs from the usual client/server approach towards building networked applications in several vital ways. A peer is both a producer and a consumer of the implemented service. Peer-peer networking has newly emerge as a new pattern for building distributed networked applications [1]. Many peer-to-peer file systems have been planned by diverse research groups during the last few years. Peer-to-peer system (P2P) is formed by a large number of nodes that can join and leave the system at any time and have equal capabilities without any centralized control or hierarchical organization. It can share the computing resources and services by directly communicating within a extensively distributed network. Handful P2P system are intended to scale to hundreds of thousands of nodes and to offer read-write access to a large community of users [2][3]. Peer-to-peer (P2P) overlay networks are distributed systems in nature, without any hierarchical organization or centralized control. The term "peer-to-peer" (P2P) refers to a class of systems and applications that utilize distributed resources to perform a function in a decentralized manner [4]. Peers form self-organizing overlay networks that are overlayed on the Internet Protocol (IP) networks, offering a mix of various features as robust wide-area routing architecture, competent search of data items, selection of nearby peers, superfluous storage, permanence, hierarchical naming, trust and endorsement [5]. A distributed hash table is, a hash table which is distributed among a set of cooperating computers, which refer as nodes. It contains key/value pairs, like a hash table, which we refer to as items. The main service provided by a DHT is the lookup operation, which returns the value associated with any given key. Distributed Hash Tables (DHTs) provide the means to map identifiers ids from a common space onto peers in an overlay network [6-7].

2. PEER-TO-PEER SYSTEM

Peer-to-peer file sharing systems are one of the most popular internet applications and have become a major source of Internet traffic. The grave function in P2P system is distributed computing, data sharing, communication and collaboration, platform services. The resources include computing power, data storage and content, network bandwidth, and presence computers, human, and other resources. Decentralization may apply to algorithms, data, and meta-data. This does not prevent retaining centralization in some parts of the systems and applications. These systems can proficiently situate the node that stores the desired data in a large system efficiently [2][4]. P2P overlay networks have become very popular over the last years due to features that makes them suitable for the enlargement or of new services like overlay multicast communication, large-scale data sharing and at ease distribution P2P networks exhibit three fundamental features: self-organization, symmetric communication and distributed control . In a P2P every participating node acts both as a client and as a server "servent" and "pays" its participation by providing access to some of it resources. Figure 1 illustrates structured system with DHT substrate.

INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY
WINGS TO YOUR THOUGHTS.....

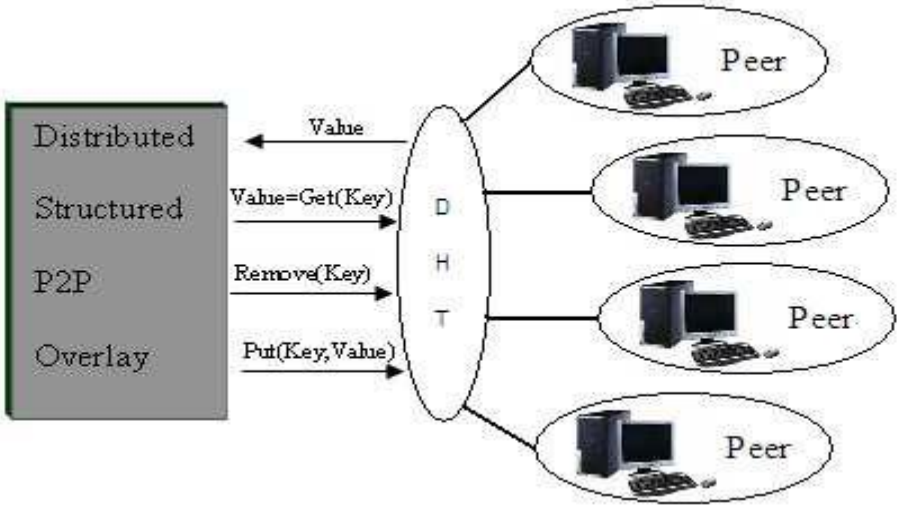

Figure 1: Structured P2P system with DHT substrate

2.1 *Fundamental Features:*

Performance:

The performance depends on total time in data read, insert and delete operations. It contain the locality of data, load balancing, the competence of the locating algorithm, and the efficiency of the routing protocol

Topology:

The organizations of nodes are located in network structure for competent locating and routing. The topology minimizes storage and enhancing scalability of the system.

Naming:

An appropriate addressing scheme can increase performance. The naming is used to stand for shared data objects, network addresses of the nodes, and the arrangement of routing requests across the network.

