Int. J. Communications, Network and System Sciences, 2018, 11, 147-173
http://www.scirp.org/journal/ijcns
ISSN Online: 1913-3723
ISSN Print: 1913-3715

P2P Overlay Performance in Large-Scale MANETs

Thomas Kunz[1], Babak Esfandiari[1], Silas Ngozi[1], Frank Ockenfeld[2]

[1]Systems and Computer Engineering, Carleton University, Ottawa, Canada
[2]Informatik, Universität Koblenz-Landau, Koblenz, Germany
Email: tkunz@sce.carleton.ca, babak@sce.carleton.ca, SilasEchegini@cmail.carleton.ca, fockenfeld@uni-koblenz.de

How to cite this paper: Kunz, T., Esfandiari, B., Ngozi, S. and Ockenfeld, F. (2018) P2P Overlay Performance in Large-Scale MANETs. *Int. J. Communications, Network and System Sciences*, **11**, 147-173.
https://doi.org/10.4236/ijcns.2018.117009

Received: July 10, 2018
Accepted: July 28, 2018
Published: July 31, 2018

Abstract

We explored how to deploy P2P overlays in ultimately large-scale Mobile Ad-Hoc Networks (MANETs). We therefore studied the performance of P2P overlays such as Chord, creating a number of flat and hierarchical MANET networks. The hierarchical network consists of clusters, interconnected by a backbone. The subnetworks (cluster or the whole network) ran OLSR as network-layer routing protocol. Each cluster had a gateway, interconnected through a backbone that deployed flooding. As we increased the number of clusters, we kept the number of nodes in the Chord overlay constant. Using simulations in *OMNeT*++, we evaluated the P2P performance. Our results show that an unmodified P2P network does not perform well even for relatively small network sizes. The performance can be improved through the use of a cross-layered P2P solution, such as OneHopOverlay4MANET. However, such cross-layered approaches require complete information about overlay nodes from the routing layer and are therefore not suitable in hierarchical MANETs. For hierarchical underlays, the performance of the P2P overlay deteriorated as we increased the number of clusters. One of the main reasons is that the backbone quickly became a performance bottleneck.

Keywords

MANET, Distributed Hash Tables (DHT), OLSR, Chord, OneHopOverlay4MANET, Flat and Hierarchical Routing

1. Introduction

MANETs find relevance in various applications especially where a rapidly deployable network is required; on a campus, during a conference, dealing with emergency operations (like natural disasters and political unrests), military scenarios, etc. The number of users in these application scenarios may vary from just a handful to hundreds of thousands of people and even more [1].

A P2P [2] network is a distributed network in which participants share a part of their hardware resources to provide, together, a certain service and content. Peers are accessible to one another directly, eliminating the need for central intermediary entities to pass through. In the case of a *pure* P2P network, any node can come and go without affecting the overall service, meaning that no central entity is needed at all to offer the service. The fact that no central entity is needed and that nodes can leave and rejoin at will make P2P an ideal choice for MANETs.

One specific application of such a P2P network would be the use of SIP (Session Initiation Protocol) to establish and manage communication sessions among first responders or soldiers. SIP [3] is a protocol standardized by the IETF that is based on a client-server architecture for storing and retrieving user registrations, and therefore it is not appropriate for use in MANETs. A solution to this is to use a P2P overlay (such as Distributed Hash Table (DHT)) to store and retrieve the registrations associated with SIP URIs. Such an approach was standardized by the IETF as the RELOAD protocol [4].

The efficiency of a P2P overlay deployed over a MANET will depend to a large extent on the selecting of a suitable DHT protocol. Many structured and unstructured P2P protocols have been proposed in the past, going all the way back to Chord [5] and CAN [6]. To improve their scalability, two different and complementary avenues could be pursued: cross-layer optimizations on the one hand and exploiting hierarchical network structures on the other hand. This paper discusses our results when evaluating both approaches and draws some conclusions for promising future work.

As discussed in more detail below, running P2P overlays "as is" on top of MANETs typically results in poor overall performance, limiting the usefulness of such P2P overlays in large-scale deployments. P2P overlays can improve their performance by utilizing information already available at the lower layers in a protocol stack, resulting in cross-layer optimizations. A number of possible cross-layer optimizations have been proposed in the literature, and we base our work on a recently-proposed approach that allows peers in the overlay to reach each other in a single logical hop (and the network layer protocol will connect these peers via the best route). As our simulation results show, such cross-layer optimizations do in fact improve the P2P overlay performance, making them interesting to non-trivial network sizes.

To further increase the P2P overlay scalability, changes to the overall network structure may be required to reduce the routing overhead inherent in large-scale MANETs. Many routing protocols for ad-hoc networks are either proactive or reactive on-demand [7] [8]. Proactive routing protocols like OLSR or DSDV originate from the traditional distance vector and link state protocols. They continuously maintain routes to all destinations in a network, whereas reactive on-demand protocols like AODV or DSR will only seek out routes to a destination when necessary, both reactive and proactive (on-demand) routing protocols scale poorly [7] [8]. This is true because of the inherent characteristics of these protocols [9]: on the one hand, the on-demand routing protocols are limited by their route discovery techniques because of the extensive use of flooding. The hop-by-hop flooding usually has a huge negative impact on network performance and often leads to large delays in route discovery [10] [11]. On the other hand, proactive routing protocols have these routes readily available, but it comes at a cost of constant route discovery throughout the lifetime of the network. It is evident therefore that both protocols have scalability issues, which get even worse in the case that nodes are mobile and links become generally unpredictable [10] [11].

Hierarchical routing architectures, when carefully planned, simplify routing tables considerably and lower the amount of routing information exchanged [12] [13]. The IP protocol implements hierarchical network addressing. IP networks have a hierarchical routing structure and networks are divided into routing domains. A routing domain typically contains a collection of co-located networks connected by routers (who are nodes) that share the routing information for the routes within said domain. Routing domains are connected by a common routing domain called the *backbone*. Within a domain, routing is performed by the nodes within that domain, whereas between domains routing is performed by domain routers connected in the backbone [12]. Nodes in each routing domain or cluster will not have to worry about the topology of the entire network but each will maintain a specific route to the gateway node (router) who will be responsible for connectivity with other clusters through the backbone. This way, routing table entries of each node are determined based only on information of nodes within the local routing domain.

A systematic evaluation and comparison of the scalability of both above approaches when deploying P2P overlays in larger MANETs is missing in the literature. This paper discusses our work to-date conducting just such an evaluation. The results point out that, for P2P overlays to become a reality in such MANETs, more sophisticated approaches may be required, and we discuss some of our efforts towards this goal at the end.

The rest of this paper is organized as follows. Section 2 presents background material on P2P overlays, including the challenges in running P2P overlays in a MANET. Section 3 discusses related work. Section 4 discusses various P2P overlay protocols we use in our work. In Section 5, we briefly describe the physical network architecture (*i.e.*, the underlay), in particular its hierarchical structure, and routing in such a network. Section 6 describes experimental setups and discusses simulation results based on simulations in *OMNeT*++. Finally, Section 7 includes a brief discussion and concludes this paper with an outlook on our future work.

2. Background

2.1. P2P Overlays

P2P overlays have been widely researched [14] [15]. A P2P protocol generally implements some form of virtual overlay network on top of a physical network topology. Nodes in the overlay form a subset of the nodes in the underlay network. P2P overlays are used for decentralized indexing and peer discovery, which can usually be achieved within a bounded number of hops [16] [17]. Data is still exchanged directly over the underlying TCP/IP network by mapping onto the underlay physical topology. Routing is achieved by the underlay routing protocol. At the application layer, peers are able to communicate with each other directly via the logical overlay links, each of which corresponds to a path through the underlying physical network [14] [16].

P2P networks can be structured or unstructured. In unstructured P2P networks, there is no relationship between any one node and any distributed network resource. Retrieving any network resource generally involves random walks and/or flooding approaches, which are inefficient and do not guarantee the discovery of a resource [18]. This is addressed in structured P2P systems, which provide efficient search strategies that guarantee content location within a small number of hops. Certain peers participating in the P2P network are responsible for certain resources [19] [20] through a mapping between the node identifier and the identifier of the resource, and data lookup query is directed towards the particular peer responsible for the requested content. Such mapping is often achieved by hashing the node identifier (such as the IP Address) and the data. Systems who support such an approach arrange these peers in various topologies to help limit the number of routing messages exchanged to access the resource. Some protocols generate a ring structure, as in Chord [17] [21], others generate a mesh, as seen in Pastry [18] [22]. For all these structured P2P overlays, the established logical topologies need to be maintained as the P2P system experiences churn (peers joining and leaving), resulting in a potentially significant amount of maintenance traffic. In the case of Chord, for example, nodes will probe their successors/predecessors to maintain the ring. Also, the main routing structure, called finger table, has to be updated every time the ring changes.

