Optimized Mesh Protocol for P2P Content Sharing

Madhavi Perumbudur [1], Maganti Venkatesh [2], G.Nageswara Rao [3], Maganti Phanindara kumar [4]

[1]Asst.Prof,CSE Dept,Mallareddy College of Engg, Dhulapally, Secunderabad,AP,India

[2] Asst.professor, Sasi Institute of Technology and Engineering, Tadepalligudem,W.G(dt)

[3]Asst.professor, Sasi Institute of Technology and Engineering, Tadepalligudem,W.G(dt)

[4] MIS Lead, Wipro

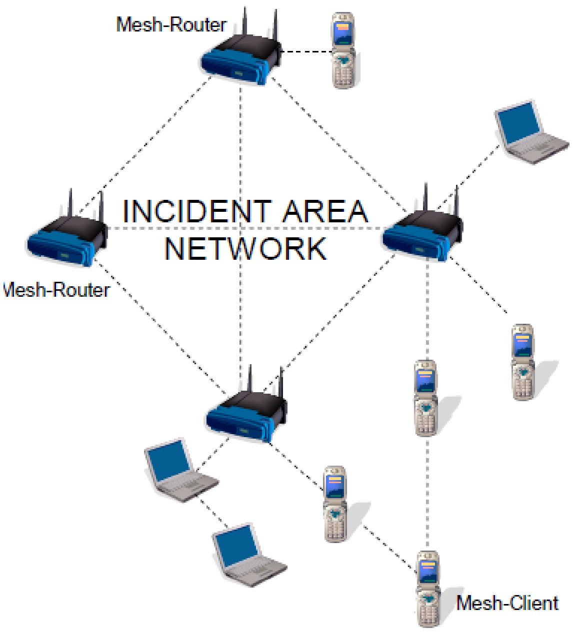
Abstract: Many Possibilities have been explored for building a scalable peer to peer communications using Distributed Hash Table (DHT) based proposals like Chord, CAN, Pastry, and Tapestry etc that provides a simple lookup service during content sharing. Althogh these DHT's exhibit several advantages that fit in a P2P context, a p2p lookup computation overhead is still noticable during initiations. Wireless Mesh Networks (WMN) offer an attractive platform for a wide range of applications driven by its categorical implementations such as Infrastructure, Client, and Hybrid based. A remarkable feature for client based WMN's is its natural architecture to support P2P communications. So in order to reduce the lookup delays we propose to use client based WMN's platform along with an optimized Chord extension. For convinience we term this hybrid implementaion as MESHCHORD protocol. As demonstrated via extensive simulations, our results indicate that the proposed MESHCHORD exhibits resilience to tcp traffic and thus manages to reduce message overhead of as much as 40% with respect to the basic Chord design.

Keywords—Wireless mesh networks, community networks, distributed hash tables, peer-to-peer resource sharing

I INTRODUCTION

Wireless Mesh Networks (WMN) have recently gained considerable popularity owing to their self-configuring, self-optimizing and self-healing capabilities. These networks offer an attractive platform for a wide range of applications, such as public safety and emergency response communications, intelligent transportation systems, and community networks. A WMN consists of two types of wireless nodes: Mesh Routers and Mesh Clients. The Mesh Routers have improved computational, communication and power resources as compared to Mesh Clients. Mesh Routers are generally static and form the multi-hop backhaul network with optional access to other auxiliary networks. In addition, Mesh Routers are also typically equipped with multiple wireless network interfaces and are therefore able to establish high capacity connections. Mesh Clients are mobile devices, which take advantage of the existing communication infrastructure provided by the Mesh Routers.

