MA-Chord: A new approach for Mobile Ad hoc network with DHT based unicast scheme

Qi Meng
School of Telecommunication Engineering
Beijing University of Posts and Telecommunications
Beijing, China
victorymeng@gmail.com

Hong Ji
School of Telecommunication Engineering
Beijing University of Posts and Telecommunications
Beijing, China
jihong@bupt.edu.cn

***Abstract*— We put DHT (Distributed Hash Table) based P2P (Peer to Peer) application – Chord into MANET (Mobile Ad hoc Network) in this paper. Then, we propose a new routing modified scheme MA-Chord which bases on a unicast scheme with DHT. MA-Chord can be efficiently used to not only provide indirect, key-based overlay routing, but also conventional direct routing. Our simulation implies that MA-Chord can markedly outperform conventional reactive Ad hoc routing (AODV). Therefore, MANET nodes can run MA-Chord for application purpose, especially P2P applications.**

Keywords- DHT; P2P; MANET

I. INTRODUCTION

A mobile Ad hoc network (MANET) [6] consists of a collection of wireless mobile nodes dynamically forming a temporary network without the use of any existing network infrastructure or centralized administration. Researchers have been attracted and motivated by the popularity of file-sharing applications over internet in peer-to-peer (P2P) system, such as Napster [1] and Gnutella [2].

Recently, the convergence of MANETs and P2P networks becomes more and more popular for P2P applications can be easily deployed in MANET. In order to solve the scaling issues when P2P applications used in MANET, some highly structured P2P lookup algorithms have been proposed, such as Chord [3], Tapestry [4] and CAN [5]. A Distributed Hash Table (DHT)[19] substrate shields many difficult issues including fault tolerance, locating objects, scalability, availability, load balancing, and incremental deployment from the distributed application designers.

So far, several studies have been done for P2P system in MANETs [11]. There is a popular approach that proposes the integration of a conventional DHT with an Ad hoc routing protocol to provide indirect routing in MANETs is Ekta [16]. Ekta, unlike MA-Chord, is based on Pastry [18], and it also uses DSR [12] for its route discoveries. Ekta has no notion of overlay clusters of physically close nodes. Thus, the routes traveled during its overlay routing process may be expected to be less efficient than those in the cluster-based MA-Chord.

To the best of our knowledge, none of the existing approaches use a DHT – which might be present in the MANET already to supply indirect routing – to provide unicasting in MANETs. Our MA-Chord protocol deploys IP unicast in DHT-based P2P application in MANETs.

This paper is organized as follows: In section II, we present MA-Chord scheme briefly, include some concepts and main processes. In section III, we do some simulations, and then we analyze them to get the comparison. We draw a conclusion in section 4 and present our future work.

II. THE MA-CHORD SCHEME

A. Network Model

Conventional Ad hoc routing protocols deliver a packet from a source node to a predefined destination node. However, indirect routing differs from direct routing in that packets are no longer routed based on the destination node's address but on a key instead. The packet is then to be delivered to the node that is responsible for the packet's key. In other words, the actual address of the final destination node is usually unknown to the sender. For this purpose, MA-Chord has been proposed.

MA-Chord is a DHT substrate particularly designed for mobile Ad hoc networks. It combines Ad hoc On-Demand Distance Vector (AODV) Ad hoc routing [10, 13] and Chord overlay routing at the network layer to provide an efficient primitive for key-based routing in MANETs. The AODV protocol is a widely used reactive MANET routing protocol [9, 14], which is described by Perkins the protocol in [15].

Each node in a MA-Chord network assigns itself a unique overlay id (for example by hashing its IP address, etc.) which defines its logical position on the virtual overlay id ring. Furthermore, in MA-Chord, a message's packet header contains a message key. MA-Chord then routes the message to that node in the network that is currently responsible for the message key – i.e. to the node whose overlay id is currently the numerically closest to the message key among all MA-Chord nodes in the network. To avoid message broadcasts (e.g. for route discovery) whenever possible, MA-Chord explicitly considers physical locality in the construction of its routing tables. See section B and C in detail.

B. The features in MA-Chord

1) Clusters Standard (Internet-based) DHTs are largely oblivious to the actual physical topology so that two overlay neighbors can be physically located arbitrarily far from each other. This can lead to a large overlay stretch as subsequent overlay hops can literally crisscross the physical network. Due to the volatile nature of physical routes in MANETs, this effect is especially prohibitive in such environments.