2.2. Challenges of P2P Overlays in MANETs

P2P overlays were originally designed for wired networks, and therefore deploying a P2P overlay over a MANET raises a number of challenging issues, as reviewed in [23]:

- **Limited bandwidth:** MANET bandwidth resources are more constrained compared to a wired IP infrastructure. P2P overlay algorithms have a high maintenance overhead and are quite wasteful with bandwidth and hence are potentially unsuitable for MANETs.
- **Logical overlay maintenance:** To maintain the routing tables, DHT protocols

are periodically sending maintenance requests and responses for route discovery and learn about unavailable peers. This increases the traffic and makes it undesirable for MANETs.

- **Physical topology changes:** Node mobility leads to breaking links between nodes and churn (leaving and joining nodes) to changes in the physical infrastructure. The P2P system needs to be informed about these changes. Node mobility is a completely new issue that has not been taken into account by P2P protocols.
- **Routing stretch:** Each logical hop in the overlay corresponds to a physical path in the underlay. Two nodes which are close in the overlay may be far apart in the MANET. Also, nodes which are physically close may be multiple logical hops apart. The minimization of routing stretch has to be considered when building the overlay routing table.
- **Infrastructure-less operation:** The lack of infrastructure makes the use of P2P protocols difficult. Therefore, a P2P protocol has to be highly adaptable and accommodated to its new setting. For example, CAN [6] relies on using static landmarks when assigning logical IDs, However, such landmarks do not exists in an infrastructure-less environment.
- **Battery power:** Most P2P overlay algorithms were designed for wired environment and are quite wasteful with resources (maintenance messages, request response etc.). In particular, they do not take into account the constraints imposed by limited battery power. Maintenance messages and other overhead have to be minimized to conserve battery power.

3. Related Work

3.1. P2P and Cross-Layer Optimizations

MANET networks and P2P applications are operating at different layers in the protocol stack. There are three different deployment approaches to accomplish a data exchange between those two different layers and also to handle the previous challenges of deploying a P2P system over a MANET:

- The legacy approach (a.k.a. *layered design*) describes a design pattern in which the overlay and underlay are strictly separated from each other. It builds the P2P overlay on top of the network layer. Both layers, the application and network layer will operate their own routing algorithms, there is no synergy between network and application layer, which leads to poor performance. **Figure 1** shows this approach diagrammatically, Backtrack Chord is an example of a legacy approach [24].
- To reduce the maintenance overhead between those two layers, the P2P overlay needs to know about the states of the underlay network. The *cross-layer design* creates a cross-layer interaction and violates therefore the layered architecture. The sharing of information between network and application layer typically improves the overall P2P performance, however. For example, each node at the network layer will send its routing table information

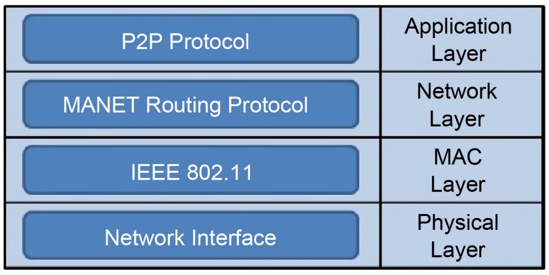

Figure 1. P2P and MANETs: Layered approach.

whenever a change occurs to the cross-layer (notification board). The overlay protocol will receive this information from the board and uses it to populate its cache lists and to build the overlay structure. **Figure 2** depicts this approach in general, CrossROAD uses this approach to allow for a communication between Pastry and OLSR [25].

- The *integrated design* integrates P2P algorithms directly into the network layer and reduces the information exchange between network and application layer. This approach also violates the layered architecture, by embedding application protocol functionalities into the network layer. **Figure 3** outlines this approach, Ekta for example integrates DSR (Dynamic Source Routing) functions into Pastry to improve routing performance [26].

3.2. P2P and Hierarchical MANETs

MANETs in general have been widely researched in recent years owing to the continuous discovery of approaches for deploying them in different scenarios or environments. These approaches however are not without their challenges. Considerable research has been devoted to P2P overlay protocols in the Internet [17] [27], with some related to P2P in mobile ad-hoc networks. However, many of the publications merely present architecture proposals and surveys [28] [29] [30]. The authors of [31] present an in-depth investigation of Chord over static and dynamic Wireless Sensor Networks (WSNs) but only consider a flat network architecture. Their investigation shows that structured algorithms like Chord are an efficient lookup algorithm for distributed P2P networking applications. However, significant overhead is generated in overlay networks that negatively impact on performance.

The authors of [28] [29] investigated a P2P file-sharing over MANET and proposed five routing approaches of different complexity. One approach integrated the DHT at the network layer, as proposed in [30]. Another approach combined the DHT with flooding, similar to what we adopt in this paper. They mentioned the complexity of route maintenance in DHT over MANET structures in general and energy constraints, but did not discuss in-depth nor show any performance results with respect to particular network architectures (whether a flat network or hierarchical). In [32], the authors evaluate the behavior of Bamboo, a structured P2P overlay for dynamic environments, in a static multi-hop environment. The authors investigate the challenges when deploying

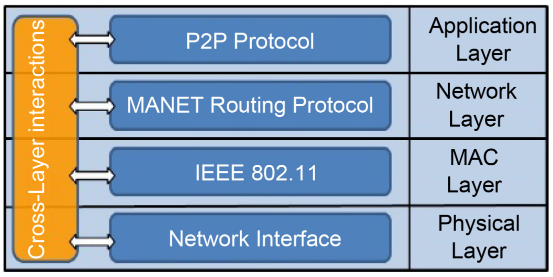

Figure 2. P2P and MANETs: Cross-layered approach.

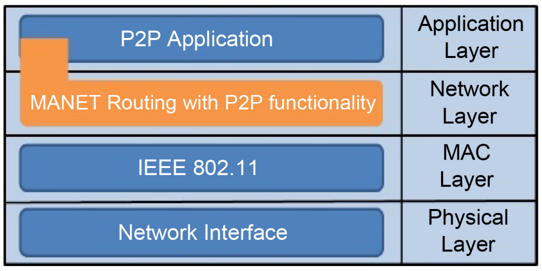

Figure 3. P2P and MANETs: Integrated approach.

structured overlays using AODV as underlay routing protocol, varying the number of nodes. They do not investigate the performance for multiple hierarchical network architectures and in the presence of node mobility. However, they point out that overlays will be necessary for medium to large scale deployments of wireless multi-hop networks, similar to the assumption underlying our work.

In summary, and as presented in [28] [31] [33] [34], structured P2P overlays based on DHTs outperform unstructured approaches when the number of nodes, resources, or the query ratios increase and are better suited for large scale MANET topologies spanning large geographical areas. What is missing is an evaluation of specific structured overlays over hierarchical networks of different sizes, the key contribution of this paper. To this end, Section 4 first summarizes our chosen P2P overlay protocols in some detail, while Section 5 explains our approach to modeling/simulating MANETs that have a hierarchical structure.

4. Cross-Layered P2P Designs

We are interested in evaluating the suitability of various (structured) P2P protocols in the presence of mobility and wireless links, starting with flat (*i.e.*, non-hierarchical) MANETs. We focus on DHTs as these are commonly used to support P2P applications in the Internet, see for example [35] to support P2PSIP. This section briefly describes three DHT protocols we selected for an in-depth evaluation: As base case, we choose Chord [5]. The other two protocols improve upon Chord in different ways: EpiChord [36] provides for parallel lookups and implements a more efficient cache structure, while OneHopOverlay4MANET [37] uses cross-layering by exchanging routing information between underlay and overlay.

4.1. Chord

Chord [5] stores key-value pairs by assigning keys to different 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. Nodes and keys are assigned an m-bit identifier using consistent hashing. Nodes and keys are arranged in an identifier circle that has at most 2^m nodes, ranging from 0 to 2^{m-1}. Each node has a successor and a predecessor. The successor to a node is the next node in the identifier circle in a clockwise direction. The predecessor is the next node in the identifier circle in the counter-clockwise direction. The concept of successor can be used for keys as well. The successor node of a key k is the first node whose ID equals to k or follows k in the identifier circle, denoted by *successor*(k). Every key is assigned to (stored at) its successor node, so looking up a key k is to query *successor*(k).

The core usage of the Chord protocol is to query a key from a client (generally a node as well), *i.e.* to find *successor*(k). The basic approach is to pass the query to a node's successor, if it cannot find the key locally. This will lead to a O(N) query time where N is the number of nodes in the ring. To avoid this linear search, Chord implements a faster search method by requiring each node to keep a finger table containing up to m entries. Recall that m is the number of bits in the hash key. The i^{th} entry of node n will contain *successor* ($n + 2^{i-1} \bmod 2^m$). Every time a node wants to look up a key k, it will pass the query to the closest successor or predecessor (depending on the finger table) of k in its finger table (the "largest" one on the circle whose ID is smaller than k), until a node finds out the key is stored in its immediate successor. With such a finger table, the number of nodes that must be contacted to find a successor in an N-node network is O(log N).

