RESEARCH ARTICLE OPEN ACCESS

Reliable Peer-To-Peer Resource Sharing In Wireless Mesh Networks

Seenia Franics[1], Aswathi Wilson[2], Thomas George[3]
Computer Science and Engineering Department,
Jyothi engineering College,
Thrissur, Kerala,
India

ABSTRACT
Wireless mesh networks are a fast developing area for new wireless communication and networking technologies.The design of efficient routing protocols for Mesh Network is challenging for several reasons: Physical network topology is random. Nodes have limited computation and memory capabilities. Energy and bisection bandwidth are scarce. Furthermore, in most settings, the lack of centralized components leaves all network control tasks to the nodes acting as decentralized peers. In this paper, we address the problem of reliable peer-to-peer resource sharing in wireless mesh networks. The well-known Chord protocol for resource sharing in wired networks, we propose a specialization that accounts for peculiar features of wireless mesh networks: namely, the availability of a wireless infrastructure, and the 1-hop broadcast nature of wireless communication, which bring to the notions of location awareness and MAC layer cross-layering. It also include the concept of simultaneous resource sharing in mesh networks. Through extensive packet level simulations, we investigate the separate effects of location awareness and MAC layer cross-layering, and of their combination, on the performance of the P2P application. The combined protocol, MESHCHORD, reduces message overhead of as much as 40 percent with respect to the basic Chord design, while at the same time improving the information retrieval performance. Notably, differently from the basic Chord design, our proposed MESHCHORD specialization displays information retrieval performance resilient to the presence of both CBR and TCP background traffic. Overall, the results of our study suggest that MESHCHORD can be successfully utilized for implementing file/resource sharing applications in wireless mesh networks.

Keywords- Wireless mesh networks, community networks, distributed hash tables, peer-to-peer resource sharing, cross-layering.

I. INTRODUCTION

WIRELESS mesh networks are a promising technology for providing low-cost Internet access to wide areas (entire cities or rural areas), and to enable the creation of new type of applications and services for clients accessing the network. Differently from other types of wireless multihop networks, wireless mesh networks are composed of two types of nodes: mostly stationary wireless access points (routers) and mobile wireless clients. Routers are connected to each other through wireless links, and provide a wireless access infrastructure to wireless clients. Some of the routers are connected to the Internet via wired links, and act as gateways for the other routers and for the clients. Among innovative applications enabled by mesh networking, we mention wireless community networks, in which users in a community (neighborhood, city, rural area, etc.) spontaneously decide to share their communication facilities (wireless access points) and form a wireless multihop network to be used by community members. Wireless community networks can be used to share the cost of broadband Internet access, but also to realize innovative services for the community, such as sharing of community-related resources, live broadcast of local events, distributed backup systems, and so on. As the above mentioned innovative applications suggest, peer-to-peer resource sharing is expected to play an important role in forthcoming wireless networks based on the mesh technology. In this paper, we investigate the feasibility of the well-known Chord algorithm [12] for peer-to-peer resource sharing in wired networks in a wireless mesh network environment.

Starting from the basic Chord design, we propose a specialization—named MESHCHORD—that accounts for peculiar features of mesh networks: namely, 1) the availability of a wireless infrastructure, which enables location-aware ID assignment to peers and 2) the 1-hop broadcast nature of wireless communications, which is exploited through a cross-layering technique that bridges the MAC to the overlay layer.

We evaluate the performance of Chord and MESHCHORD in a wireless mesh network environment through extensive packet-level simulations. The results of the simulations show that MESHCHORD outperforms the basic Chord design both in terms of reduced message overhead for overlay maintenance (mainly achieved by location awareness), and in terms of increased information retrieval efficiency (mainly achieved by cross-layering). Since communication bandwidth is a limited resource in wireless networks, we evaluate Chord/MESHCHORD performance also in presence of different types of background traffic. We consider congestion unaware, Constant Bit Rate (CBR) traffic, and congestion- controlled TCP traffic. To the best of our

knowledge, this is the first similar investigation presented in the literature on P2P approaches for wireless networks. The results of our simulations show that, while Chord information retrieval performance drastically degrades in presence of moderate background traffic, MESHCHORD is relatively resilient to the presence of background traffic, and provides. Acceptable information retrieval performance also in this situation. Interestingly, the lower message overhead generated by MESHCHORD w.r.t. Chord has a positive effect on congestion-controlled TPC traffic, which observes a relatively
Higher throughput.