To exploit physical locality in its overlay, MA-Chord uses Random Landmarking [17]. Instead of having fixed landmark nodes – which simply are not available in MANETs – fixed landmark keys are used.

2) Routing Tables MA-Chord maintains three different routing tables: an AODV-style routing table for physical routes from a node to specific target nodes, as well as a stripped down Chord routing table for indirect routing.

To avoid the maintenance overhead, the only proactive routing table maintenance that a MA-Chord node performs is the periodic pinging of its "left" (i.e. the node who has the largest overlay id smaller than the node's own) and "right" (i.e. the node who has the smallest overlay id larger than the node's own) leaf as this is necessary to guarantee overlay routing convergence. All other routing entries are gained or updated implicitly by overhearing data packets.

3) Routing When a node wants to send a packet to a specific key, it consults its Chord routing table to determine the closest prefix match, as stipulated by standard Chord. Next, it consults its AODV routing table for the physical route to execute this overlay hop. Intermediate nodes on the physical path of an overlay hop consult their AODV table for the corresponding next physical hop. This process continues until the packet reaches the eventual target node that is responsible for the packet key – i.e. whose overlay id is the numerically closest to the packet key.

C. The main processes of MA-Chord scheme

Conventional Ad hoc routing protocols route a data packet from a source node to a destination node based on the destination's node address. MA-Chord, on the other hand, routes data packets based on an overlay key. Therefore, when a node A wants to send a data packet to a specific node B using MA-Chord, node A obviously needs to know node B's current MA-Chord (i.e. overlay) id.

We are using a very simple and straight-forward address resolution scheme. Whenever a node assigns itself a new overlay id (e.g. by joining the network, by moving to a different MA-Chord cluster), it will publish its new, current overlay id at a certain location in the network. For this purpose, the node hashes its node address (e.g. its MAC or IP address), thereby acquiring an overlay key (OK). Next, the node simply routes a packet containing its current overlay id toward its OK. The node currently responsible for that OK stores the originator's current overlay id and becomes its temporary address server. As is shown in Figure 1.

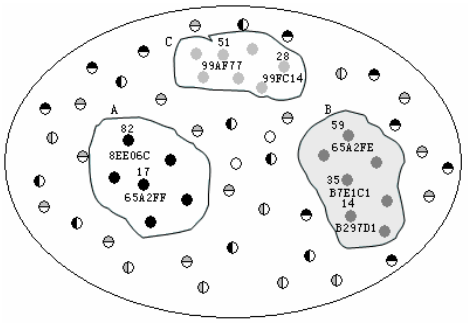

Figure 1. Overview of MA-Chord

1) The process of address publication

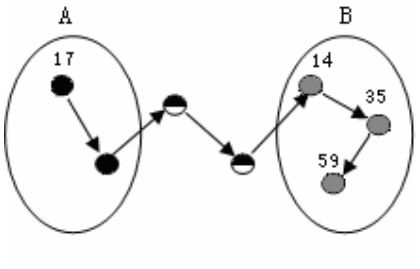

Figure 2. Process of Address Publication

The process of MA-Chord's address publication is shown in Figure 2. We take node 17 for example. This process can be summarized as follows:

Step 1: Node 17 gets the overlay id "65A2FF";

Step 2: Node 17 hashes its node id into the overlay id space and gets OK (Overlay Key) h (17)→"65A2FF";

Step 3: Node 17 sends a packet containing its new overlay id to its OK;

Step 4: MA-Chord delivers the packet to node 4 whose id is "B297D1";

Step 5: Node 4 forwards the packet to node 35 whose id is "B7E1C1";

Step 6: The packet arrives at node 59 whose id is "65A2FE" which is closest to node 17' OK.

After 6 steps, node 59 becomes node 17's address server and stores its current overlay id.

Analogously, when a node A now wants to send a data packet to some node B whose current overlay id is still unknown to node A, it hashes B's node address to obtain node B's OK. Using that OK, node A then sends a request to node B's address server to acquire node B's current overlay id. Once node A has learned about node B's id, it can use that overlay id to send data packets destined for node B using MA-Chord.

