Scientific Research Publishing

Communications and Network, 2020, 12, 99-121
https://www.scirp.org/journal/cn
ISSN Online: 1947-3826
ISSN Print: 1949-2421

A P2P Approach to Routing in Hierarchical MANETs

Thomas Kunz, Silas Echegini, Babak Esfandiari

Systems and Computer Engineering, Carleton University, Ottawa, Canada
Email: tkunz@sce.carleton.ca, SilasEchegini@cmail.carleton.ca, babak@sce.carleton.ca

How to cite this paper: Kunz, T., Echegini, S. and Esfandiari, B. (2020) A P2P Approach to Routing in Hierarchical MANETs. *Communications and Network*, **12**, 99-121.
https://doi.org/10.4236/cn.2020.123006

Received: June 30, 2020
Accepted: August 14, 2020
Published: August 17, 2020

Abstract

We present an effective routing solution for the backbone of hierarchical MANETs. Our solution leverages the storage and retrieval mechanisms of a Distributed Hash Table (DHT) common to many (structured) P2P overlays. The DHT provides routing information in a decentralized fashion, while supporting different forms of node and network mobility. We split a flat network into clusters, each having a gateway who participates in a DHT overlay. These gateways interconnect the clusters in a backbone network. Two routing approaches for the backbone are explored: flooding and a new solution exploiting the storage and retrieval capabilities of a P2P overlay based on a DHT. We implement both approaches in a network simulator and thoroughly evaluate the performance of the proposed scheme using a range of static and mobile scenarios. We also compare our solution against flooding. The simulation results show that our solution, even in the presence of mobility, achieved well above 90% success rates and maintained very low and constant round trip times, unlike the flooding approach. In fact, the performance of the proposed inter-cluster routing solution, in many cases, is comparable to the performance of the intra-cluster routing case. The advantage of our proposed approach compared to flooding increases as the number of clusters increases, demonstrating the superior scalability of our proposed approach.

Keywords

MANET, Routing, Hierarchical Networks, DHT, P2P, Chord, OLSR, OMNeT++

1. Introduction

Mobile Ad hoc Networks (MANETs) are increasingly gaining popularity and finding applications in a range of areas, including emergency response networks, intelligent transportation systems, outdoor enterprises, small businesses, etc.

[1] [2] [3]. One important characteristic is that they are self-organizing and self-configuring wireless multi-hop networks which do not rely on any existing infrastructure; as nodes are by themselves, servers and clients [4] [5]. Each node must act as a router to forward traffic unrelated to its own use.

The number of users in MANET applications may vary from just a handful to hundreds of thousands of people and more [6]. As MANETs and mobile devices become increasingly popular and the ensuing networks grow larger, more research effort focuses on devising protocols for route establishment and maintenance in these networks. In a flat network of several interconnected mobile devices, and spanning a large geographical area, for instance, the network will typically incur increasing overheads for route maintenance and establishment and other network functions. This scalability limitation is true for almost any MANET routing protocol proposed for flat networks. Most routing protocols for ad-hoc networks are either proactive (table-driven) 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 routing protocol approaches scale poorly [7] [8]. On-demand routing protocols are limited by their route discovery techniques, extensively using flooding. Hop-by-hop flooding usually has a negative impact on network performance and often leads to large delays in route discovery [2] [9]. Proactive routing protocols have these routes readily available, but it comes at a cost of constant route discovery throughout the lifetime of the network. The scalability challenge gets even worse in the case that nodes are mobile and links become generally unpredictable [2] [9].

A hierarchical routing architecture, when carefully planned, simplifies routing tables considerably and lowers the amount of routing information exchanged [3] [10], thus increasing search efficiency and increasing scalability. This is best exemplified by the global Internet, which employs a hierarchical architecture and routing structure. The Internet is divided into routing domains. A routing domain typically contains a collection of co-located networks connected by routers (who are nodes) and linked in a common routing domain called the backbone [3].

In this paper, we deploy hierarchical routing to tackle the scalability and performance issues mentioned above. More specifically, the proposed backbone routing solution exploits the capabilities of a P2P overlay. P2P applications have gained increased popularity as they allow the exchange of information and resources among network users without the need for a dedicated server. In P2P applications, nodes are connected through a logical overlay network and respond to queries or requests from other nodes to assign or utilize a variety of resources (data streaming, file sharing, etc.) [11] [12]. P2P applications can be implemented in either unstructured (e.g. Gnutella [13]) or structured topologies.