WMNs can be divided into three main types [1]: Infrastructure, Client, and Hybrid. In an Infrastructure WMN, Mesh Clients gain access to each other or to the backhaul network through Mesh Routers and are not actively involved in the routing and forwarding of packets. Thus, all Mesh Clients gain access to Mesh Routers via a single wireless hop. In Client WMNs, Mesh Clients communicate with each other directly, without involving any Mesh Routers. A Client WMN is essentially a pure multi-hop mobile ad-hoc wireless network [2]. A Hybrid WMN combines the connectivity pattern of both the Infrastructure and Client WMNs. In these networks, both the Mesh Clients and Mesh Routers are actively involved in the routing and forwarding of packets. In addition, the Mesh Clients can also access the wireless backhaul network via multiple client hops. A typical scenario where a Hybrid WMN might be employed is in emergency response and disaster recovery situations, where traditional communications infrastructure might not be available. In such a case, a hybrid WMN can provide a so-called incident area network, as illustrated in Figure 1.



Hybrid/Client WMNs commonly adopt the IEEE 802.11 standard [3] at the Physical and MAC layers. Routing in a multi-hop wireless mobile network is a challenging task. In contrast to the relatively static Mesh Routers, Mesh Clients can be highly mobile, resulting in routes frequently being severed. In addition, as Mesh Clients are relatively resource constrained devices, the routes should preferably be established via Mesh Routers. Consequently, the type of nodes, i.e. Mesh Client versus Mesh Router, should be taken into consideration during the route establishment, lookup process. The routing protocols used for Hybrid WMNs can be broadly categorized into two types: Reactive and Proactive. In reactive routing protocols, the routes are established only when required, generally via flooding of Route Request packets in the network. While, in proactive routing protocols the routes are established before actual usage, through periodical exchanges of connectivity information. Both protocols have their individual advantages. Reactive protocols focus on minimizing control packet overhead while the proactive protocols attempt to minimize the route establishment delays.

In this paper, we investigate the feasibility of the Chord algorithm [4] derivative for peer-to-peer content sharing in wired networks to a wireless mesh network environment. Starting from the basic Chord implementation, we propose a specialization named MESHCHORD – that accounts for peculiar features of mesh networks: namely, i) the availability of a wireless infrastructure, which enables location-aware ID assignment to peers, and ii) the 1-hop broadcast nature of wireless communications, which is exploited through a cross-layering technique that bridges the MAC to the overlay layer. These features tend to reduce the overall look up delays that is dominant in prior p2p sharing communications.

II RELATED WORK

Various Distributed Hash Table (DHT) proposals have been highlighted in the literature to address the problem of realizing delay free distributed peer-to-peer communications. The various DHT approaches proposed in the literature mainly differ on the structure imposed to the virtual overlay and on the mechanism used to route search requests in the overlay. Among them, we cite Chord [4] (which we briefly describe in the next section), CAN [5], Pastry [6], and Viceroy [7]. However, these DHT approaches have been designed and optimized for operation in wired networks, and issues such as limited bandwidth, node mobility, and so on, are still potential threats to wireless mediums.

Recent research[8] have addressed the problem of enabling P2P resource sharing in mobile ad hoc networks (MANETs). Some of them proposed extension/modification of existing P2P approaches to work efficiently on MANETs. Among them, we cite extension/modifications of Gnutella, and of Pastry. Others proposed their own solutions, mostly tailored at efficiently dealing with peer mobility. Among

ISSN-2320-7884 (ONLINE)
ISSN-2321-0257 (PRINT)

them, we cite ORION, Mobiscope, RBB, and the service discovery protocol proposed in [9]. A standard technique used to improve performance of P2P algorithms when used in wireless networks is cross layering, i.e., taking advantage of information delivered from lower layer protocols (typically, the network layer) when constructing the logical links between peers. The idea is to try to enforce locality as much as possible, i.e., peers which are close in the (logical) overlay topology should be as close as possible also in the physical network topology.

Although a careful design of the overlay improves the efficiency of P2P systems for MANETs, the combination of node mobility, lack of infrastructure, and unreliable communication medium has hindered the application of P2P approaches in medium to large size ad hoc networks. As a consequence of this, P2P approaches have been successfully applied to MANETs composed of at most a few tens of nodes, and the problem of designing scalable P2P systems for ad hoc networks remains open.