The protocol specifies how nodes can join and leave such a DHT, redistributing key values as the group of nodes changes. The connections among the peer nodes, stored in the predecessor and successor lists, as well as the finger table, are logical hops in the overlay. Routing between these entries requires the traversal of multiple physical hops, which as achieved by the underlying routing protocol. As discussed previously, in the absence of any interactions between underlay and overlay, the logical topology may be a poor fit for the underlying physical topology: nodes physically far apart may be neighbors in the overlay and vice versa. In a network where the topology is dynamic, this will lead to many long physical routes with a high chance of routing failure, which in turn results in lookup failures at the DHT overlay.

4.2. EpiChord

EpiChord [36] enhances the Chord DHT lookup algorithm with improvements on lookup performance and maintenance traffic reduction. EpiChord is able to achieve O(1)-hop lookups, and, in the worst case under heavy network load, an

O(log n) performance. Nodes populate their caches (successor, predecessor, cache list) by observing network traffic, which significantly reduces the network load. EpiChord nodes only sends probes to ensure stability of the basic ring structure as a backup mechanism if the lookup traffic rate is too low. Simulations have shown that EpiChord reduces lookup latencies and path lengths by a factor of 3 [36], compared to Chord.

To locate a given key *k*, the node initiates p queries in parallel to the node succeeding and the p − 1 nodes preceding the key. If the lookup fails, it iterates through all preceding and succeeding nodes. If a node is probed and…

- it owns the key it responds with the value associated with the key and information about current immediate predecessors.
- it is a predecessor of the key, it will provide information about its immediate successor and the next best-known hop to the node storing the key.
- it is a successor of the key, it will provide information about its immediate predecessor and also the next best-known hop to the node storing the key.

4.3. OneHopOverlay4MANET

The OneHopOverlay4MANET protocol is a DHT based P2P overlay network and builds a ring structure similar to Chord and EpiChord, to assure lookups in one logical hop. It also uses the cross-layer approach to pass routing information between network and application layer for increased lookup performance. The improvement is achieved in one of two ways: the protocol reduces the maintenance traffic, and logical lookups (which are typically successful over a single hop) are forwarded over the optimal physical path as determined by the underlay routing protocol. The protocol uses the manager-based method of cross-layering, sharing information between application and network layer through a notification board.

Every peer maintains logical routing information to every other peer in the overlay, which enables the protocol to fetch a key in one logical hop (O(1)). Every peer stores 4 keys (+ timestamp) of the closest nodes to its own ID in the predecessor and successor list, other keys are stored in a cache. These lists and the cache are populated via the routing information from the underlay through cross-layering. Every underlay node sends its routing table information whenever a change occurs to the notification board. A peer subscribes to this notification board to receive updates about changes at the physical level. The received data is then used to populate and maintain its cache lists. In the case of a pro-active underlay routing protocol such as OLSR, no additional control messages at the overlay are required to learn about all peers in the network.

OneHopOverlay4MANET sends single lookup requests to single destinations to reduce network load. A peer will respond to a lookup query in one of the following ways:

- If it is a successor of the looked up key, it will respond with information about its own predecessor, and information about the node succeeding and the two nodes preceding the best node that holds the key.

- If it is a predecessor of the looked up key, it responds with information about its own successor, information about the node succeeding and the two nodes preceding the best node that may hold the key.
- If it is the immediate successor of the key, it responds with the value of the key and information about its successor and predecessor

In case of a failure to resolve the lookup in one logical hop, it reverts to DHT routing and goes through peers one by one in the underlying Chord ring.

Differences to EpiChord: OneHopOverlay4MANET does not use parallel lookups, it only sends single lookup requests to reduce network load. No probing stabilization messages for new neighbor discovery or consistency check of the overlay are sent, as the overlay routing tables are already assumed up-to-date from the underlay updates through the notification board. EpiChord divides the logical address spaces into slices to keep the overlay consistent and each peer has to maintain a certain number of entries per slice. OneHopOverlay4MANET does not employ this functionality, as again its routing tables are already well populated due to the cross-layer information exchange from the underlay.

5. P2P over Hierarchical MANETs

We model a hierarchical MANET as shown in Figure 4. A number of individual MANETs, each running its own local routing protocol, are interconnected through a backbone. Each MANET can be thought of as a cluster, with a cluster head or gateway that provides nodes within the cluster access to the backbone

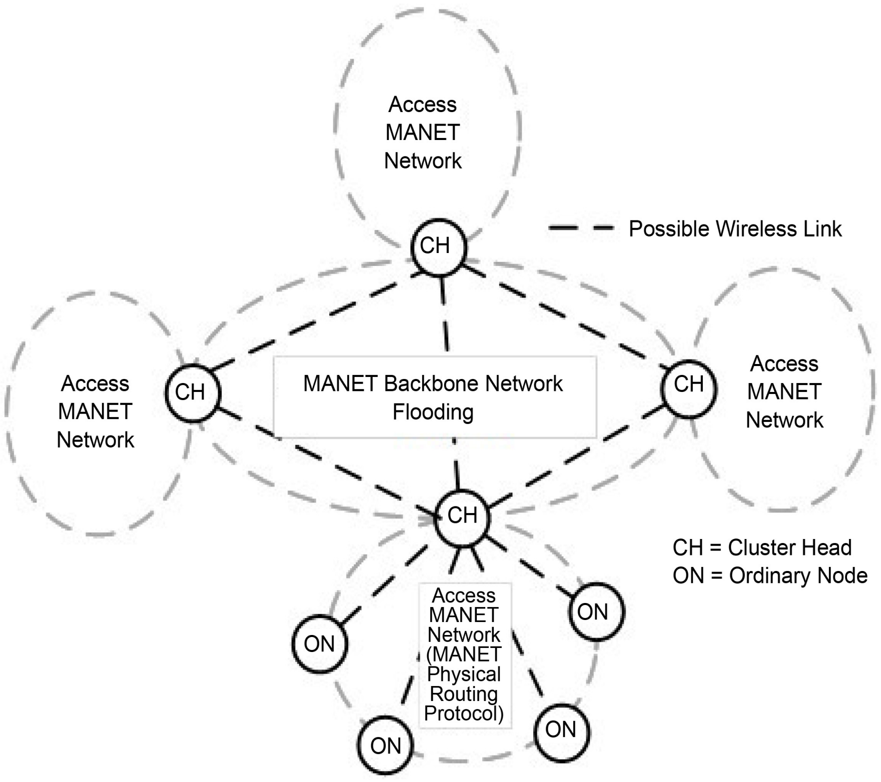

Figure 4. Hierarchical MANET.

(and through it, to all other nodes in the network). Different routing options exist in the backbone, and for reasons explained below, we choose to employ flooding.

The hierarchical structure of IP addresses facilitates the hierarchical routing approach in fixed networks such as the Internet. In our context, however, nodes do not necessarily belong to a single network throughout their lifetime. As nodes are mobile, they may change their cluster membership, clusters may join the network or leave, or clusters may merge and split. So we require a more general hierarchical routing architecture that supports various mobility scenarios.

In our design, each cluster runs OLSR as MANET/cluster routing protocol, as it has a number of useful capabilities such as HNA (Host Network Association) messages. Each cluster has a gateway, which advertises reachability to all other nodes outside a cluster by periodically flooding an HNA message within the local cluster. These HNA messages essentially advertise a global default route through this gateway to all other nodes in the cluster. The gateways are interconnected through a backbone, and a simple, robust but costly routing solution is to flood all messages through this backbone. Each gateway will receive such packets, and, based on the destination IP address and its own cluster-specific routing table, determine whether the packet should be forwarded inside its associated cluster. This will allow nodes to communicate between clusters, and support host mobility: as a host X leaves cluster A and joins cluster B, the cluster A gateway will lose the routing table entry for host X, but the gateway for cluster B will eventually contain a routing table entry for X. Packets destined to X will then not be picked up and forwarded by the cluster A gateway but the gateway for cluster B.

6. Experiments

This section describes the experimental setup, performance metrics, and results of our study to evaluate the scalability of P2P overlays in larger MANETs. We start with an evaluation of various DHT protocols over a flat MANETs, followed by experiments that evaluate the impact of a hierarchical architecture on the P2P overlay performance.

6.1. Flat MANETs

In a first evaluation, we implemented OneHopOverlay4MANET [37] in *OMNeT++* Version 4.6, using both the *INET framework*, Version 2.0, which provides implementations of MANET routing protocols such as OLSR, and the *OverSim framework*, Version 20121206, which provides implementations of common P2P protocols such as Chord and EpiChord. The goal is to determine whether P2P overlays can provide reasonably good performance for non-trivial MANETs under different mobility models. We first describe the simulation parameters and metrics, followed by the data we collected and a discussion of the results.