In structured P2P networks, the topology is controlled to render search and queries between peers more efficient. Structured P2P networks mostly utilize

Distributed Hash Tables (DHT) as a substrate [14] to implement the lookup services in the network. By using a DHT, a key-value pair is assigned to each node and resource in the overlay network [12]. Also, P2P applications, by their very design, do not require any dedicated infrastructure (such as well-known servers), and therefore fit well with the MANET vision of nodes joining and leaving, running the network in a self-organizing and self-healing manner. We therefore developed a routing scheme that exploits the services offered by a P2P overlay for efficiently routing data packets in the underlay between clusters.

The main contribution of our work is to propose and evaluate a scalable routing solution for hierarchical MANETs. As discussed above, we believe that P2P applications are a good fit with the server-less, self-organizing nature of MANETs. We build a backbone routing solution around a structured P2P overlay, using the key abstraction of a Distributed Hash Table (DHT). The proposed solution enables us to route IP data packets end-to-end, both within and across clusters. As the results demonstrate, the inter-cluster routing performance is almost as good as the intra-cluster performance. Ultimately, we envision a scenario where we exploit a deployed P2P MANET application such as P2PSIP to provide routing services at low costs: many of the costs associated with building and maintaining the DHT overlay would be incurred on behalf of the application(s), not the routing protocol.

The remainder of this paper is organized as follows. The next section reviews some relevant related work. Section 3 outlines our proposed routing solution, Section 4 discusses the performance of our proposed solution using OMNeT++ simulations, comparing our scheme with the simple flooding through the backbone scheme outlined earlier. Finally, Section 5 summarizes our results and discusses possible future work.

2. Related Work

As discussed earlier, treating the whole MANET as a single, flat routing domain runs into scalability issues. An alternative to flat MANETs is clustering or hierarchical routing. Such protocols group nodes into clusters of nodes. Mobility within a single cluster is tracked only by other nodes in the same cluster. Members of other clusters only need to know how to reach these clusters, typically via a gateway node. These gateways nodes form a routing backbone, enabling end-to-end connectivity.

In the Internet, BGP is the typical inter-domain routing protocol which interconnects networks [15]. BGP works well with managing autonomous systems which are fixed networks governed by a single entity. It was designed to cope with the scale and operational challenges of the Internet. Compared to the Internet, MANETs are considerably smaller yet have highly dynamic environments. In MANETs, nodes are generally mobile and when an ad-hoc network supports sub-networks, these sub-networks may be mobile as well. As a result, a range of challenges like single networks splitting, multiple networks merging, or

nodes moving between networks may ensue. When this becomes the case, either new inter-domain routing protocols are required, or the dominant BGP needs to be modified, which can lead to poor performance [15].

There exists a wide range of clustering and (hierarchical) routing protocols specifically designed for MANETs, some of it summarized in a number of survey papers [16] [17]. One of the earliest clustering protocols is LCA [18], developed for packet radio networks. LCA organizes the nodes into clusters according to the proximity of the nodes. Each cluster has a cluster head and all nodes in a cluster are in the direct transmission range of the cluster head. The choice of the cluster head is based on node identifiers, where the node with the largest identifier in a given area becomes the cluster header. The gateways in the overlapping region between clusters are used to connect clusters. LCA specifies that there should only be one designated gateway to interconnect clusters at a given time. A pair of nodes within transmission range of each other can also be used to connect clusters if there are no nodes in the overlap region.

In CGSR [19], packets are routed alternately between the cluster leaders and the gateways. The authors define several extensions that can be added to CGSR, such as priority token scheduling and gateway code programming, to control access to the channel. In addition, they define a LCC (Least Cluster Change) algorithm, designed to reduce the number of changes in the cluster leader, since such changes can generate significant overhead.

[20] [21] take a different approach to clustering and present two clustering algorithms. The first of these is intended for "quasi-static" networks in which nodes are slow moving, if moving at all. The other algorithm is designed for higher mobility. Both algorithms assign different weights to nodes with the assumption that each node is aware of its respective weight. The weights are in turn used to determine the cluster leaders. If two cluster leaders come into contact, the one with the smaller weight must revoke its leader status.