Thus, differently from the basic Chord design, MESHCHORD has the potential to provide satisfactory performance in a mixed application environment, where the P2P application coexists with different types of application-layer traffic. Summarizing, the study reported in this paper suggests that MESHCHORD can be successfully utilized for implementing resource sharing applications in wireless mesh networks. The rest of this paper is organized as follows: In Section 2, we critically discuss related work and this paper's contribution. In Section 3, we present the basic Chord design and our proposed specialization MESHCHORD for wireless mesh network scenarios. In Section 4, we valuate Chord and MESHCHORD performance through extensive, packet-level simulation. Finally, Section 5 concludes, and outlines possible directions for future work

II. RELATED WORK AND CONTRIBUTION

Several Distributed Hash Table (DHT) approaches have been proposed in the literature to address the problem of realizing distributed peer-to-peer resource sharing. 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 [12] (which we briefly describe in the next section), CAN [9], Pastry [11], 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 not relevant. Recent papers 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 [2] and of Pastry [8]. Others proposed their own solutions, mostly tailored at efficiently dealing with peer mobility. 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. Approaches based on this idea are [2], [6]. 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.

III. MESHCHORD

In this section, we shortly describe the two-tier architecture used in our design, and the basic Chord operations. We then describe in details MESHCHORD, our proposed specialization of Chord for wireless mesh networks. 3.1 Network Architecture Similarly to [5], we assume a two-tier architecture for file/ resource sharing (see Fig. 1): the lower tier of the architecture is composed of (possibly) mobile mesh clients (clients for short), which provide (and use) the content to be shared in the P2P system; the upper tier of the architecture is composed of stationary mesh routers, 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 exclusively to a mesh router forming the DHT at the upper tier of the architecture. We assume routers are stationary and plugged, but they can be switched on/off during network lifetime. This is to account for situations that might arise in some mesh network application scenarios (e.g., community networking), in which routers might be managed by users and occasionally shut down. Also, changes in the upper tier topology might be caused by failures of some router. Mesh clients are the content users and providers: they share file/local resources with other mesh clients, as well as access resources shared by others. When a client u wants to find a certain resource, it sends to its responsible router a (a mesh router within its transmission range) a Find Key message, containing the key (unique ID) of the resource to find (see next section for details on key assignment to node/resources). 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 message 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, and on the procedures needed to deal with client mobility (handoff between responsible mesh routers, locating a mobile client in the network, etc.), and to add/remove shared resources to/ from the distributed index, the reader is referred to [5].

A. Basic Chord Operations

The DHT approach investigated in this paper is Chord [12]. Chord is based on the idea of mapping both peer (mesh router) IDs and resource IDs (keys) into the same ID space, namely, the unit ring ½0; 1_. Each key resides on the peer with the smallest ID larger than the key (see Fig. 2), i.e., peer p manages keys comprised between its own ID and the ID of the predecessor of p in the unit ring (denoted range (p)).

Associated with each key is the IP address of the mesh client holding the corresponding resource? Chord maps peer and resource IDs into the unit ring using a hashing function, named Sha1, which has the property of uniformly distributing IDs in ½0; 1_. Indeed, IDs in Chord are represented through m-bit numbers, i.e., at most 2m distinct (peer or resource) IDs are present in the Chord system. In the following, we set m ¼ 24, which corresponds to having about 16 millions possible IDs. This is a reasonable ID space for wireless mesh networks, in which the number of shared resources is expected to be in the order of several thousands, and the number of mesh routers (peers) in the order of a few hundreds. Indeed, larger ID spaces can be easily included in our design with no modification, and with negligible impact on performance. This allows dealing with larger networks and number of shared resources, which might lead to a non-negligible probability of conflicting ID assignment with relatively low values of m (see also Section 4).

The main operation implemented by Chord is the lookup(x) operation, which can be invoked at any peer to find the IP address of the peer with ID ¼ x, if x is a peer ID, or the IP address of the peer responsible of key x in case x is a resource ID. Lookup operations are used both for query resolution (Find Key operation) and for overlay maintenance. To speed up lookup operations, every peer maintains a table of up to m distinct peers (fingers). The ith finger of peer j, with 1 _ i _ m, is the peer which has the smaller ID larger than j) 2i_1. Note that some of the fingers (especially for low values of i) can actually coincide (see Fig. 2).