Locating & Routing:

Competent algorithms minimize overhead of requests queries and increase both scalability and performance. The algorithms used to locate data and rout to the node that stores the preferred data efficiently.

Scalability:

The facility of the system to remain obedient with an increasing number of nodes and data elements. Nodes can join and leave the system freely, and affect the performance slight.

Reliability:

It includes data replication, node failure detection and recovery, and the existence of multiple guarantees for location in turn to avoid a single point of failure. The avoidance of failure within the system and the ease of recovery by reliability [2][3].

2.2 Goals:

Cost reduction:

When the main cost becomes too large, a P2P design can help spread the cost over all the peers. Centralized systems that serve many clients bear the majority of the cost of the system.

Resource aggregation improved performance and interoperability:

Each node in the P2P system brings with it convinced resources such as compute power or storage space. A decentralized approach lends itself naturally to aggregation of resources.

Increased autonomy:

Users of a distributed system are disinclined to rely on any centralized service provider. They desire that all data and work on their behalf be performed locally. P2P systems support this level of autonomy because they require that the local node do work on behalf of its user.

Anonymity:

Autonomy is the notion of anonymity and privacy. A user may not want anyone or any service provider to know about his or her concern in the system. It is difficult to ensure anonymity with a central server because the server will be able to identify the client, at least by internet address. Users can avoid to provide any information about themselves to anyone in P2P system by employing a P2P structure in which activities are performed locally[4][5].

3. DISTRIBUTED HASH TABLE (DHT)

Distributed Hash Tables (DHT) are algorithms used in modern peer-to-peer applications, which provide a reliable, scalable, fault tolerant and efficient way to manage P2P networks in a true peer to peer manner. Hash tables are data structures that can be used to map keys to values, store key/value pairs, and retrieve values using the given keys. They are very useful and efficient in traditional applications. DHT based on P2P systems, files are linked to keys produced by hashing the file name. Each node in the system handles a portion of the hash space and is conscientious for storing a certain range of keys. After a lookup for a certain key, the system will return the identity for example, the IP address of the node storing the object with that key. The DHT functionality allows nodes to put and get files based on their key, and has been proved to be a useful substrate for large distributed systems and a number of projects are

INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY
WINGS TO YOUR THOUGHTS.....

proposing to build Internet-scale facilities layered above DHTs. Figure 2 present Distributed Hash Table [2][8].

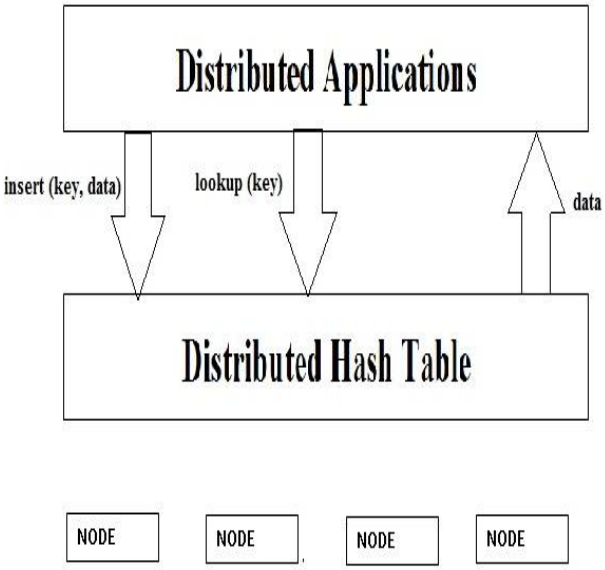

Figure 2: Distributed Hash Table

3.1 Consistant Hashing and Hash Function:

Hash table is a lexicon in which keys are mapped to array positions by a hash function. Hash function can map keys to integers, usually to get an even distribution on a smaller set of values. Hash functions are not reversible. More than one key map to the same position in hash table is called a collision. Perfect hash function is a collision-free hash function that maps each different key to a distinct integer. A hash table that uses a perfect hash has no collisions. Minimal perfect hash function maps each different key to a distinct integer and has the same number of possible integers as keys, which means that *n* keys will map to 0...:*n*-1 with no collisions [2].