CEDAR [22] builds a set of nodes (*i.e.*, a core) to perform route computation instead of creating a cluster topology. Using the local state information, a minimum dominating set of the network is approximated to form the core. CEDAR establishes QoS routes that satisfy bandwidth requirements using the directionality of the core path. Link state and bandwidth availability is exchanged to maintain important information for computing QoS routes.

Kleinrock and Kamoun investigated a hierarchical routing scheme with the goal of reducing routing table size [23], approach also adopted by [24]. The authors determined that the length of the routing table is a strict function of the clustering structure. Clustering generally has the unwanted side-effect of an increase in path length, and so the goal was to find an optimal clustering scheme that optimizes path length. It was determined that the number of entries in a node's forwarding table is minimized when the number of level-i clusters in each level-(i + 1) cluster is e, and the number of levels in the clustering hierarchy equals $\ln N$. In this case, the forwarding table contains $e\ln N$ entries.

The Landmark Routing technique [25] is a distinct approach to building a hierarchy as it is based on landmarks, as opposed to transmission ranges. A landmark is a router whose location is known by its neighboring routers up to some radius. All routers within that radius know how to reach the landmark. A hierarchy of such landmarks is built by increasing the radius of some of the routers. Nodes have hierarchical addresses based on the landmarks with which they are associated. A source node routes to a destination by sending the packet to the lowest level landmark with which both nodes are associated. As the packet approaches the destination, the granularity of routing knowledge about that destination improves, and so the packet can be accurately routed to the destination.

Our proposed solution differs in two respects from the state-of-the-art. First, we assume that clusters are determined by the application requirements. For example, we may have different platoons of soldiers (in a military context) or different first responder crews joining in as a group. We are therefore not exploring specific clustering schemes. Second and more importantly, none of the existing protocols uses a P2P overlay. There is a significant body of work on overlay routing for structured P2P overlays, and how to adapt such overlays to dynamic MANETs, summarized for example in [6]. However, these protocols only support operations at the DHT or overlay level, and are not used for routing IP data packets. As discussed in the introduction, we believe that P2P applications are a good fit with the server-less, self-organizing nature of MANETs. The proposed solution enables us to route IP data packets end-to-end, both within and across clusters using the services provided by a structured P2P overlay. Ultimately, we envision a scenario where we exploit a deployed P2P MANET application such as P2PSIP [11] to provide routing services at low costs: many of the costs associated with building and maintaining the DHT overlay would be incurred on behalf of the application(s), not the routing protocol.

3. DHT-Based Backbone Routing Protocol

3.1. General Design

In a MANET, nodes are often mobile and there may even exist mobile subnetworks which are interconnected through a backbone [7] [26]. In flat MANETs, routing is challenging in the presence of mobility because links become unpredictable as a result of the dynamic nature of these networks. This becomes even more challenging in cases where there are mobile sub-MANETs. In such setups, the routing scheme deployed in the backbone network must be able to maximize the backbone bandwidth, enhancing throughput and reducing end-to-end delays with respect to schemes without a backbone. It must do so without compromising (in fact, possibly enhancing) scalability and fault tolerance.

Distributed Hash Tables (DHTs) are a form of a distributed database that can store and retrieve information associated with a key, in a network of peers/nodes that can join and leave the network at any time. In this paper, we leverage this distributed data sharing functionality to provide extra information with which

each gateway can carry out routing through the backbone efficiently. By leveraging this functionality, different forms of mobility, including single nodes switching cluster membership and whole clusters merging/splitting, can be managed. The DHT strategy is used as a lookup strategy in structured P2P systems to specify the location of objects in peers, and it provides all the needed services to look up objects in a decentralized P2P system. The DHT retains mapped information about nodes and peers in the form of key/value pairs (k, v) so that data can be easily located in the overlay network. In DHTs, each peer maintains a storage space to keep a hash table. Many structured P2P systems like Chord [27] or OneHopOverlay4MANET [12] are based on a DHT.