A more recent trend of research pushes the idea of cross-layering a step forward, basically collapsing the overlay and network layer into a unique, location-aware layer, which implements a sort of geographic hash table. The technique proposed in [10] is targeted towards MANETs, and is based on the idea of mapping the IDs of the objects to share to trajectories, and to let the nodes which are closer to that trajectory manage the corresponding ID. In [11], the authors suggest using a two-tier architecture, where the sensor nodes store the data, and a certain number of proxy nodes implements the distributed indexing mechanism.

III PRELIMINARIES

We assume a two-tier architecture for file/resource sharing: the lower tier of the architecture is composed of (possibly) mobile mesh clients (clients for short), which provide the content to be shared in the P2P system; the upper tier of the architecture is composed of stationary mesh routers (routers for short), which implement a DHT used to locate file/resources within the network. Unless otherwise stated, in the following we use the term peer to refer to a router forming the DHT at the upper tier of the architecture.

We assume routers are stationary, but they can be switched on/off during network lifetime. When a client u wants to find a certain resource, it sends to its responsible

router (a mesh router within its transmission range) a FindKey packet, containing the key (unique ID) of the resource to find (see next section for details on key assignment to router/resources).

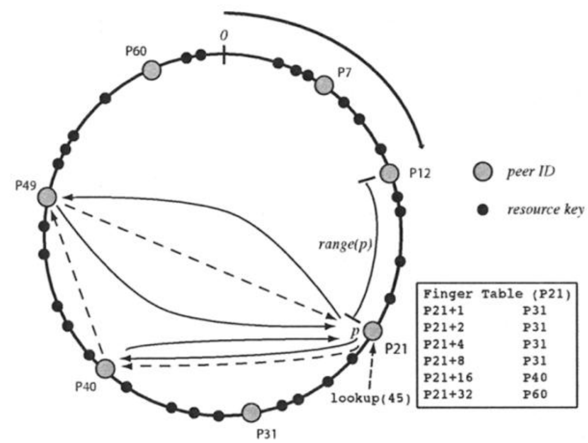


Fig. 3.1-Basic Chord operations. m is set to 6.

The responsible router forwards the resource request in the DHT overlay according to the rules specified by the Chord protocol (see below), until the resource query can be answered. In case of successful query resolution, a packet containing the IP address of the client holding the requested file/resource is returned to client u through its responsible router a. For details on the rules for responsible router selection, on the procedures needed to deal with client mobility, and to add/remove resources from the distributed index.

IV MESHCHORD PROTOCOL

Our protocol variant (MESHCHORD), adds two simple modifications to the basic Chord design and residing WMN protocols: locationaware peer-ID assignment, and MAC-middleware crosslayering. For what concerns the idea is to exploit locality, and to assign peers which are close in the physical network with close-by IDs in the unit ring. This choice is motivated by the observation that, according to Chord specifications, most of the packets are exchanged between a peer and its successor/predecessor in the unit ring. More specifically, location-awareness is implemented by assigning IDs to peers

In the first modification, we use the (Hop Count) – (Mesh Router Count) as the routing metric instead of the standard hop count, to facilitate preferential routing of packets via Mesh Routers. By selecting routes which minimize this metric, we can guarantee that that the established routes primarily consist of Mesh Routers. We use 4 bits of the reserved AODV header to include a Mesh Router count variable, which is incremented every time a RREQ packet is forwarded by a Mesh Router.

Our second modification to AODV maximizes the channel diversity of paths, which comprise of multi-radio nodes. In this case, multiple links can exist between neighboring nodes. The choice of link (interface) to use between two nodes will not affect the routing metric, and we need some other method to select the best link. In our protocol, nodes forwarding a RREQ packet also recommend a channel, which is subsequently used to communicate with the next hop. In order to minimize co channel interference, nodes recommend least loaded channels for next hop communication. If a hop shares multiple channels with the sender of the RREQ packet, it will receive multiple copies of it. If possible, the node will create a Reverse Route via the recommended channel (interface). We use the remaining 7 reserved bits in the AODV header to convey the recommended channel information to the next hop.