6.1.1. Simulation Setup and Metrics

To compare the performance of the three DHT protocols, we ran a number of simulations under the same conditions. Table 1 summarizes the main simulation parameters. Every parameter is fixed through all simulations. Every simulation is repeated 5 times and the results are averaged to reduce measurement noise.

To evaluate and compare the DHT performance, we use a collection of metrics that capture both the efficiency and effectiveness of the protocols. More specifically, we measured the lookup success ratio (successful lookups in the DHT as a percentage of total attempted lookups), the lookup latency of successful lookups, the average number of logical hops that successful lookups traveled in the DHT, and the network traffic (in bytes/sec) generated by the DHT protocol. All protocols require a routing protocol at the underlay, so we only count the additional P2P traffic generated by each protocol.

We are particularly interested in evaluating the performance of the three candidate protocols under different mobility scenarios. We selected 4 distinct mobility models for that purpose:

- Stationary: all nodes are distributed uniformly across the simulation area, and remain at their location throughput the simulation. This random initial distribution of node locations is true for all other mobility models as well, however, nodes also move during the simulation following specific patterns.
- Random WayPoint: In the Random Waypoint mobility model the nodes move in line segments. For each line segment, a random destination position (distributed uniformly over the area) and a random speed is chosen. Once a node arrives at the destination, it waits for a specified amount of time before repeating this process. We use a relatively high rate of mobility, with average node speeds of 20 meters/sec, a standard deviation of 8 meters/sec, and 0 wait time once nodes reach their destination.

Table 1. Simulation parameters.

Simulator	OMNeT++ 4.6
Underlay routing protocol	OLSR
MAC protocol	IEEE 802.11
Topology size	2000 m × 2000 m
Number of Nodes	30
Simulation time	500 seconds
Transmission range	750 m
Network stabilization time	20 seconds
Join delay	20 seconds
Lookup interval	30 seconds,
Parallelism	3 (EpiChord only)
Simulation repetition	5 times

- Linear Mobility: This is a linear mobility model with speed, angle and acceleration parameters. The angle only changes when the mobile node hits a simulation area boundary: then it reflects off the boundary at the same angle. We use node speeds that average 20 meters/second, with a standard deviation of 8 meters/second.
- Mass Mobility: A node moves within the simulation area according to the following pattern. It moves along a straight line for a certain period of time before it makes a turn. This moving period is a random number, distributed with an average of 2 seconds and standard deviation of 0.5 second. When it turns, the new direction (angle) in which it will move is a normally distributed random number with average equal to the previous direction and standard deviation of 30 degrees. Its speed is also a normally distributed random number, with an average speed of 20 meters/second, with a standard deviation of 8 meters/sec. This pattern of mobility is intended to model node movement during which the nodes have momentum, and thus do not start, stop, or turn abruptly (unlike the Random WayPoint model). When a node hits a simulation area boundary, it reflects off the boundary at the same angle.

6.1.2. Simulation Results

Figure 5 shows that, as expected, Chord performs poorly overall. Its lookup success ratio is already relatively low in a purely static scenario, where messages get lost due to collisions. As nodes move and the network topology starts to change, its performance deteriorates further, resulting in a lookup success ratio of as low as 15%. EpiChord does better but also visibly suffers from mobility. OneHopOverlay4MANET performs significantly better, achieving very high lookup success ratios in static scenarios, deteriorating somewhat with mobility.

Figure 6 shows the average latency for successful DHT lookups. In a stationary network, OneHopeOverlay4MANET has the lowest lookup latency. The latency increases under mobility, as lookup traffic now competes with triggered updates to reflect topology changes in the underlay routing protocol. The lookup latency in particular for Chord seems to outperform the other protocols, but this is skewed by the fact that a much smaller percentage of lookups are successful. These lookups are typically to nodes that are physically close.

Figure 7 shows the logical hop counts for successful lookups. In a stationary network, Chord lookups traverse almost 2.5 logical hops, adding a lot of forwarding traffic to the network. EpiChord on the other hand is able to resolve every lookup in about 1.2 logical hops. This is due to the parallel lookup mechanism and the additional predecessor cache list for a more efficient lookup routine. OneHopOverlay4MANET resolves lookups in one logical hop. This single logical hop is than mapped by the underlay routing protocol to the ideal route (as defined by the routing metric). Under mobility, the logical hop count drops for Chord and EpiChord, as lookups resulting in longer traversals of the DHT typically fail. OneHopOverlay4MANET consistently achieves a high

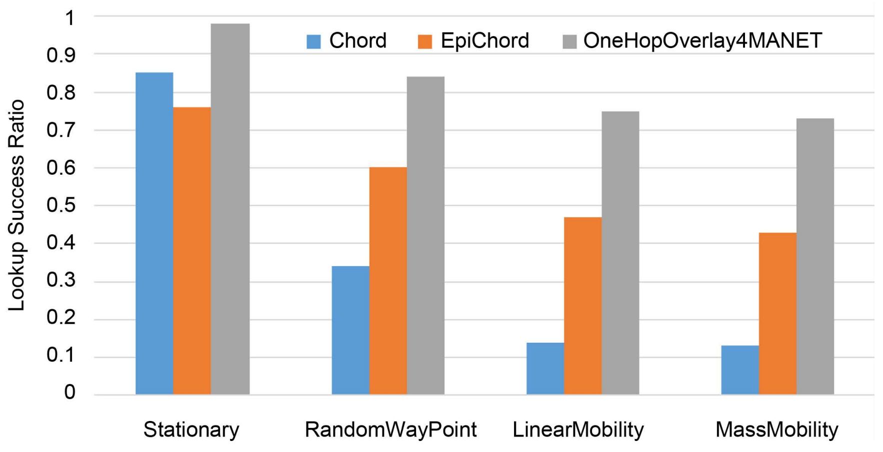

Figure 5. Lookup success ratios.

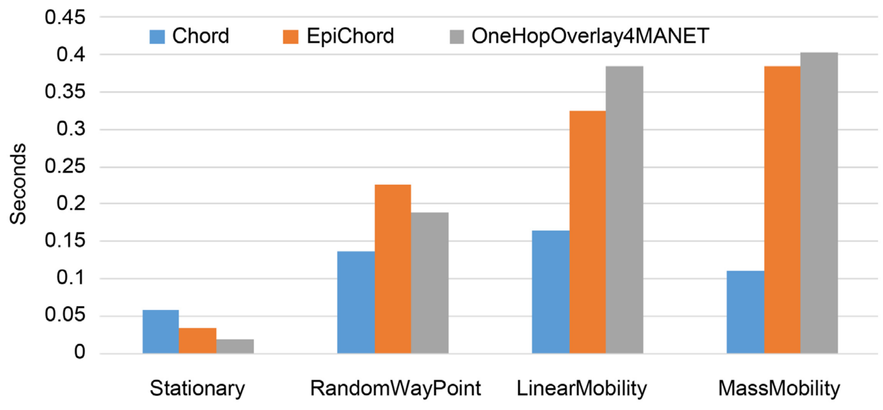

Figure 6. Lookup latencies.

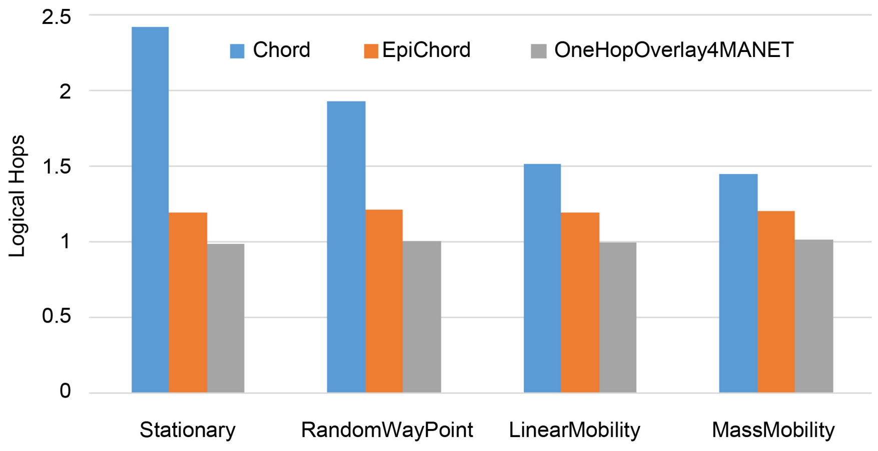

Figure 7. Logical hop counts.

percentage of successful lookups over a single logical hop. As DHT requests travel fewer logical hops, this also reduces the network traffic. We measured the additional network traffic generated by each DHT protocol. In a stationary network, where all three protocols achieve comparable lookup success rates, the DHT traffic (including the lookups themselves) is significantly lower for One-

HopOverlay4MANET (30% of Chord's traffic and 60% of EpiChord's traffic).