We divide a flat MANET of N nodes comprising h nodes and g gateways into c clusters. Each cluster is therefore a MANET on its own and has (at least) one gateway node. The gateway nodes are connected in a backbone MANET and connect their local clusters to the rest of the network. Each cluster is basically a separate and independent routing domain, running its own intra-domain routing protocol. Similarly, the backbone, which is its own a MANET, may require a routing protocol to coordinate routing among the gateway nodes.

Within a cluster, we selected OLSR [28] as our routing protocol. While this addresses intra-cluster connectivity, routing between nodes in different clusters is more complex. For example, a node A_1 in cluster A needs to route a packet to a node B_1 in cluster B, as shown in **Figure 1**. The source node A_1 creates an IP data packet using the address of the destination node B_1. A_1 does not know how to forward the packet to B_1 because its routing table does not have an entry for B_1. The OLSR protocol supports an optional auxiliary functionality, "Host and Network Association (HNA) Message", whereby a gateway announces reachability of other nodes or subnets to its local cluster members. The gateways in each cluster (here GW_A for cluster A and GW_B for cluster B) will periodically broadcast HNA messages to all nodes in their cluster to advertise reachability of nodes outside the cluster through them (Step 1 in **Figure 1**). We advertise a global default route (0.0.0.0/0.0.0.0). Each node will create an entry in its routing table that IP forwarding defaults to when looking up a route. Packets destined for nodes outside the local cluster will consequently be forwarded to the appropriate cluster gateway. In case a cluster has multiple gateways, each advertising reachability of the outside world, the closest gateway (in terms of hop count) will be used.

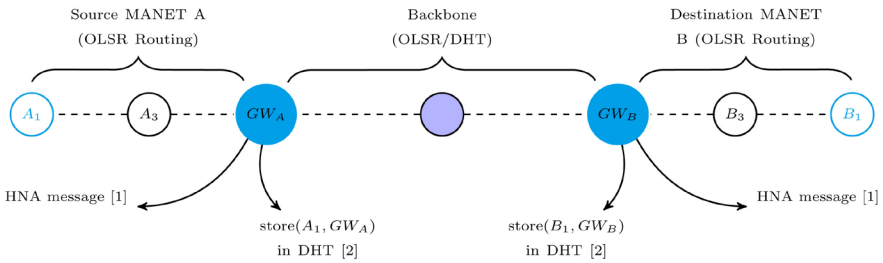

Figure 1. Initial routing setup.

Once a cluster gateway received a data packet destined to a node outside its cluster, it will forward it over the backbone. This requires a routing solution/protocol, simply running another instance of OLSR on the nodes forming the backbone will not suffice. Consider the example in Figure 1: once GW_A receives a packet destined for B_1, it needs a way to forward it to the appropriate gateway, here GW_B. We developed two solutions for this problem: **Flooding** and **DHT-based Routing**.

Flooding: Flooding is a simple approach for the MANET backbone. A source node in the backbone will broadcast a packet in the backbone once, and all other nodes, except the destination gateway, will rebroadcast these packets. The gateway that knows (from its local routing table) that the destination node is a member of its local cluster will forward the data packet into its local cluster, using the intra-cluster routing protocol. In this case, the backbone has no need of a specific (unicast) routing protocol. We implemented a flooding protocol to use as a base case to evaluate the performance of our solution.

DHT-Based Unicast Routing: Using a unicast protocol like OLSR in the backbone will carry out routing functions among the members of the backbone network (the gateways nodes). Each gateway GW_X participates in a P2P overlay to advertise reachability to all nodes X in its local cluster by storing a key-value pair (X, GW_X) as follows. Gateways learn about all nodes in their cluster through OLSR, which will populate a local routing table. Using this information, each gateway node GW advertises reachability to all nodes N in its cluster by storing the following key-value pair in the DHT: (k, GW) (Step 2 in Figure 1). Key k is generated by hashing each node's IP address.

3.2. Routing in Static Scenarios

Figure 2 shows the data packet forwarding, assuming that the network is static. Once A_1 has a data packet to transmit, it will consult its routing table. Using the default route created by the HNA messages, it routes the packet to its gateway GW_A (Step 3 in Figure 2). Once the gateway receives such packets, it executes *SHA*-1 on the IP address of B_1 to generate key k. It then queries the DHT using this key. This retrieves the associated gateway, GW_B, for B_1's cluster (Step 4 in Figure 2). GW_A forwards the packet to GW_B through the backbone (Step 5 in Figure 2) using tunneling (using IP-in-IP encapsulation). Once GW_B receives the packet, it decapsulates the datagram and forwards the packet within its cluster based on the local instance of the intra-domain routing protocol (Step 6 in Figure 2).