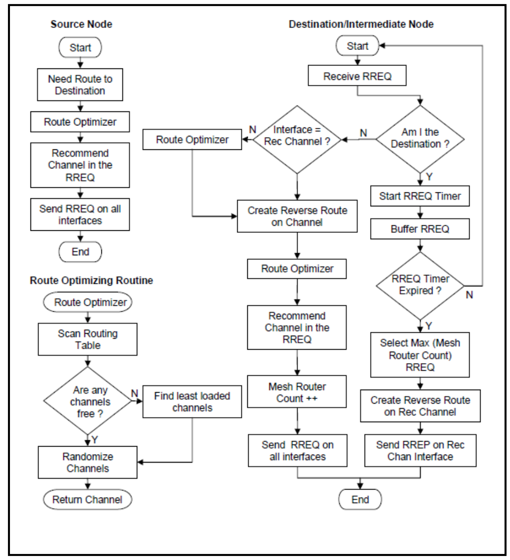


Fig 4.1: Route Establishment and Optimization

The route establishment and optimization process is shown in Fig. 4.1. When a source node needs to discover a route, it first executes the Route Optimization Function (ROF). The ROF first scans the existing routing tables and finds the interfaces which are not being used in any of the active data connections. In case none of the interfaces is free, it examines the Network Interface Queue (IFQ) of each interface. The IFQ is a drop-tail FIFO buffer, established between the Link and MAC layers, and holds packets which are to be transmitted on to the Physical Layer. The IFQ length gives the current number of the packets, which are awaiting transmission.

Algorithmic extension to basic chord design is as follows.

Algorithm : Lookup(data key δ)

$\mu \leftarrow$binary-convert(δ)
lower$\leftarrow$ 2, upper$\leftarrow D + 1$
while lower $\leq$ upper do
mid$\leftarrow$(lower+upper)/2
x$\leftarrow$ μ.prefix(mid)
bucket label$\leftarrow$ DHT-get(fn(x))
if bucket label=NULL then
{
a failed DHT-get
}
upper$\leftarrow$$fn$(x).length
else if bucket label covers δ then

```
        {
                reach the target leaf bucket
        }
        return fn(x)
        else {x is an ancestor of the target leaf node}
        lower← fnn(x,μ).length
        return NULL
```

V PERFORMANCE

A certain number of queries is generated during Chord/MESHCHORD lifetime. Queries are generated uniformly over time (every t_{query} seconds); when a new query is generated, we uniform randomly choose the peer which issues the query on behalf of the client and the ID of the key to be searched is chosen uniformly at random in [0,1] (expressed as an m-bits binary number). In order to better understand the application behavior, we have run separate set of experiments with and without client-generated queries. In case of no client-generated queries, what is evaluated is the packet overhead and load distribution for building and maintain (e.g., updating finger tables) the overlay network. In both sets of experiments, the simulated time interval was proportional to node size.

Let us first consider the total number of overlay- and network-level packets for increasing values of network size, in case of iterative or recursive lookup implementations. In the iterative case, while the number of exchanged overlay-level packets with MESHCHORD is only marginally smaller than with Chord, we have a considerable reduction in number of network-level packets (in the order of 30%). In the recursive case the situation is different: the number of overlay-level packets exchanged with MESHCHORD is bigger than with Chord, owing to the inefficiency of the cross-layering mechanism. However, thanks to the better matching between overlay and physical links achieved by location aware ID assignment, MESHCHORD is considerably superior to Chord in terms of number of exchanged network level packets (percentage reduction in the order of 35%). When comparing the relative Chord/MESHCHORD performance in the iterative and recursive case, we observe as expected a better efficiency at the overlay level of the recursive technique (only marginal improvements in case of MESHCHORD, owing to inefficiency in cross-layering), and a considerable decrease in number of network-level packets for both Chord (in the order of 24%) and MESHCHORD (in the order of 27%) in case of recursive lookup implementation. Thus, MESHCHORD with recursive lookup implementation, despite some inherent inefficiency in the cross layering mechanism, is the best solution for what concerns total network-level traffic.

