Hypercube Peer-to-Peer for Resource Location on WiMax

Rhia TROGO
Software Technology Department
College of Computer Studies
De La Salle University-Manila
2401 Taft Avenue, Manila, Philippines
+63252744611 loc. 302
trogor@dlsu.edu.ph

ABSTRACT

Hypercube on peer-to-peer environment for WiMax is explored in this paper. Hypercube is a concept in graph theory that is an $n$-regular graph where vertex set is the set of bit string of length $n$, such that there is an edge between two vertices if and only if they differ exactly one bit. The advantage of using a hypercube is that the number of hops is limited to $n$, given that there are $2^n$ nodes. In this paper, the possibility of using hypercube to aid in finding resources in WiMax using SIP (Session Initiation Protocol) is considered. This paper shows that the lookup complexity using a hypercube is $O(\log, n)$ given that there are $n$ nodes and $b$ is the base of the hypercube.

Key Words: Peer-to-Peer, Hypercube, WiMax, DHT.

1.INTRODUCTION

WiMAX (Wireless Microwave Access) IEEE 802.16 is a Metropolitan Area Network that supports a maximum range of 50km and supports peak data rates up to 75 Mbps. Frequency range is 10 to 66 GHz for line of sight and 2 to 11 GHz for non-line-of-sight. This tries to address the shortcomings of (Code Division Multiple Access) CDMA-based technologies like radio devices that support wide area coverage but slow data transfer rate and IEEE 802.11 which provides faster data transfer rate but smaller area coverage. [4]

According to a study conducted by In-Stat/MDR, the fixed wireless broadband (FWB) market will grow up to more than $1.2$ billion by the end of 2007. Based from study, packet-based network traffic growth surpasses traditional voice network traffic and is growing 125 percent per year. Voice traffic is increasing at less than 10 percent per year. [1]

WiMax is a technology that allows wireless connection among network components across a great distance. This technology that is designed for the Internet’s packet-based network traffic can be used for voice network. However, for effective communication, it should be able to discover network components fast. To do this, an organized topology should be observed. In wireless devices, the organization of network components is usually ad hoc, making it hard to locate resources. WiMax offers wider coverage that would mean more network components in ad hoc organization.

They are seeing WiMax as an answer to the significant broadband gaps in the US and in other countries. WiMax is said to cost one-fifth of a 3G mobile network making WiMax the more economical solution. Aside from lower total cost of ownership, WiMax also provides higher flexibility in service delivery. It also provides scalability in handling rural and urban subscriber densities.

One of the foreseen applications of WiMax is IP Telephony services. Session Initiation Protocol (SIP) is used for most telephony services to locate destination nodes. SIP is emerged from IETF (Internet Engineering Task Force) as the session management protocol and it was also chosen as the signaling protocol in the 3GPP (Third Generation Partnership Project) framework. SIP relies on external entities such as servers maintained by organization or service providers. The servers manage the users’ registration, location and forwarding of SIP messages to the location where the recipient is reachable. [9]

There are two logical elements that play a key role in the architecture, registrar and proxy servers. Registrars are the SIP entities where SIP users register their contact information once they connect to the SIP network. A SIP user agent communicates to its registrar server, the SIP user name of the user using the device that is called the address of records (AOR) for that user and the addresses where the user is reachable. Usually, contact information is stored in the form of IP addresses or resolvable names. [9]

An association of the AOR and a contact address is called a binding. SIP registrars exploit an abstract service called location service and return the bindings for the SIP entities issuing a binding retrieval request. [9]

Proxy servers are needed because SIP users only know the AOR. Proxy servers are responsible for talking to another proxy server that is responsible for the domain of the recipient node. The proxy server that is responsible for the domain of the recipient node will then contact the registrar to retrieve the bindings of the AOR. [9]

Registrars and proxies may be co-located in one node. Usually, user agents have a preconfigured outbound proxy server where all outgoing requests are sent and through which all the responses to the issued requests, or new requests are received. [9]