This approach is similar to on-demand routing protocols such as AODV or DSR: routes to new destinations are only discovered when the first data packet is destined to such a node. However, our DHT lookups are done via unicast routing in the P2P overlay, rather than broadcasting/flooding the backbone. Furthermore, unlike such protocols, any topology changes in the backbone will be hidden from the gateways (*i.e.*, no route maintenance), reducing the need to rediscover routes. Connectivity among gateways is maintained by the OLSR instance executing in the backbone.

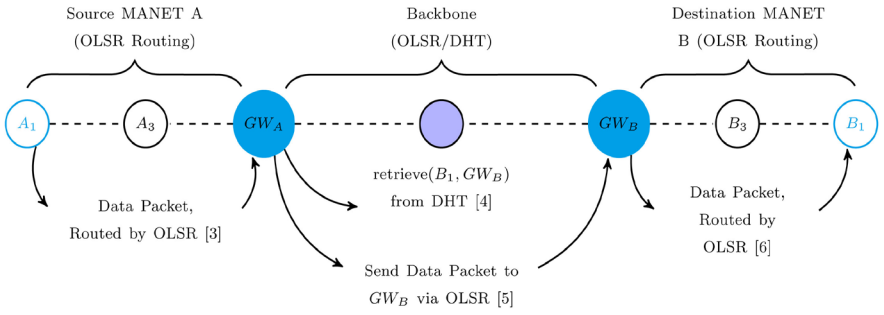

Figure 2. Packet forwarding.

3.3. Routing in the Presence of Mobility

We expanded this basic scheme to include routing support for various mobility scenarios which are likely to result in MANET environments: mobility within the local cluster, whole clusters joining or leaving, clusters merging or splitting, and nodes switching cluster membership or completely disappearing (can no longer be reached). Nodes moving within their local cluster are simply a variation of the static scenario. The challenging mobility scenarios are when clusters merge/split or a node changes cluster membership. All nodes use their (single) radio to communicate on a fixed frequency. Once a node joins a different cluster, it will exchange OLSR control messages with other nodes, effectively joining this cluster. Upon leaving a cluster, the node will stop this control message exchange, allowing other nodes to detect its absence. Similarly, when groups of nodes come into radio contact with each other, they will form in essence one MANET, exchanging routing information with other nodes, forming a single (merged) cluster with multiple gateways. Once these groups separate physically, they will also stop forming a single cluster, splitting into two or more disjoint clusters with their own gateway(s).

In the basic routing case we described earlier, there was no need to update the DHT with any information because the node-gateway mapping stored in the DHT stayed the same for the entire network lifetime. With mobility, it becomes necessary for the gateways to update the DHT with recent routing information regarding the nodes they can now reach or no longer reach. For example, when two clusters (A and B) merge, the ensuing cluster will have two gateways GW_A and GW_B who will now both advertise reachability of all cluster nodes in the DHT. Similarly, when the clusters split, the gateways will have to delete entries they initially advertised in the DHT, for nodes they can no longer reach. In the case of nodes switching cluster membership, a gateway GW_X to the cluster X which the node n arrived in will learn about the presence of n in its cluster via the intra-domain routing protocol, OLSR, and will advertise to the DHT that it can now reach node n. Similarly, the gateway GW_Y to cluster Y which node n left will have to update the DHT that it can no longer reach node n. Temporarily, depending on the order in which these updates occur, a node may be advertised as being reachable through no gateway, one gateway, or both gateways.

To support these scenarios, we store more than one gateway IP address under the key representing a destination node. We serialize a list of potential gateways and store this as the value *v* in the *PUT* message. This way, a source gateway that retrieves this information can use the hop count information in its routing table to select the closest gateway. While selecting the closest gateway through the backbone does not necessarily translate to the shortest path to a destination node, it serves to minimize the traffic through the backbone. Figures 3-5 show the way the DHT is updated to support the different mobility scenarios. In each of these figures, hosts A_1, A_2 and gateway GW_A are members of cluster A while hosts B_0, B_1, B_2 and gateway GW_B are members of cluster B. B_2 is a host previously in cluster B that recently moved to cluster A. The arrows represent DHT queries and responses while the dotted lines indicate wireless connections (potentially multi-hop). In order to provide a solution that supports all our mobility scenarios, every gateway queries the DHT before adding an entry to it.