In order to facilitate join/leave operations, each peer maintains also the ID of its predecessor in the Chord ring (peer P12 for peer P21 in Fig. 2). When a lookup (k) operation is invoked at peer p and the operation cannot be resolved locally (because k is not within range (p)), a message is sent to the peer p0 with largest ID < k in the finger table of node p. If p0 cannot resolve the lookup operation, it replies to peer p with a message containing the ID of the peer p00 with largest ID < k in its own finger table. Peer p then forwards the request to peer p00, and so on, until the lookup operation can be resolved (in at most m steps).3 Referring to Fig. 2, a lookup operation for key 45 issued at node P21 is first forwarded to node P40, and then to node P49, which is responsible for the key and can resolve the lookup.

To deal with dynamic join/leaves of peers in the systems, the following procedures are implemented: When a new peer p joins the network, it first needs to initialize its predecessor and finger table. This is done by sending requests to any peer currently joining the network peer p is aware of (called hook peer). Then, the finger tables and predecessor pointers of currently active peers must be updated to account for the new peer joining the network. Finally, peer p must contact its successor s in the ring so that the key range previously managed by s can be split with p. In case no (active) hook peer can be found, the join operation fails, and the peer cannot join the Chord overlay. When an existing peer p leaves the network, it first informs its predecessor and successor in the ring about its intention of leaving the network, so that they can change their finger tables and predecessor pointers accordingly; then, peer p transfers to its successor the key range it is responsible for.

Finally, we mention that, in order to deal with dynamic network conditions, each active peer in the network periodically performs a Stabilize operation, which verifies and possibly updates the content of the finger table and predecessor pointer. The period between consecutive Stabilize operations is a critical parameter in the Chord design: if the period is relatively short, the network is more reactive, but a higher message overhead is generated; on the other hand, a longer stabilize period reduces message overhead, at the expense of having a less reactive network. How to set the stabilize period in order to satisfactorily address this tradeoff when Chord is executed in wireless mesh networks is carefully investigated in Section 4.

B. Location Awareness

The first modification we propose to the basic Chord design Concerns the function used to assign ID to peers (hash function Sha1 is still used to assign key to files/resources). 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 messages are exchanged between a peer and its successor/predecessor in the unit ring. More specifically, location awareness is implemented by assigning IDs to peers according to the following function (see [5]):

$$ID(x,y) = \begin{cases} \frac{x\Delta}{s^2} + \lfloor \frac{y}{\Delta} \rfloor \cdot \frac{\Delta}{s} & \text{if} \lfloor \frac{y}{\Delta} \rfloor \text{ is even} \\ \frac{(s-x)\Delta}{s^2} + \lfloor \frac{y}{\Delta} \rfloor \cdot \frac{\Delta}{s} & \text{if} \lfloor \frac{y}{\Delta} \rfloor \text{ is odd} \end{cases} ,$$