6.1.3. Discussion

As expected from the literature review, running a layered P2P overlay over a MANET protocol stack results in poor performance, even for networks of relatively modest size. The performance can be significantly improved using cross-layer optimizations, as shown here for OneHopOverlay4MANET. In a stationary network, this protocol achieves almost 100% lookup success ratio and induces low overheads. However, this performance starts to deteriorate under mobility. The scalability of this approach is likewise limited, as the underlying MANET routing protocols do not scale well.

6.2. Hierarchical MANETs

A second avenue to increase the scalability of a P2P overlay is to introduce a hierarchical structure into the network, reducing the routing protocol control message overhead. Individual nodes no longer know routes to all possible destinations, which precludes the use of the cross-layered optimizations in OneHopOverlay4MANET. We therefore focus primarily on Chord as representative P2P overlay. As we know from the results in the preceding sections, Chord does not perform particularly well for larger networks and more aggressive mobility models. We therefore reduced the scenarios to a smaller number of nodes and only consider one mobility model, the Random Waypoint model.

6.2.1. Simulation Setup and Metrics

The simulated network has 20 nodes (10 gateways and 10 hosts). The main network components are as follows:

- *Backbone*: The backbone refers to the part of the overall MANET that interconnects the smaller MANETs (clusters), providing a path for the exchange of information through the cluster gateways. Only cluster gateways are members of the backbone, and communication in this region is done via flooding. Communication via the backbone only becomes necessary when there is a need to reach other nodes in other clusters.
- *Cluster gateway*: Within each cluster, all participants use a proactive routing protocol (OLSR) [38] for intra-cluster routing. The gateway in each cluster only provides connectivity as a router/relay for inter-cluster communication via the backbone. The other role of gateways is to periodically inject Host and Network Association (HNA) messages, (an OLSR type IV message) into their local network domain according to the *RFC*-3626 specification [39]. This provides their cluster members with a dedicated route for inter-cluster communication.
- *Hosts*: Hosts are nodes who participate in the overlay but are not gateway nodes.

We ran each simulation for 600 seconds and repeated each scenario 10 times to introduce some randomness. In each run, the 10 hosts are placed, together with the 10 gateway nodes, within a simulation area of 2000 m × 2000 m. The idea is to keep the total number of overlay nodes constant while varying the number of clusters. For configurations where not all gateway nodes are actively providing connectivity among clusters, the remaining gateway nodes act as regular wireless nodes in the underlay, participating to support routing/physical connectivity. The first host to join Chord is the bootstrap node who then broadcasts join call messages to all potential overlay hosts. After successfully joining the overlay, every 5 seconds, every host generates a PUT message with a random key to store data in the overlay. After about 100 seconds of simulation time, hosts start issuing random GET queries for resources stored in the DHT. They do so periodically, every 30 seconds. The DHT application that every host runs keeps track of every resource stored in the DHT. Thus, an overlay host will only issue a GET query for resources available in the DHT. Therefore, if a GET query fails, it will not be as a result of the unavailability of the requested resource in the DHT because such a request would not have been issued in the first place.

For a flat network architecture, queries will be resolved within a single cluster. As the number of clusters increases, the gateways will then be involved in routing queries to hosts outside a cluster. To quantitatively evaluate the protocol performance, the following three metrics are collected during the simulation runs: PUT/GET Success Ratios, PUT/GET Latency, and Backbone Traffic.

6.2.2. Simulation Results

In this section, we discuss the results of our simulations for various scenarios. Statistics are presented for different network configurations; first a flat network of 20 nodes, then 2 clusters of 10 nodes each, 4 clusters of 5 nodes each, 5 clusters of 4 nodes each and finally 10 clusters of 2 nodes each. The margin of error displayed on all graphs represents the 95% confidence intervals.

As earlier stated, and with reference to [12] [13], hierarchical routing increases the scalability of large networks by increasing the robustness of routes and reducing the amount of network topology information each node/router has to track. Combining this with a P2P resource sharing mechanism like Chord, for example, will enable decentralized resource sharing among peers in large-scale networks. In the case of OLSR, all protocol control messages are broadcast, and **Figure 8** shows the number of routing protocol messages broadcast at the MAC layer for different numbers of clusters. The orange line shows the total number of HELLO messages, the olive line shows the total number of HNA messages and the pink line shows the total number of TC messages.

Unsurprisingly, the number of HELLO messages is constant across/independent of the number of clusters. Every node broadcasts a Hello message every 2 seconds and according to *rfc* 3626 [39], "HELLO messages MUST not be forwarded." HNA messages on the other hand are sent every 5 seconds and only by the gateway nodes when our network has two or more clusters. As we increase the number of clusters, keeping the number of nodes N constant, each cluster contains fewer nodes, resulting in a reduction of MPRs. As the MPRs implement

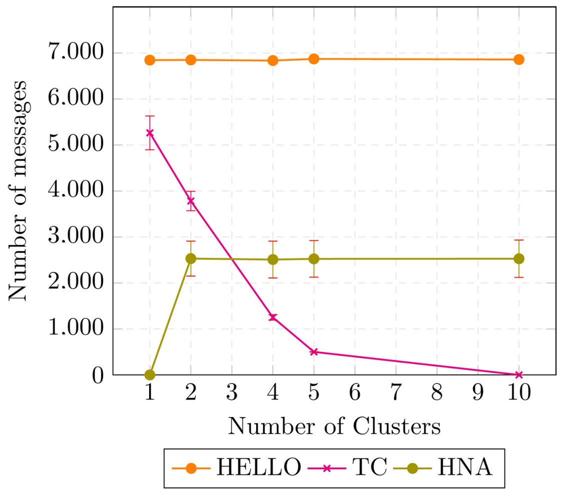

Figure 8. Number of broadcast control messages with varying number of clusters.

network-wide broadcasting by retransmitting HNA and TC messages, the broadcast cost will be lower in smaller clusters. This balances out the increased generation of HNA messages as the number of gateway nodes/clusters grows, as shown in **Figure 8**. The reduction in the number of TC messages as the number of clusters increases on the other hand clearly demonstrates the potential benefits of hierarchical routing. Adding up all OLSR control messages, the number drops from over 12,000 MAC layer broadcasts in a flat network to slightly over 9000 broadcasts in the case of ten clusters.

The metrics used for performance measure are lookup success, lookup latency (RTT), and the level of traffic in the backbone network. All graphs presented below show these metrics as the number of clusters increases (*x-axis*) for both static and mobile scenarios. A light-green line indicates the percentage of PUT requests that were successful while a leaf-green line shows the percentage of GET requests that were successful. A cyan line indicates the latency for PUT requests and a blue line indicates the latency for GET requests. Finally a brown line indicates traffic in the backbone counting both user and control data while an orange line indicates only the control traffic in the backbone (measured in the absence of any PUT/GET requests).

In the static scenarios, nodes are spread out within the network area and retain their positions for each simulation run, but assume new random positions for different runs. For the flat network architecture, there is no activity with the gateway nodes save for routing support in the underlay network. Thus, there is no active gateway node. In the 2, 4, 5 and 10 cluster scenarios, having 10, 5, 4 and 2 nodes each, there are 2, 4, 5, and 10 active gateways per scenario who provide connectivity from their clusters to other clusters via the backbone network as stated above. The 10 hosts always join Chord within a short period of time (*usually* 8 *to* 45 *seconds*) and participate in the overlay routing. In the flat network, all queries are resolved within the same cluster, with the underlay routing protocol *OLSR* providing the shortest route and requests are routed in a multi-hop fashion [38]. All hosts are equipped with single IEEE 802.11b wireless radio while the gateway nodes have two radios: one for communication with other nodes in the same cluster, the second one for communication in the backbone.

The success ratio represents the percentage of Chord operations (PUT or GET) that eventually succeed. **Figure 9** shows these ratios as a function of the number of clusters for both PUT and GET. There is no statistically significant difference in the success ratios for PUTs and GETs—they are fairly close to each other, their 95% confidence intervals overlap and the mean success ratios fall into the overlap region. For the flat network (single cluster), only a few operations ultimately fail. As the number of clusters increases, the success ratios drop. We believe this is a result of the increase in packet collisions in the backbone with increased overhead traffic due to flooding.

A few references, such as [8] [31] postulate that hierarchal routing, in general, will result in higher end-to-end latency/delay. The statistics we collected confirm this assertion, see also **Figure 10**. As the number of clusters increases, the backbone carries more and more traffic, becoming more congested. So connections through the backbone will incur more delays, leading to the increased end-to-end latency.