The consistent hash function assigns each node and key an m-bit identifier using a base hash function such as SHA-1. A node's identifier is chosen by hashing the node's IP address, while a key identifier is produced by hashing the key. We will use the term "key" to refer to both the original key and its image under the hash function, as the meaning will be clear from context. Also, the term "node" will refer to both the node and its identifier under the hash function. The identifier length m must be large enough to make the prospect of two nodes or keys hashing to the same identifier insignificant. Consistent hashing assigns keys to nodes as follows. Identifiers are ordered in an identifier circle modulo 2^m. Key k is assigned to the first node whose identifier is equal to or follows the identifier of k in the identifier space. This node is called the successor node of key k denoted by successor (k). If identifiers are represented as a circle of numbers from 0 to 2^m-1, then successor(k) is the first node clockwise from k. Figure 3 shows an identifier circle with m=3. The circle has three nodes: 0, 1, and 3. The successor of identifier 1 is node 1, so key 1 would be located at node 1. Similarly, key 2 would be located at node 3, and key 6 at node 0. In this example, key 1 is located at node 1, key 2 at node 3, and key 6 at node 0 [2][9].

4. DISTRIBUTED HASH TABLE ALGORITHM (CHORD)

A distributed hash table (DHT) is a class of a decentralized distributed system that provides a lookup service like a hash table; (key, value) pairs are stored in a DHT, and any participating node can competently salvage the value associated with a given key. Chord is a DHT algorithm that is based on key/value pairs. Topologically Chord resembles ring. In the network overlay, the id of the P2P network client represents the key and the corresponding value is the data that is stored at the corresponding nodes. In Chord, each node stores only a subset of all the keys, which increases the reliability of the network. Chord can be used to implement a number of different services, including distributed information lookup services. Chord is a protocol and algorithm for a peer-to-peer distributed hash table. A distributed hash table stores key-value pairs by assigning keys to different computers known as "nodes". A node will store the values for all the keys for which it is responsible. Chord specifies how keys are assigned to nodes, and how a node can discover the value for a given key by first locating the node responsible for that key. Chord has been designed with six goals as

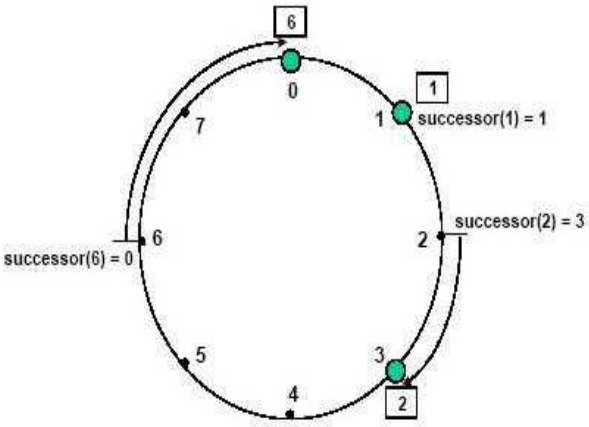

Figure 3: An identifier circle consisting of the three nodes 0, 1, and 3

1. Scalability means that the system can still function efficiently when the size of the network grows.
2. Availability means that the system can function even when the network partitions or some nodes fail.
3. Load-balanced operation means that the key/value pairs are evenly distributed in the system.
4. Dynamism means that the system can handle speedy changes in network topology.
5. Updatability means that data can be updated using the DHT algorithm.
6. Locating according to "proximity" means that if search value is stored in a node near the query originating node, then the data should be fetched from there, instead of over

great distances [10]. The Figure 4 presents a Chord network that consists of 3 nodes that contain the values 6,1,2. The size of key space is three bits, leading to node IDs ranging from 0 to 7.

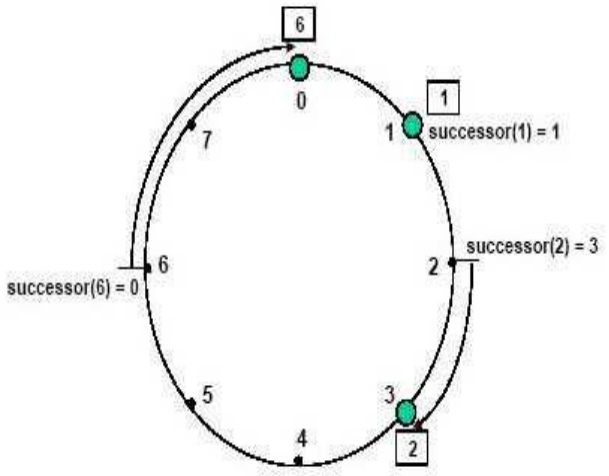

Figure 4: Chord has a ring like routing geometry

4.1 Chord Features:

Load balance:

Chord acts as a distributed hash function, spreading keys evenly over the nodes. It provides a degree of natural load balance by using a uniform hashing function, in which it spreads keys evenly over the nodes of the Chord system.