2) The process of address resolution

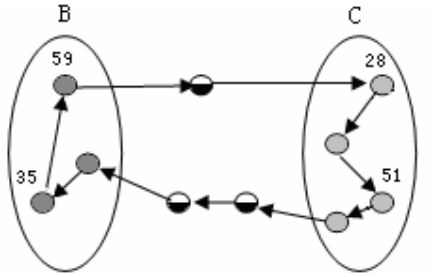

Figure 3. Process of Address Resolution

The process of MA-Chord's address resolution is shown in Figure 3. We suppose that node 51 wants to communicate with node 17 but does not know node 17's current overlay id. Hence, node 51 needs to contact node 17's current address server. Therefore, this process can be summarized as follows:

Step 1: MA-Chord hashes node 17's address to get its OK: h (17)→"65A2FF ";

Step 2: Node 51 sends a request towards key "65A2FF ";

Step 3: The key "65A2FF " routed through node 35 to node 59 which is the current address server for node 17.

After 3 steps above, node 59, then, sends a response containing node 17's current overlay id back towards the requester's overlay id, which, in our example, is routed through node 28 whose id is "99FC14" back to node 51.

Note that, since MA-Chord routes based on an overlay key, the response does not necessarily take the exact reverse path of the request.

3) The process of unicast scheme

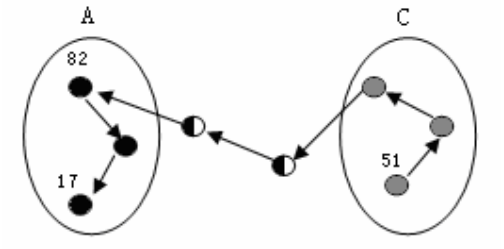

Figure 4. Process of Unicast

After the response from the address server, node 51 can now begin the actual unicast. Using the overlay id provided by node 59, node 51 sends its data packet towards that id. Figure 4 shows how MA-Chord routes the data packet to node 82 (id "8EE06C") during the first overlay hop, and then on to node 17.

III. SIMULATIONS AND ANALYSES

We evaluated MA-Chord's unicast performance using Opnet. MA-Chord's results were compared to the results of both a popular reactive routing protocol – AODV [5] .For all simulations, we used the 802.11 standard with a transmission range of 250m.

Furthermore, we chose a node density of 100 nodes / km2. All nodes were constantly (i.e. 0s pause time) moving around according to the random waypoint model. Each simulation run lasted one simulated hour.

First, we evaluated the success rate that MA-Chord and AODV achieve in networks of varying sizes (150, 200, and 250 nodes). A constant node velocity of 1.4 m/s was used (a quick walking pace). To measure the success rate, every node sends out a request to a random node every 10 seconds.

The target node will then send back a response to the requesting node. The success rate is simply defined as the ratio between the total number of successfully received responses and the total number of sent requests. Note that in the case of MA-Chord this request/response communication can be preceded by an additional address resolution communication if the destination node's current overlay id is unknown.

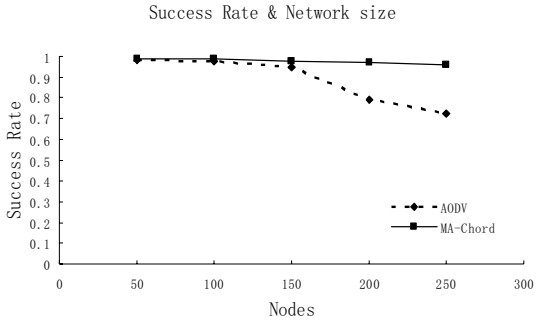

Figure 5. Success Rate & network size

Figure 5 shows the success rates of the respective protocols versus the different network sizes. Only MA-Chord achieves success rates of above 95% for all network sizes. The success rate of AODV drop quickly as the network size increases. The reason for MA-Chord's markedly better success rates is that MA-Chord uses numerous short routes that are likely up-to-date. AODV will try to route a packet on a direct route from the source to the target. As the network size increases, these long routes become ever more volatile and break frequently, which results in their lower success rates.

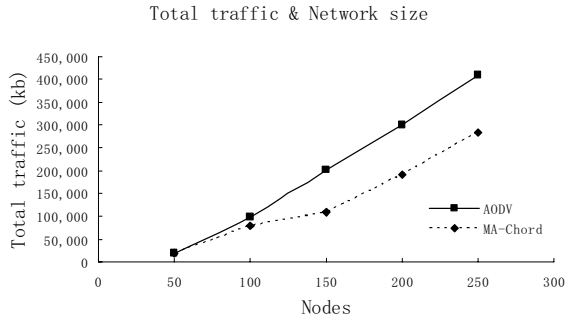

Figure 6. Total traffic & network size

Figure 6 depicts the total amount of traffic exchanged during one simulated hour. These numbers include any

packets generated by the routing agents and the applications – i.e. data packets, routing packets, control packets, etc. One can see that MA-Chord produces far less overall traffic than AODV routing agents do. This is because MA-Chord often uses short and recently updated routes, and, therefore, it rarely has to engage in AODV-style route discoveries. AODV on the other hand frequently needs to discover or repair its long and direct routes. MA-Chord achieves much better success rates compared to the AODV routing protocol.