The trend in both previous metrics makes sense if we assume that the backbone becomes a traffic bottleneck. With a flooding based protocol in the backbone, the number of packets through the backbone increases as more hosts are located in other clusters. The statistics we gathered from our simulations show that, as the number of clusters increases, the number of packets going through the backbone also increases, and this is true for both user data and maintenance traffic. This can be seen in **Figure 11**. We gathered statistics to investigate the traffic growth with control traffic only (*maintenance traffic*) and then with both control and user data traffic. The orange plot shows the growth of control/maintenance traffic in the backbone while the brown plot shows the growth

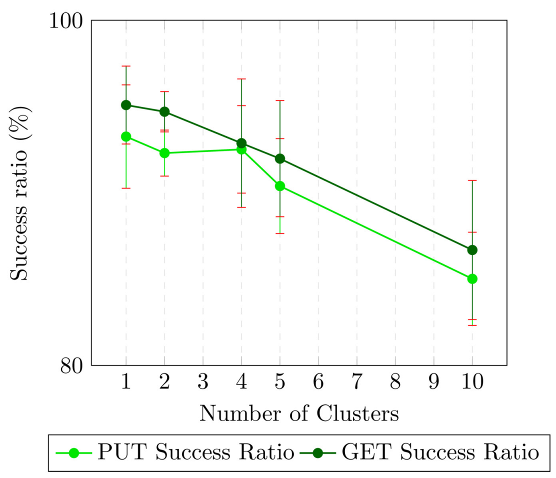

Figure 9. Success ratios with varying number of clusters, static networks.

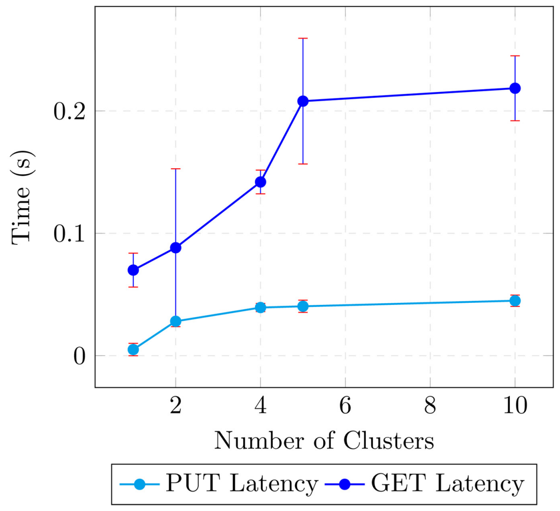

Figure 10. Latencies with varying number of clusters, static networks.

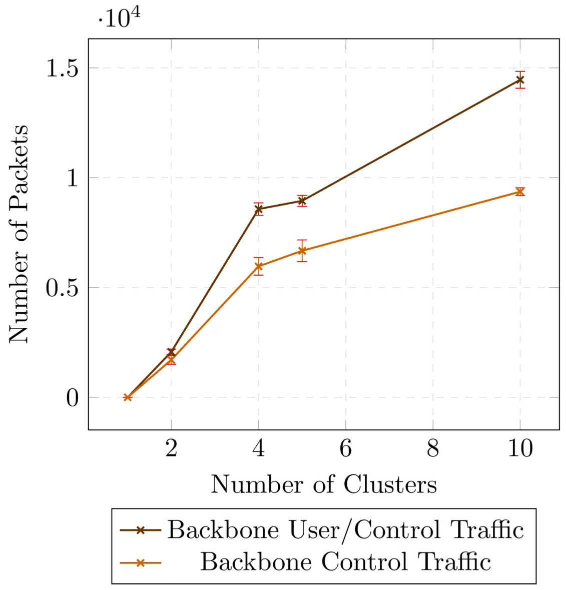

Figure 11. Backbone control and user data traffic, static networks.

with both control and user traffic. The difference between both plots shows the growth of user data traffic in the backbone as the number of clusters increases. The growth in both types of traffic in the backbone is more than linear: as the number of clusters increases, the number of gateways (one per cluster) will increase as well. Each data packet in the backbone will be (re-)broadcast by each gateway, leading to a linear traffic growth. But in addition, a higher and higher fraction of both control and user traffic is being sent through the backbone as the number of peers that are in other clusters increases with the increase in the number of clusters as well. As the backbone has radios that operate with a finite transmission rate (11 Mbps), accessing a shared medium, this growth in traffic will lead to an increase in the number of packet collisions on the backbone and an increase in the packet delay. Since we are using broadcast as the primary means of communication, and unlike in the case of unicast packets in IEEE 802.11, packets that collide will not be retransmitted but will be lost. This would explain the reduction in PUT and GET success ratios observed earlier.

As shown earlier, in a MANET environment, mobility typically has a negative effect on the efficiency of structured protocols like Chord. This issue is related to the method of choosing overlay neighbors and establishing overlay links in these protocols. Performance depends to a large extend on the stability and optimality of the overlay. As the stability of the underlay network can not be guaranteed, failure of even one link in the overlay can cause failure of the whole search process.

For our mobile scenarios, hosts are spread out within the same network area of 2000 m × 2000 m and move about randomly based on the Random Waypoint mobility model parameters described earlier. For a flat network, this cluster area is the entire 2000 m × 2000 m network area, whereas for other scenarios, the total area is divided into smaller portions to form clusters. In each simulation run, nodes start off from new random positions. As all nodes (hosts and gateways) are mobile, there will be topology changes within a cluster, tracked by OLSR, and mobility-related topology changes in the backbone.

As before, the success ratio represents the percentage of queries (PUT or GET) that are eventually delivered to the correct responsible node in the overlay. For both flat and multi-cluster architectures, one challenge with networks that support mobility is the frequent breaking of links which causes high overhead in the underlying network, link and routing table instability [38] [40]. Even though the overlay is oblivious to any changes at the network layer, if the OLSR routing table is unstable during routing convergence, and links are unpredictable or unavailable, packets will be dropped at the MAC layer and the ratio of failed queries will increase. From our simulation results, shown in **Figure 12**, similar to the static network scenarios, there is no statistically significant difference in the success ratio of PUTs compared to GETs (based on the overlap of the 95% confidence intervals). For small numbers of clusters, mobility does result in lower success ratios than the comparable static network cases. The results for 2, 4, and 5 clusters are statistically identical, based on the 95% confidence intervals. The one noticeable outlier is the result for 10 clusters. Here, as we have only one host per cluster, no topology changes occur within a cluster (the single host is always within transmission range of the cluster gateway). The success ratio in this case is not only higher than for other scenarios, but even higher than the static scenarios, with the difference statistically significant. As of yet, we have no real explanation as to why this is the case, and will explore this further as future work.

The latency statistics we obtained from the mobile scenarios, seen in **Figure 13** show an increase in the round trip times for lookup queries, up to the 5 cluster scenario. These latencies are higher than the ones we observed in the static networks. It is unclear whether the case of 10 clusters represents an outlier or is part of a trend and, as mentioned, it warrants further study. We simply note here

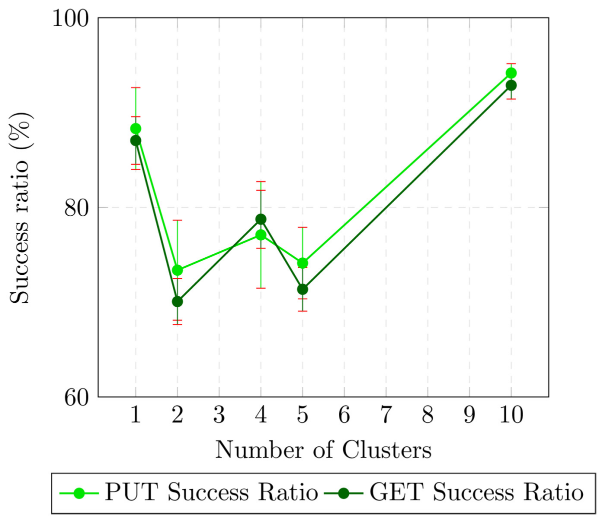

Figure 12. Success ratios with varying number of clusters, mobile networks.

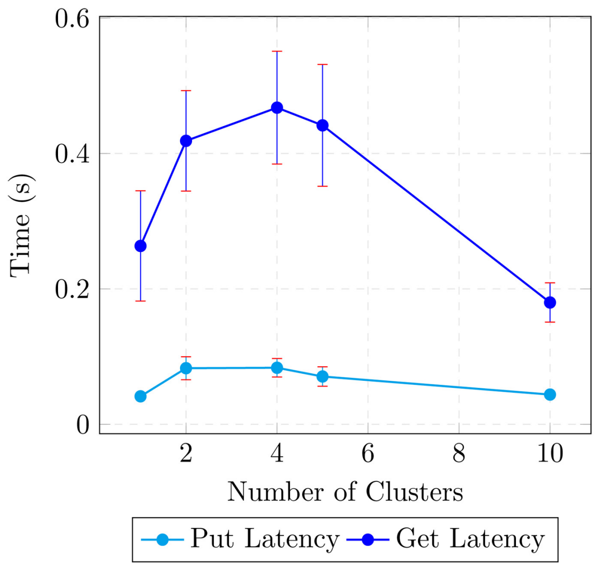

Figure 13. Latencies with varying number of clusters, mobile networks.

that the resulting latency, for the reason already mentioned, is quite comparable to the static case.