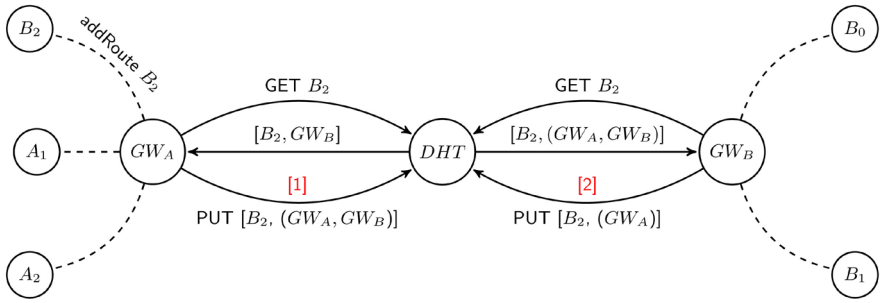

Figure 3. DHT updates [A].

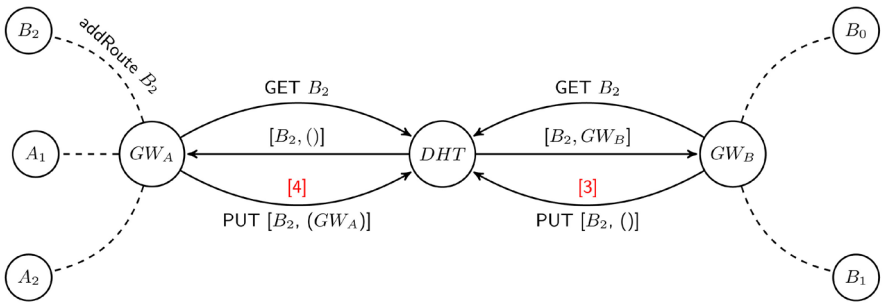

Figure 4. DHT updates [B].

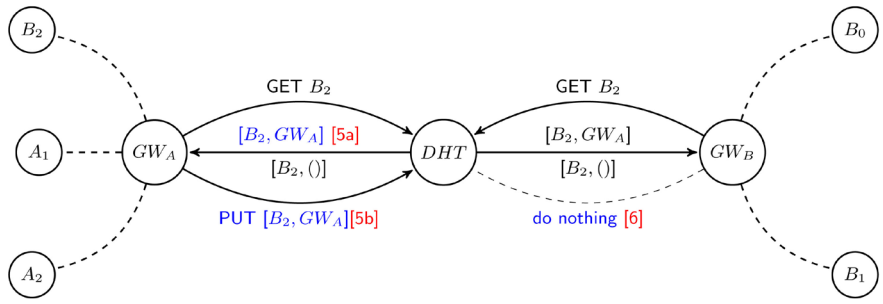

Figure 5. Addressing possible race condition.

Queries are distinguished from another via different code names: ADD_ROUTE, DELETE_ROUTE, LOCAL_UPDATES. Gateways will handle these queries differently. The three mobility cases together with the steps involved in updating the DHT in each case are described below.

1) ADD_ROUTE

Initially, every gateway advertises reachability to all hosts in its cluster in the DHT as explained earlier. Subsequently, in events like clusters merging or hosts joining, the gateways will learn about these new hosts after their routing tables have been updated by the intra-domain routing protocol, OLSR. The gateway then hashes the individual IP addresses of the new hosts and queries the DHT. This query will either be successful and return a valid value, a NULL value, or completely fail. According to the result of a query for a host B_2, a gateway GW_A will do the following:

a) Retrieving a valid value from the DHT implies that an entry for B_2 already exists in the DHT. If the associated list of gateways already contains GW_A, this entry need not be updated (Step [5a] in **Figure 5**). Otherwise, GW_A will add its IP address to the existing entry and update the DHT (Step [1] in **Figure 3**).

b) Returning a NULL value implies that no entry exists for B_2 in the DHT. GW_A will then hash the IP address of B_2 to generate key k and initiate a PUT message to advertise its reachability to N in the DHT with the message: $PUT(k, GW_A)$ (Step [4] in **Figure 4**).

c) When a query fails (neither returns a valid value nor a NULL value), GW_A chooses a random back-off interval and re-issues the query to the DHT.