Since these are servers, the point of failure of the system will be the proxy server and the registrars. Peer-to-peer can make the system more fault-tolerant by making all the nodes proxy servers and registrars, similar to [8].
Peer-to-peer systems are inherently scalable, robust and fault tolerant due to the lack of a centralized server and the network organizes itself. This causes high latency for locating resources of interest in the P2P overlay network. Internet telephony can be viewed as an application of P2P architecture where the participants form a self-organizing P2P overlay network to locate and communicate other participants. [8]

Peer-to-peer systems have been used for sharing files in systems that employed, Gnutella, Gnutella2 and FastTrack, which was used to implement Kazaa. These algorithms used servants, hubs and supernodes that followed the same concept. The supernode, hub or servant were used to process queries for the other nodes in the peer-to-peer system. Supernodes, hubs and servants were distributed throughout the system. Ordinary nodes may be elected to become a supernode. An architecture that makes use of a supernode still has partial information and still lacks a centralized server. [11]

SIP has been implemented using P2P using the Chord DHT (Distributed Hashing Table) in [8]. Chord provides the lookup complexity of $O(\log n)$. This paper would like to explore the possibility of using Hypercube instead.

Hypercube has been used in the past to harness computing power for parallel computing in the form of hypercube multiprocessors [10]. Researches to find the shortest path in the hypercube were conducted by [7]. Finding the shortest path is necessary for a faster transportation of communication in the network.

An example of a design and implementation of a hypercube multiprocessor is in [10]. The hardware configuration of the hypercube multiprocessor is realized in [10] by using 16 processing elements (PEs) with a single CPU and five parallel interfaces per PE. Four programmable parallel interfaces (PPI) are interconnected with four adjacent PEs in a hypercube manner, and the other PPI is directly connected to the control computer. [10] focused on the communication mechanism between adjacent PEs of the hypercube multiprocessor and communication between the PE and the control computer.

This paper will not be focusing on the physical organization as in [10] and [7], but rather, will focus on the logical organization of processors. This paper will be focusing on the possibility of using Hypercube in locating resources in a P2P environment.

2. SOLUTIONS FOR SIP AND AD HOC NETWORKS

Previous efforts in providing IP Telephony services using SIP and organization in ad hoc networks are discussed in this section.

Skype is a proprietary P2P VoIP client developed by KaZaa. It has three components, the proxy server, registrar and authentication server. Skype makes use of the concept of a super node that is contacted by peer nodes. The peer nodes associate themselves to the super node. The authentication server where the peer nodes are authenticated is centralized, making it a single point of failure.

[8] like Skype[2], makes use of super nodes. The supernodes perform the P2P lookup while SIP still maintains a proxy server to use for SIP lookup.

dSIP [9] is a decentralized SIP with the assumption that there is no network infrastructure in place and the network components are in a mobile ad hoc network (MANET). It provides a P2P solution by allowing each node to be the registrar and the proxy server. It used Chord for the distributed hash table (DHT).

Plug and Play Application Platform for Mobile Networks (PnPAP) [6] uses super peers and peers. The super peers are used to cache peers and to perform bindings. Peers can also cache existing peers for as long as they have not moved.

Peer-to-Peer based Service over Wireless Ad Hoc Networks [3] uses cluster-based routing where several clusters are formed with the ad hoc nodes, each with a cluster head fully aware of all the other member of the respective cluster and is responsible for communication to them.

Each node in the ad hoc network sends a periodic HELLO message to all its neighbors. Each node that receives the HELLO message computes the degree of the number of adjacent neighbors of the node. This degree information is then broadcasted in the subsequent Hello MESSAGES. Adjacency table is used to embed 2-hop neighbor information in the HELLO messages. Hence, each node gets to know about its degree as well as that of its 1-hop and 2-hop neighbors. The degree information a cluster head is chosen. The node with the highest degree in its 1-hop neighborhood and the node that has the highest degree in the 1 hop neighborhood of any of its 1-hop neighbors are chosen.