Flooding, in general, is quite robust to mobility. The traffic in the backbone (**Figure 14**) is almost identical to the backbone traffic in the static network scenarios (**Figure 11**). In both cases, the amount of traffic is primarily dependent on the Chord maintenance and user data traffic sent through the backbone, as well as the size of the backbone (number of gateways/clusters). The one conclusion from both these figures is that the backbone traffic, growing at a rate faster than linear, will cause the backbone to become a performance bottleneck and limit the scalability of running a flat P2P protocol such as Chord over a hierarchical network architecture.

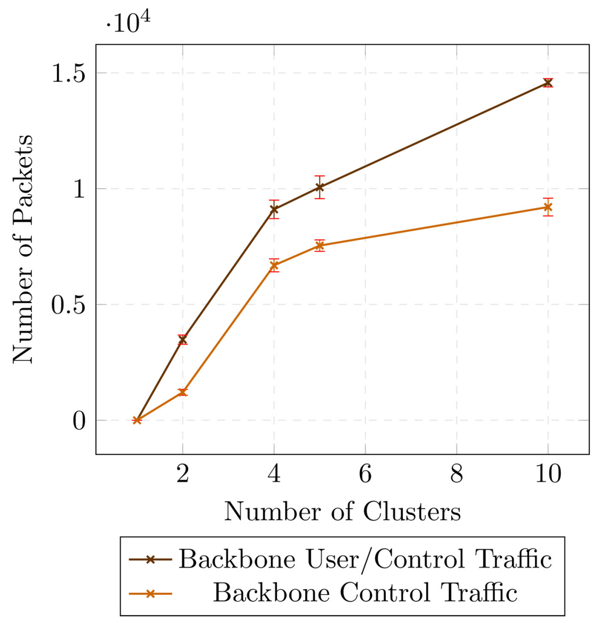

Figure 14. Backbone control and user data traffic, mobile networks.

6.2.3. Discussion

We have presented the quantitative performance of Chord DHT over a hierarchical MANET with a flooding-based protocol in the backbone. We used a version of OLSR which has the OLSR auxiliary extension (*HNA message*) [39] for external network communication according to *RFC* 3626, as the underlay routing protocol in each cluster.

Lookup success ratios in the static scenarios fell as the number of clusters increased due to flooding in the backbone network. In the flat network architecture, where there was no gateway activity in the backbone, we expected to see close to 100% success ratios, the least latency and zero traffic since every operation (Chord maintenance, PUTs and GETs) is confined to within a single cluster and the cluster is well-connected. This was indeed what we observed, except that the PUT and GET success ratios averaged only around 95%. These operations have a time-to-live parameter within which it is expected that they should reach their destination node. As a result of the per-hop count decrement, when these queries expire in the path from source to destination, *i.e.* TTL = 0, those query packets are discarded and a failure message is sent to the originator. The originator may then decide to re-issue the request with a larger TTL value. This was in fact the reason for the majority of queries that failed in the flat static network scenario. In other architectures, in addition to this problem, the growth of packet collisions in the backbone network caused additional failures. This traffic growth also impacted latency, because irrespective of the proximity of two nodes in the overlay, the per hop path length from one node in one cluster to another node outside the cluster will be higher. When studying the latencies separately for queries that reach other nodes within the same cluster and nodes that are in a separate cluster, we did confirm that within-cluster queries have very short RTTs/latencies. But the added latency for queries to nodes in other clusters increased the overall average latency, as shown in **Figure 10** and **Figure 13**.

In the mobile scenarios, success ratios were lower in general, with the exception of the 10 cluster scenario. The mobility of nodes triggered frequent link failures and instability of the underlay routing table. As links became unpredictable or unavailable, packets were dropped at the MAC layer. Furthermore, a few times, we observed that peers "failed" in the overlay as they briefly left and returned. Thus, a lookup issued after a node failure but before stabilization has completed failed either because the node responsible for the key may have failed or some node's finger tables and predecessor pointers may be inconsistent at the time.

In both static and mobile scenarios, the traffic in the backbone increases faster than linearly in the number of clusters, for the reasons explained before. This will cause the backbone to become a performance bottleneck, limiting the overall scalability of our architecture.

7. Conclusions and Future Work

To recall, we are interested in deploying P2P overlays in larger MANETs. To study the feasibility of such an approach, we thoroughly investigated the performance of various P2P overlays. If we limit ourselves to a flat MANET, cross-layer optimization will provide improved performance, lowers overheads, and allows us to run P2P overlays in networks of 10s of nodes. However, all overlays we studied suffer significant performance degradation under mobility. Also, the scalability of such an approach is limited by the inherent limitations on MANET routing in flat architectures.

To further improve scalability, we then studied the performance of Chord as a representative P2P overlay when deployed over a hierarchical MANET. The network is formed by clusters that are interconnected via gateways, with routing in the backbone done by flooding. As the results show, this does indeed limit the routing protocol control message overhead (see **Figure 8**). However, our results show that, as the number of clusters increases, the backbone quickly becomes a performance bottleneck, limiting the gains from such an approach.

In this work, we limited mobility to clusters moving, as well as nodes moving within a cluster, but kept the overall network structure (number of clusters, number of nodes per cluster, etc.) the same.

Going forward, we will explore a number of additional issues. First, we will study why the 10 cluster scenario in the mobile case is such an outlier, outperforming even the static scenarios. Second, we have to add scenarios that will explore the performance of Chord (or other structured P2P overlays) as nodes move between clusters, and clusters join, leave, merge, or split. Finally, we need to improve/reduce the backbone traffic to increase the overall scalability. This could be done in a number of ways. One is to reduce the flooding with a more efficient routing protocol in the backbone, which should limit the growth of the backbone traffic as the number of clusters grows. This routing protocol then needs to be able to track node mobility (which gateway to route a data packet to?). This will allow more clusters to join the network through a shared backbone, but ultimately still limit scalability: as more clusters exist, the probability that an overlay message (for maintenance or a user query) will have to be transmitted through the backbone will increase in flat P2P overlays. Therefore, alternatively, we will study whether we can devise a hierarchical P2P overlay that exploits, as much as possible, communication within a cluster. Even for our rather small network scenarios, we already observed that queries that involve nodes within the same cluster have better performance (lower latency, higher success ratio) than queries between nodes in different clusters. This motivates us to design an overall solution that is highly scalable: at the underlay because we exploit some form of hierarchical routing at the network layer, as well as at the overlay, where we use a hierarchical P2P solution that keeps many requests (or failing that, at least many maintenance messages) within a single cluster. This latter approach may then also be amenable to cross-layer optimizations, which are now confined to a single cluster or subnetwork, potentially improving the performance even further.

Acknowledgements

The research was sponsored by the Army Research Laboratory/US Army RDECOM-Americas and was accomplished under Cooperative Agreement Number W911NF-16-1-0345. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory/US Army RDECOM-Americas or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Abid, S.A., Othman, M. and Shah, N. (2015) A Survey on DHT-Based Routing for Large-Scale Mobile Ad Hoc Networks. *ACM Computing Surveys* (*CSUR*), **47**, 20.

[2] Schollmeier, R. (2001) A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Applications. *Proceedings of the First International Conference on Peer-to Peer Computing*, Linkoping, 27-29 August 2001, 101-102.

[3] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M. and Schooler, E. (2002) Sip: Session Initiation Protocol. RFC 3261.

[4] Jennings, C., Lowekamp, B., Rescorla, E., Baset, S. and Schulzrinne, H. (2014) RE-

source LOcation And Discovery (RELOAD) Base Protocol. Internet Requests for Comments, RFC 6940, January. https://tools.ietf.org/html/rfc6940

[5] Stoica, I., Morris, R., Karger, D., Kaashoek, M. and Balakrishnan, H. (2001) Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications. *ACM SIGCOMM Computer Communication Review*, **31**, 149-160. https://doi.org/10.1145/964723.383071

[6] Ratnasamy, S., Francis, P., Handley, M., Karp, R. and Shenker, S. (2001) A Scalable Content-Addressable Network. *ACM SIGCOMM Computer Communication Review*, **31**, 161-172. https://doi.org/10.1145/964723.383072

[7] Sharma, C. and Kaur, J. (2015) Literature Survey of AODV and DSR Reactive Routing Protocols. *International Journal of Computer Applications*, *International Conference on Advancements in Engineering and Technology*, 14-17. https://pdfs.semanticscholar.org/536c/c8bf91b50ce44f42f3c2d701fc38b80c6e50.pdf

[8] Kaur, H., Sahni, V. and Bala, M. (2013) A Survey of Reactive, Proactive and Hybrid Routing Protocols in MANET: A Review. *Network*, **4**, 498-500.