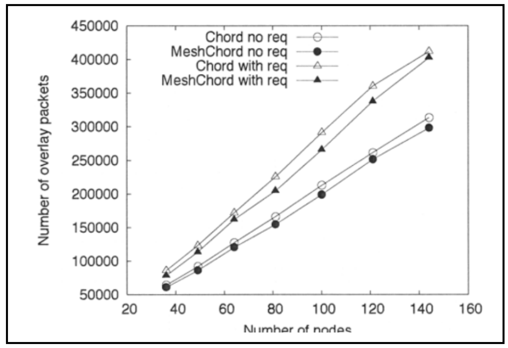


Fig 5.1- Iterative LookUp implementation Comparisons

VI CONCLUSION AND FUTURE WORK

The objective this paper is that, contrary to what functions in MANET environments, the Chord approach can be successfully utilized for implementing p2p sharing context in wireless mesh networks. However, the basic Chord design is effective only under relatively static network conditions and in presence of modest background traffic. With respect to the basic Chord design, our proposed MESHCHORD protocol achieves a considerable reduction in message overhead during lookup delays, and a significant improvement in information retrieval performance. This performance improvement allows an effective realization of the P2P overlay also under very dynamic network conditions and in presence of considerable TCP background traffic. Results indicate that MESHCHORD message overhead does not lead to network congestion by itself, overlay maintenance still requires the exchange of a relatively high number of messages in the network, which could induce performance degradation when other tasks prevalent in p2p applications like trackers, replication detection procedures executed concurrently with MESHCHORD. Quantifying application-layer performance degradation when several applications coexist with the P2P overlay is a matter for future research.

VIII REFERENCES

[1] I. F. Akyildiz and X. Wang, "A Survey on Wireless Mesh Networks," IEEE Communications Magazine, vol. 43, pp. S23-S30, 2005.

[2] A. A. Pirzada and C. McDonald, "Establishing Trust in Pure Ad-hoc Networks," presented at Proceedings of the 27th Australasian Computer Science Conference (ACSC), 2004.

[3] IEEE, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications 802.11," 1997.

[4] I. Stoica, R. Morris, D. Karger, M. Kaashoek, H. Balakrishnan, "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications", Proc. ACM Sigcomm, Aug. 2001.

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

[6] A. Rowstron, P. Druschel, "Pastry: Scalable, Decentralized Object Location and Routing for Large Scale Peer-to-Peer System", Proc. IFIP/ACM Middleware (LNCS 2218), pp. 329–350, 2001.

[7] D. Malkhi, M. Naor, D. Ratajczak, "Viceroy: A Scalable and Dynamic Emulation of the Butterfly", Proc. ACM Symposium on Principles of Distributed Computing (PODC), Jul. 2002.

[8] Claudia Canali, M. Elena Renda, Paolo Santi, and Simone Burresi, Enabling Efficient Peer-to-Peer Resource Sharing in Wireless Mesh Networks, 2008 IEEE

[9] F. Sailhan, V. Issarny, "Scalable Service Discovery for MANET", Proc. IEEE PerCom, 2005.

[10] O. Landsiedel, S. Gotz, K. Wehrle, "Towards Scalable Mobility in Distributed Hash Tables", Proc. IEEE Conf. on Peer-to-Peer Comput- ing, 2006.

[11] P. Desnoyers, D. Ganesan, P. Shenoy, "TSAR: A Two Tier Sensor Storage Architecture Using Interval Skip Graphs", Proc. ACM SenSys, Nov. 2005.