Cluster Formation acts as the SIP proxy and a registrar server. The 1-hop nodes adjacent to the cluster heads join the cluster identified by the cluster head with the highest node degree. Each cluster head maintains connectivity with its neighboring cluster heads through gateway nodes selected by the following procedure. The HELLO message can detect the cluster heads which are 2 hops away but not those 3 hops away. To detect cluster heads that are 3 hops away, a cluster adjacency table is maintained at each node. Each cluster member gets information about its 2-hop cluster heads from the HELLO messages. Each cluster member creates its own cluster adjacency table for its 2-hop away cluster heads with the intermediate 1-hop neighboring node, sending HELLO message as the gateway node. The cluster adjacency table is then appended to the HELLO message as an extension and sent to all the 1-hop neighbors. This way, any cluster head which are 3 hops away and identifies the cluster adjacency table relaying node as the gateway node.

HyperCup [11] proposes a hypercube graph topology for peer-to-peer systems that organizes nodes so that the $2^n$ nodes are just $n$ nodes away. Due to the dynamic nature of P2P systems, nodes will join and leave arbitrarily and the number of nodes can be less than $2^n$, in a case such as this, the closest node stand in the place of the other nodes.
3. HYPERCUBE
Hypercube is a concept in graph theory that is \( n \)-regular graph where the vertex set is the set of bit string of length \( n \), such that there is an edge between two vertices if and only if they differ exactly one bit. The advantage of using a hypercube is that the number of hops is limited to \( n \) given that there are \( 2^n \) nodes. Assuming that the base of the hypercube is 2. Given this, the look up complexity is \( O(\log n) \).

4. HYPERCUBE ON PEER-TO-PEER

4.1. Supernodes
The supernodes will be used as the registrar and the SIP proxy servers. This is to minimize the signaling because as the number of nodes in a hypercube increase so does the degree of vertices and the number of neighbors of each node.

Each supernode will have its own cache. The cache entries of supernodes will have a specific expiration so that the cache keeps up to date information. Time to Live (TTL) will be employed as well for supernodes that can no longer be reached. This needs to be considered as this system is for a peer-to-peer system where nodes join and leave arbitrarily.

4.1.1. Registrar
The Registrar is where the contact information is sent once the node has connected to the system. The Registrar upon the registration of the node will attach to it a binary address that will be used by the SIP proxy as well.

4.1.2. SIP Proxy
The SIP proxy is the one to return the contact information of the requested node given its AOR. The SIP proxy uses the binary address that is assigned by the registrar. The SIP proxy will perform differently from the usual SIP servers as the supernode will now use the binary number of a node to locate the node in the system.

4.2. Peer nodes
Peer nodes on the other hand will maintain a cache as well but will employ a lazy update. If the node contacts a peer node using the cache and is unable to do so, then information of the peer node is deleted from the cache.

Each node will have its own cache that will contain the details of a caller that will be used for binding. This makes it possible for the peer node to do without a proxy server if it has the entry of what is being requested for by another node. The cache entries of supernodes will have a specific expiration so that the cache keeps up to date information. Time to Live (TTL) will be employed as well for supernodes that can no longer be reached. This needs to be considered as this system is for a peer-to-peer system where nodes join and leave arbitrarily.

4.3. Node Startup
The graph of the proposed system is as follows where \( G \) is a hypercube graph and \( V \) is the set of vertices and \( E \) is the set of edges.

\[
V(G) = \{ p_1, p_2, \ldots, p_s | p \in P \land s \in S \}
\]

where \( P \) is the set of peer nodes and \( S \) is the set of supernodes.

\[
E(G) = \{ p_1, p_2, \ldots, s_t | p \in P \land s \in S \}
\]

Where each \( p \) \( s \) connects \( p \) to an \( s \), each \( p \) \( p \) connects \( p \) to a \( p \) and \( s \) \( s \) connects \( s \) to an \( s \).

During registration, the supernode contacted, will have to make sure that it still can accommodate a new node following the properties of hypercube. If it no longer can, another supernode will be sent the REGISTER message. Since this is a hypercube structure, the node should be found after asking \( \log n \) times, where \( b \) is equal to two in this discussion for simplicity, as in HyperCup [11]. The supernode will confirm that it has received the message and will inform the node that it is registered.

The peer node contacts a bootstrapped supernode initially. The supernode will see if the node can be accommodated otherwise, the REGISTER message is sent to another node that can accommodate the node.