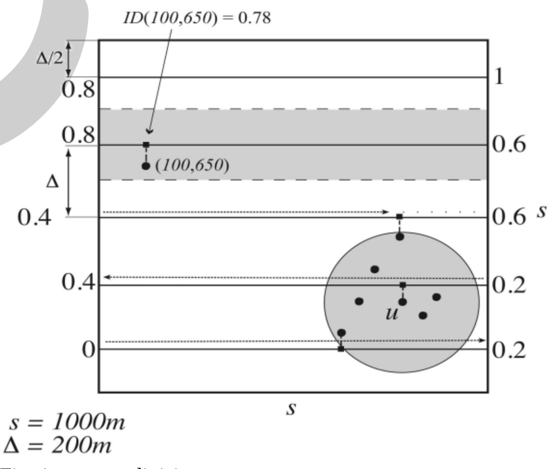

Fig. 1 segment division

Where ID(x, y) is the ID of a peer with coordinate's s is the side of the deployment region, and _ is a parameter which defines the "granularity" of location awareness: the lower the value of _, the closer the peers must be in the physical network in order to be mapped into close-by regions of the unit ring. Fig. 3 shows an example of location-aware ID assignment when s ¼ 1; 000 m and _ ¼ 200 m. The deployment region is divided into s=_ ¼ 5 sub-regions of width _ and length s (shaded area). A segment of the unit ring bisects each sub-region, with alternate left/right orientation. This is required to ensure that peers in the same physical vicinity and close to the border of the deployment region are mapped to close-by segments of the unit ring (see Fig. 3). Each bisecting segment of the unit ring has virtual length _=s. Peers in the same sub -region of the deployment area (shaded area) are mapped to the same segment of the unit ring, thus converting physical proximity into proximity in the unit ring. In the example reported in Fig. 3, a peer located at coordinates (100; 650)is assigned ID 0.78 in the unit ring.

Note that the above location-aware ID assignment function requires that peers are aware of their location, which can be easily accomplished in wireless mesh networks through, e.g., the use of GPS receivers.

C. Cross-Layering

The second contribution of the MESHCHORD proposal concerns the introduction of a MAC cross-layering technique. This technique aims at speeding up the lookup operations by exploiting the information that is available at the MAC layer due to the 1-hop broadcast communication occurring in wireless networks. The basic idea is that a peer u may capture packets for which it owns relevant information, even if they are not destined to u. This technique is motivated by the possibility that peer u, that may actually be able to resolve a lookup request, is physically close to the peer invoking the lookup operation, while they are far away in the unit ring.

More specifically, whenever a peer u receives a packet at the MAC layer, u sends it up to the application layer for further processing, even if the packet was not destined to u. If the packet does not contain a lookup request, it is discarded. Otherwise, u checks if it may resolve the lookup(x) operation. This occurs if x is comprised between u's ID and the ID of the predecessor of u in the unit ring. In this case, u sends a message containing its own ID to the peer that invoked the lookup(x) operation. It is important to note that, since the lookup process is invoked for both query resolution and overlay maintenance, cross-layering may improve the performance of both these operations.

D. Interactions between Location Awareness and Cross-Layering

Location awareness is designed to map neighboring peers to close-by IDs in the unit ring. On the other hand, cross layering tends to be more effective when physical neighbor peers have faraway IDs in the unit ring (this results in a higher likelihood of "capturing" packets). Hence, an ideal ID mapping function should, for a certain peer u, both 1) map pred(u)'s and succ(u)'s IDs to peers which are physical neighbors of u, and 2) assign the remaining u's physical neighbors faraway IDs in the unit ring. The design of a mapping function which achieves both properties 1 and 2 is a very complex combinatorial problem, which is left for future work. In this paper, we focus on property 1, and show that, even in the presence of location-aware peer ID assignment, cross- layering is indeed beneficial to the performance of the P2P application. This can be explained referring back to Fig. 3. In the lower right corner of the deployment area, the transmission range of a peer u is shaded. It is easy to see that, if the transmission range and parameter Δ are adequately set (in the figure, Δ= TX Range= 200 m), although most u's physical neighbors are actually assigned close-by IDs in the unit ring, we do have a number of u's physical neighbors whose IDs are far away in the unit ring. These physical neighbors have opportunities for capturing packets generated by/destined to peer u, explaining the relative benefits of cross-layering that can be observed also in presence of location-aware ID assignment (see Figs. 7 and 10).

E. MESHCHORD Analysis

Using techniques similar to those reported in [5], we provide a theoretical characterization of MESHCHORD performance in random networks, which are formally defined as follows:

Definition 1. In a random network deployment, n peers are distributed uniformly at random in a square region R=(0,s)2. The Peers have transmission range r, with

$$r = 2s\sqrt{\frac{2\log n}{n}}.$$

The specified value of r in the above definition guarantees that the resulting network is connected w.h.p.4 (see [5]). We first show that MESHCHORD location-aware ID assignment preserves the property of uniformly distributing IDs in the unit ring. This result implies that the nice properties of the original Chord design (load balancing on the overlay, query resolution in O (log n) steps, etc.) are preserved in MESHCHORD.

Lemma 1. Assume a random network deployment; then, the peer IDs computed according to mapping ID(x; y) are distributed uniformly at random in the [0, 1] interval (unit ring).