[9] Quispe, L.E. and Galan, L.M. (2014) Behavior of Ad Hoc Routing Protocols, Analyzed for Emergency and Rescue Scenarios, on a Real Urban Area. *Expert Systems with Applications*, **41**, 2565-2573. https://doi.org/10.1016/j.eswa.2013.10.004

[10] Moussaoui, A. and Boukeream, A. (2015) A Survey of Routing Protocols Based on Link-Stability in Mobile Ad Hoc Networks. *Journal of Network and Computer Applications*, **47**, 1-10. https://doi.org/10.1016/j.jnca.2014.09.007

[11] Ahmad, I., Ashraf, U. and Ghafoor, A. (2016) A Comparative QoS Survey of Mobile Ad Hoc Network Routing Protocols. *Journal of the Chinese Institute of Engineers*, **39**, 585-592. https://doi.org/10.1080/02533839.2016.1146088

[12] O'Driscoll, A., Rea, S. and Pesch, D. (2007) Hierarchical Clustering as an Approach for Supporting P2P SIP Sessions in Ubiquitous Environments. 9*th IFIP International Conference on Mobile Wireless Communications Networks*, Cork, 19-21 September 2007, 76-80. https://doi.org/10.1109/ICMWCN.2007.4668184

[13] Belding-Royer, E.M. (2002) Hierarchical Routing in Ad Hoc Mobile Networks. *Wireless Communications and Mobile Computing*, **2**, 515-532. https://doi.org/10.1002/wcm.74

[14] Vu, Q.H., Lupu, M. and Ooi, B.C. (2009) Peer-to-Peer Computing: Principles and Applications. Springer Science & Business Media, Heidelberg.

[15] Pérez-Miguel, C., Miguel-Alonso, J. and Mendiburu, A. (2013) High Throughput Computing over Peer-to-Peer Networks. *Future Generation Computer Systems*, **29**, 352-360. https://doi.org/10.1016/j.future.2011.08.011

[16] Vu, Q.H., Lupu, M. and Ooi, B.C. (2010) Routing in Peer-to-Peer Networks. In: *Peer-to-Peer Computing*, Springer, Berlin, 39-80.

[17] Stoica, I., Morris, R., Liben-Nowell, D., Karger, D.R., Kaashoek, M.F., Dabek, F. and Balakrishnan, H. (2003) Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications. *IEEE/ACM Transactions on Networking*, **11**, 17-32.

[18] Shen, X.S., Yu, H., Buford, J. and Akon, M. (2010) Handbook of Peer-to-Peer Networking. Vol. 34, Springer Science & Business Media, Berlin. https://doi.org/10.1007/978-0-387-09751-0

[19] Dhara, K., Guo, Y., Kolberg, M. and Wu, X. (2010) Overview of Structured Peer-to-Peer Overlay Algorithms. In: Shen, X.S., Yu, H., Buford, J. and Akon, M., Eds., *Handbook of Peer-to-Peer Networking*, Springer, Berlin, 223-256. https://doi.org/10.1007/978-0-387-09751-0_9

[20] Korzun, D. and Gurtov, A. (2012) Structured Peer-to-Peer Systems: Fundamentals of Hierarchical Organization, Routing, Scaling, and Security. Springer Science & Business Media, Berlin.

[21] Kniesburges, S., Koutsopoulos, A. and Scheideler, C. (2014) Re-Chord: A Self-Stabilizing Chord Overlay Network. *Theory of Computing Systems*, **55**, 591-612. https://doi.org/10.1007/s00224-012-9431-2

[22] Guo, Z., Yang, S. and Yang, H. (2010) P4P Pastry: A Novel P4P-Based Pastry Routing Algorithm in Peer to Peer Network. 2*nd IEEE International Conference on Information Management and Engineering*, Chengdu, 16-18 April 2010, 209-213.

[23] Castro, M.C., Kassler, A.J., Chiasserini, C.-F., Casetti, C. and Korpeoglu, I. (2010) Peer-to-Peer Overlay in Mobile Ad-Hoc Networks. In: Shen, X.S., Yu, H., Buford, J. and Akon, M., Eds., *Handbook of Peer-to-Peer Networking*, Springer, Berlin, 1045-1080. https://doi.org/10.1007/978-0-387-09751-0_37

[24] Lee, S., Quan, L., Lee, K., Cho, T. and Jang, J. (2004) A Peer-to-Peer Search Scheme over Mobile Ad Hoc Networks.

[25] Delmastro, F. (2005) From Pastry to CrossROAD: CROSS-Layer Ring Overlay for Ad Hoc Networks. 3*rd IEEE International Conference on Pervasive Computing and Communications Workshops*, Kauai Island, 8-12 March 2005, 60-64. https://doi.org/10.1109/PERCOMW.2005.38

[26] Pucha, H., Das, S. and Hu, Y. (2004) Ekta: An Efficient DHT Substrate for Distributed Applications in Mobile Ad Hoc Networks. 6*th IEEE Workshop on Mobile Computing Systems and Applications*, Windermere, 3 December 2004, 163-173. https://doi.org/10.1109/MCSA.2004.11

[27] Rhea, S., Godfrey, B., Karp, B., Kubiatowicz, J., Ratnasamy, S., Shenker, S., Stoica, I. and Yu, H. (2005) OpenDHT: A Public DHT Service and Its Uses. *ACM SIGCOMM Computer Communication Review*, **35**, 73-84.

[28] Ding, G. and Bhargava, B. (2004) Peer-to-Peer File-Sharing over Mobile Ad Hoc Networks. *Proceedings of the* 2*nd IEEE Annual Conference on Pervasive Computing and Communications Workshops*, Orlando, 14-17 March 2004, 104-108. https://doi.org/10.1109/PERCOMW.2004.1276914

[29] Yan, L. (2005) Can P2P Benefit from MANET? Performance Evaluation from Users Perspective. In: *International Conference on Mobile Ad-Hoc and Sensor Networks*, Springer, Berlin, 1026-1035. https://doi.org/10.1007/11599463_99

[30] Caleffi, M. and Paura, L. (2009) P2P over MANET: Indirect Tree-Based Routing. *IEEE International Conference on Pervasive Computing and Communications*, Galveston, 9-13 March 2009, 1-5.

[31] Krishna, P.P.M., Subramanyam, M. and Prasad, K.S. (2015) Investigation of Chord Protocol in Peer to Peer-Wireless Mesh Network with Mobility. *International Journal of Electronics and Communication Engineering*, **9**, 934-938.

[32] Castro, M.C., Villanueva, E., Ruiz, I., Sargento, S. and Kassler, A.J. (2008) Performance Evaluation of Structured P2P over Wireless Multi-Hop Networks. 2*nd International Conference on Sensor Technologies and Applications*, Cap Esterel, 25-31 August 2008, 796-801.

[33] Al Mojamed, M. and Kolberg, M. (2016) Structured Peer-to-Peer Overlay Deployment on MANET: A Survey. *Computer Networks*, **96**, 29-47. https://doi.org/10.1016/j.comnet.2015.12.007

[34] Zahn, T. and Schiller, J. (2006) DHT-Based Unicast for Mobile ad Hoc Networks. 4*th Annual IEEE International Conference on Pervasive Computing and Communications Workshops*, Pisa, 13-17 March 2006, 5.

https://doi.org/10.1109/PERCOMW.2006.42

[35] Jennings, C., *et al*. (2014) Rfc 6940: Resource Location and Discovery (Reload) Base Protocol. https://tools.ietf.org/html/rfc6940

[36] Leong, B., Liskov, B. and Demaine, E.D. (2004) EpiChord: Parallelizing the Chord Lookup Algorithm with Reactive Routing State Management. 12*th IEEE International Conference on Networks*, Singapore, 19 November 2004, 270-276. https://doi.org/10.1109/ICON.2004.1409145

[37] Mojamed, M.A. (2016) A One Hop Overlay System for Mobile ad Hoc Networks. Ph.D. Dissertation, University of Stirling, Stirling.

[38] Jacquet, P., Muhlethaler, P., Clausen, T., Laouiti, A., Qayyum, A. and Viennot, L. (2001) Optimized Link State Routing Protocol for Ad Hoc Networks. *Proceedings. IEEE International Multi Topic Conference*, Lahore, 30 December 2001, 62-68.

[39] Clausen, T. and Jacquet, P. (2003) RFC 3626. Optimized Link State Routing Protocol (OLSR).

[40] Wongsaardsakul, T. and Kanchanasut, K. (2007) A Structured Mesh Overlay Network for P2P Applications on Mobile Ad Hoc Networks. In: *International Conference on Distributed Computing and Internet Technology*, Springer, Berlin, 67-72. https://doi.org/10.1007/978-3-540-77115-9_6