A peer node can send a REFRESH message to the supernode to inform the supernode that it is still up. If it does not, after a certain TTL, the peer node is removed from its cache.

4.4. Node Discovery
A node can request for a node’s information by sending its request to its supernode. The premise here is, as in email, the peer node trying to contact the other node knows the AOR of the node to be contacted. Given the AOR, it will then contact the supernode. The supernode, based on the AOR should be able to find the binary equivalent and from there, it should be able to find the node it is
looking for. Contact information is then sent to the requesting node. The requesting node can then save this information in its cache. The cache will then be used the next time a node needs to be retrieved.

Figure 2. Hypercube with binary number labels [5]

On the other hand, a peer node may act like a SIP proxy as well since it has a copy of the information of its neighboring nodes as well. It also keeps in its cache recently contacted nodes so that it can also be contacted by other peer nodes should there be no super nodes available. The following section discusses how the system can recover from a node or supernode failure.

4.5. Node Failure

There are two types of failures, one where the node leaves properly or announced departure and is able to announce to the other nodes that it will be leaving and another is the unannounced departure from the system which usually happens when an unexpected failure occurs.

4.5.1 Announced Departure

If the node is to perform an announced departure, it needs to send its neighbor list to the closest node. The closest node will then contact the nodes in the neighbor list so that they are informed that the previous node is no longer up.

The same is true, should a supernode perform an unannounced departure. It will give a copy of its registered nodes to the closest supernode. The closest supernode will then contact the supernodes in the neighbor list so that they are informed that the previous supernode is no longer up.

4.5.2 Unannounced Departure

If the node experiences a failure and executes an unannounced departure, the closest node will then take its place in the system. In a case of unannounced departure the node that left will not be able to inform the other nodes, however, in this event, other nodes may still reach the neighbors of the failed node as each node has a \( \log_b n \) number of connections. These connections provide other paths by which other nodes can communicate with them. Through this process, the hypercube can be rebuilt. The neighbor nodes to the failed node can then be used as their temporary connection.

The super nodes exchange REFRESH messages so that the other supernodes are aware that they are still up. If after a period of time no REFRESH message is received, the supernode is assumed to have failed. The nodes that are unable to contact the super node will then perform node startup again but this time contacting other supernodes so that they are redirected to another node. Their previous supernode’s address will be included in the request so that they are assigned to the node closest to their previous supernode.

5. RESULTS

From Section 2, it can be seen that there are two main types of implementations, centralized SIP proxy and registrar or peer-to-peer implementations. Centralized SIP proxy and registrar servers are not desirable for the single point of failure. Peer-to-peer implementations are decentralized and thus, offers fault tolerance. The main drawback however, is the latency caused by locating resources.

Most solutions to SIP as stated in section 2. of this paper use Chord. The fastest lookup is \( O(1) \) which only happens if the proxy server and registrar is centralized next to this is the lookup complexity of \( O(\log n) \).
To make the look up complexity of hypercube to become at par with the complexity of Chord, the base of the hypercube must be changed to 10 so that the lookup complexity now becomes $O(\log n)$ as well. The advantage of a hypercube is load balancing since there can only be $\log b n$ neighbors per node. To strictly follow the definition of hypercube as defined in [5], base 16, 32 or 64 can be used. Moreover, since this solution is peer-to-peer it provides fault tolerance and security.

This shows that hypercube implementations can perform better than other lookup algorithms given that the base is greater than 10.

6. RECOMMENDATIONS

While this is a theoretical proof that hypercubes can perform better than systems using Chord since they offer a lookup complexity of $O(\log n)$ it is best that this is tested on an actual SIP on WiMax environment. Another paper will be written in the future regarding this as there is an ongoing collaboration with a foreign research institute to explore this possibility further. Moreover, further study should be made as to identify the optimal degree of the hypercube for supernodes.

This can be used not only on SIP WiMax but also with other applications in distributed file storage and computing.

7. ACKNOWLEDGEMENTS

Much gratitude is owed to Dr. Nelson Marcos for providing his invaluable contribution to this paper.

8. REFERENCES