Proof. The proof is along the same lines as proof of Theorem 1 in [5].
We now turn to analyzing the stretch factor, which is Formally defined as follows:

Definition 2. Given a lookup operation on key k on an overlay network O, let l(k) be the hop distance in the physical network between the peer at which the lookup is invoked and the peer that manages the key range to which k belongs; furthermore, let P(k) be the path traversed by the lookup(k) in the overlay network, and let l(P(k)) be the hop length of P(k) in the physical network. The stretch factor is defined as:

$$stretch(O) = \max_k \left\{ \frac{l(P(k))}{l(k)} \right\}$$

Informally speaking, the stretch factor measures how close a virtual overlay is to the topology of the underlying network. The next theorem provides an upper bound to MESHCHORD stretch factor in random networks:

Theorem 1. Assume a random network deployment; then, MESHCHORD stretch factoris O(√(n log n) w.h.p

Proof. Given Lemma 1 and Chord's properties, we have that a lookup operation is resolved traversing O(log n) hops in the virtual overlay, w.h.p. By Lemma 1 of [5], under the assumption of random network deployment, each hop in the virtual overlay corresponds to hops in the physical network,

$$O\left(\sqrt{\frac{n}{\log n}}\right)$$

Hence, a lookup operation is resolved traversing O(√ nlogn) hops in the physical network, w.h.p. We now observe that, given a key k, l(k) ⩾ 1 whenever the peer issuing the lookup(k) operation cannot resolve the query locally. It is easy to see that, given a random key k, Lemma 1 implies that the probability of locally resolving a lookup (k) converges to 0 as n → ∞. It follows that l(k) ⩾ 1 w.h.p., and the theorem is proved.

Note that the bound stated in Theorem 1 is the same holding for the GeoRoy protocol of [5]. This means that, in asymptotic terms, we can expect comparable performance in terms of message overhead between the two protocols. However, MESHCHORD implements also cross-layering, which is not considered in GeoRoy.

IV. CONCLUSIONS

In this paper, we have carefully investigated through Packet-level simulation the performance of the Chord approach for peer-to-peer resource sharing in wireless mesh networks. We have also proposed a specialization of the basic Chord approach called MESHCHORD, which exploits peculiar features of wireless mesh networks (location awareness and 1-hop broadcast nature of wireless communications) to improve performance.

The main finding of the study reported in this paper is that, contrary to what happens in MANET environments [3], the Chord approach can be successfully utilized for implementing file/resource sharing applications 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, 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 background traffic.

REFERENCES

[1] M. Caesar, M. Castro, E.B. Nightingale, G. O'Shea, and A. Rowstron, "Virtual Ring Routing: Network Routing Inspired byDHTs," Proc. ACM SIGCOMM, pp. 351-362, 2006.

[2] M. Conti, E. Gregori, and G. Turi, "A Cross-Layer Optimization of Gnutella for Mobile Ad Hoc Networks," Proc. ACM MobiHoc, May 2005.

[3] C. Cramer and T. Fuhrmann, "Performance Evaluation of Chord in Mobile Ad Hoc Networks," Proc. ACM Int'l Workshop Decentralized Resource Sharing in Mobile Computing and Networking (MobiShare), pp. 48-53, 2006.

[4] T. Fuhrmann, "Scalable Routing for Networked Sensors and Actuators," Proc. IEEE Comm. Soc. Conf. Sensor and Ad Hoc Comm. Networks (SECON), pp. 240-251, 2005.

[5] L. Galluccio, G. Morabito, S. Palazzo, M. Pellegrini, M.E. Renda, and P. Santi, "Georoy: A Location-Aware Enhancement to Viceroy Peer-to-Peer Algorithm," Computer Networks, vol. 51, no. 8, pp. 379-398, June 2007.

[6] D.B. Johnson and D.A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," Mobile Computing, vol. 353, pp. 153-181, 1996.

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

[8] H. Pucha, S.M. Das, and Y.C. Hu, "Ekta: An Efficient DHT Substrate for Distributed Applications in Mobile Ad Hoc Networks," Proc. IEEE Workshop Mobile Computing Systems and Applications (WMCSA), 2004.

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

[10] G. Riley, "The Georgia Tech Network Simulator," Proc. ACM SIGCOMM Workshop Models, Methods and Tools for Reproducible Network Research (MoMeTools), 2003.

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

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