Decentralization:

Chord is fully distributed and no node is more important than any other. This improves strength and makes Chord appropriate for loosely-organized peer-to-peer applications. Chord is a completely distributed system, meaning no node is more important than any other node, making Chord appropriate for dynamic and loosely-organized P2P applications.

Scalability:

Chord automatically scales well to large numbers of nodes and Chord keeps the cost of a Chord lookup down, since it grows as the log of the number of nodes. The cost of a Chord lookup grows as the log of the number of nodes, so even very large systems are viable. No parameter tuning is required to achieve the scaling

Availability:

Chord provides a high level of availability meaning that Chord nodes can always be found not influence the system changes. Chord provides for availability to newly tied nodes and takes care of node failures. Chord automatically adjusts its internal tables to reflect newly joined nodes as well as node failures. It ensures that, without major failures in the underlying network, the node responsible for a key can always be found. This is true even if the system is in a incessant state of change.

Flexible naming:

Chord provides for flexible naming making no constraints on the structure of names and keys meaning applications can be flexible in mapping names to Chord keys. Chord places no constraints on the structure of the keys it looks up, the Chord key-space is flat that gives applications a large amount of flexibility in how they map their own names to Chord keys [9-10].

5. CONCLUSION

This paper present P2P systems based on distributed hashing table those are the latest P2P system, which can support load balance, competent locating and scalability. We proposed an efficient routing algorithm Chord for DHTs with arbitrarily slanted identifier distributions. Chord, can be implemented to afford efficient, reliable and scalable peer-to-peer overlay networks. These networks can be used to store and salvage material and information in a decentralized manner. Many distributed peer-to-peer applications need to determine the node that stores a data item. The Chord protocol solves this demanding problem in decentralized manner. Chord provides an efficient, distributed naming service for peer-to-peer system. Peer-to-peer distributed hash table (DHT) systems are simple to determine detailed data when their complete identifiers or keys are known in advance.

ACKNOWLEDGMENT

I am honored to work with my Guide, Dr. A.S. Alvi. I will also like to thank him for his guidance and support during this thesis work. I will like to express my gratitude to whole staff for their collaborative effort and technical support. Finally, I will like to thank to my whole family for their invaluable support and motivation during this entire period.

REFERENCES

[1] Zihui Ge, Daniel R. Figueiredo, Sharad Jaiswal, Jim Kurose, Don Towsley, "Modeling Peer-Peer File Sharing Systems," IEEE INFOCOM 2003

[2] Ming Xie Computer Science, University of Ottawa, "P2P Systems Based on Distributed Hash Table," September 26, 2003

[3] J-M. Busca, F. Picconi, P. Sens, "Pastis: a highly-scalable multi-user peer-to-peer File system," INRIA Technical Report 5288. August 2004

[4] Dejan S. Milojicic, Vana Kalogeraki, Rajan Lukose, Kiran Nagaraja, Jim Pruyne, Bruno Richard, Sami Rollins, Zhichen Xu, "Peer-to-Peer Computing," HP Laboratories Palo Alto HPL-2002-57 (R.1) July 3rd , 2003

[5] E. K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim, "A survey and comparison of peer-to-peer overlay networks schemes," IEEE Communications Surveys and Tutorials, vol. 7, no. 2, pp. 72–93, 2nd quarter 2005.

[6] A Dissertation submitted to the Royal Institute of Technology (KTH) in partial fulfillment of the requirements for the degree of Doctor of Philosophy December 2006, "Distributed k-ary System: Algorithms for Distributed Hash Tables,"

[7] Fabius Klemm, Sarunas Girdzijauskas, Jean-Yves Le Boudec, Karl Aber, "On Routing in Distributed Hash Tables," The work presented in this paper was carried

out in the frame-work of the EPFL Center for Global Computing

[8] Timo Tanner, "Distributed Hash Tables in P2P Systems - A literary survey," HUT T-110.551 Seminar on Internetworking 2005-04-26/27

[9] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications," In the Proceedings of the 2001 ACM SIGCOMM Conference.

[10] Matthew Baker, Russ Fink, David Trimm, Adam Whisman, "Implementing a Distributed Peer to Peer File Sharing System using CHEWBACCA – CHord, Enhanced With Basic Algorithm Corrections and Concurrent Activation," In Partial Fulfillment of the Requirements of CMSC 621, Advanced Operating Systems Fall, 2003.