IV. CONCLUSION

Mobile Ad hoc networks are inherently complex in nature. Our simulation results show that our unicast approach based on MA-Chord generally outperforms AODV routing protocol. This might be especially useful for MANETs that are already running a DHT application in the first place. In this case, nodes would no longer have to maintain a separate Ad hoc routing protocol in addition to their DHT, but instead they could let MA-Chord handle their point-to-point routing as well.

The simulation results presented in this paper provide a first and encouraging look at the performance of DHT-based unicasting in MANETs. To further evaluate DHT-based unicasting, additional simulations will be needed. It will be particularly interesting to study the effects that varying network parameters such as node density, request rates, request distributions, etc. might have on the performance.

In the near future, we will do more research in this area, such as we want to use Pastry or TORA as the MANET routing protocol to do the similar research. And in the further research we will take the node velocity and node density into consideration.

ACKNOWLEDGMENT

This study is supported by the Project of National Natural Science Foundation of China (60672124).

REFERENCES

[1] Napster, http://www.napster.com/.

[2] Gnutella, http://www.gnutella.org.

[3] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications," Proc. ACM SIGCOMM 2001, pp.149-160, 2001.

[4] A. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object Location and routing for large-scale peer-to-peer systems," Proc. 18th IFIP/ACM International Conference on Distributed Systems Platforms (Middleware 2001), 2001.

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

[6] R. Sholmeier, I. Gruber and M. Finkenzeller, "Routing in Mobile Ad Hoc Networks and Peer-to-Peer Networks, a Comparison", Inter. Workshop on Peer-to-Peer Computing, Pisa, Italy, May 2002.

[7] FIPS 180-1. Secure Hash Standard. U.S. Department of Commerce/NIST, National Technical Information Service, Spring_eld, VA, Apr. 1995.

[8] C. E. Perkins and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers," Comp. Commun. Rev., Oct. 1994, pp. 234–44.

[9] C. E. Perkins and E. M. Royer, "Ad-hoc On-Demand Distance Vector Routing," Proc. 2nd IEEE Wksp. Mobile Comp. Sys. and Apps., Feb. 1999, pp. 90–100.

[10] Jakob Eriksson, Michalis Faloutsos and Srikanth Krishnamurthy. "Scalable Ad Hoc Routing: The Case for Dynamic Addressing". In INFOCOM 2004.

[11] Y. Charlie Hu, Saumitra M. Das and Himabindu Pucha. Exploiting the Synergy between Peer-to-Peer and Mobile Ad Hoc Networks. In Proceedings of HotOS-IX: Ninth Workshop on Hot Topics in Operating Systems, Lihue, Kauai, Hawaii, May 18-21, 2003

[12] D. B. Johnson and D. A. Maltz. Dynamic Source Routing in Ad Hoc Wireless Networks Kluwer Academic, 1996.

[13] C. E. Perkins and E. M. Royer, "Ad Hoc On Demand Distance Vector (AODV) Routing," IETF Internet draft, draft-ietf-manet-aodv-02.txt, Nov. 1998.

[14] J. Broch, D. B. Johnson, and D. A. Maltz, "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks," IETF Internet draft, draft-ietfmanet-dsr-01.txt, Dec. 1998.

[15] C. E. Perkins and E. M. Royer, "Ad hoc on-demand distance vector routing," Proc. of 2nd IEEE workshop on Mobile Computing Systems and Applications, pp.90-100, 1999.

[16] H. Pucha, S. M. Das, and Y. C. Hu. "Ekta: An Efficient DHT Substrate for Distributed Applications in Mobile Ad Hoc Networks". In Proc. of IEEE WMCSA. December 2004.

[17] R. Winter, T. Zahn, and J. Schiller. "Random Landmarking in Mobile, Topology-Aware Peer-to-Peer Networks". In Proc. of FTDCS, May 2004.