2) DELETE_ROUTE

When a host B_2 leaves a cluster, or clusters split, gateways learn that one or multiple hosts left after their routing tables have been updated by the intra-cluster routing protocol. The gateways will hash the IP address of a host B_2 to generate key k and query the DHT, which will return the value currently stored against key k (Step [2] in **Figure 3**, [3] in **Figure 4**, or [6] in **Figure 5**). If the value retrieved contains GW_A's IP address, it removes its IP address from the list and updates the DHT (Step [2] in **Figure 3**). Otherwise, for a single entry corresponding to its IP address, it removes the entry, and updates the DHT with a NULL value (Step [3] in **Figure 4**).

It is difficult to synchronize the DHT updates between GW_A and GW_B for host B_2. Consequently, race conditions may occur. For example, consider **Figure 4** and assume that both gateways queried the DHT concurrently. GW_A updates the entry by adding itself to the list of gateways reaching B_2. This information will be lost when GW_B overrides the entry with NULL (believing itself to have been the only gateway registered for the node. To handle this, each gateway will periodically check with the DHT to make sure that its update was successful (Steps [5] and [6] in **Figure 5**).

3) LOCAL_UPDATES

Each gateway maintains a separate data structure, *TEMP_ROUTING_TABLE*, which caches all the routing information for the different destinations X, Y, …,

Z retrieved from the DHT. With mobility, this routing information becomes stale. To check that the information is still valid, the gateways query the DHT periodically to refresh this information. If this information is not used, the routing cache expires (soft state).

4. Performance Evaluation

We used OMNeT++ as our simulation platform [29]. The INET Framework, an open-source library for the OMNeT++ environment [30], provides implementations of MANET routing protocols such as AODV or OLSR, as well as an IEEE 802.11 MAC layer. OverSim [31] is a flexible framework for overlay network simulation and includes implementations of some structured P2P protocols (*i.e.* Chord [27], ... etc.). We added an implementation of basic flooding through the backbone and implemented the solution outlined in the previous section. The details of this implementation are available in [32]. We selected Chord as the overlay DHT protocol in the results reported here.

We explored two different scenarios, a stationary scenario and one with node mobility. For the stationary scenario, we varied the number of clusters while keeping the overall number of nodes constant. For the mobile scenario, we fixed the number of clusters at 4, varying mobility-related parameters. Application data is generated via periodic ping packets. Some of these pings are exchanged between nodes in the same cluster (referred to as "Local Cluster" later on), measuring the intra-cluster routing performance. In the case of a single cluster, all ping messages are exchanged among nodes in the same cluster, effectively forming a flat MANET. Other ping exchanges involve nodes in two different clusters and therefore have to be routed via the backbone. This then allows us to evaluate the inter-cluster routing performance. The performance metrics we collected are:

1) Ping Success Rate (PSR): The ratio of the ping echo responses received to the ping echo requests generated by the sources.

2) Ping Round Trip Time (PRTT): Time elapsed between sending the echo request and receiving the echo response.

3) Routing Overhead: The average of the total OLSR traffic (HELLO, Topology Control (TC), Host and Network Association (HNA) messages) sent by nodes in the local clusters, and the average of the OLSR and DHT maintenance traffic sent by the gateway nodes in the backbone.

4.1. Static Scenarios

In the static scenario, all nodes are uniformly distributed in their different cluster areas, and remain members of the same cluster for the entire network lifetime. **Table 1** summarizes our key simulation parameters. Initially, all 40 nodes participate in a flat network where all nodes are hosts who participate to support routing in the network and 30 of the 40 nodes exchange ping messages with each other. The 40 hosts are then divided into 2, 4, 5, 6, 8 and 10 clusters having 2, 4,

5, 6, 8 and 10 gateways. The clusters are MANETs and the gateways are part of a backbone MANET through which they inter-connect all the clusters. Gateways are equipped with two radios with which they can communicate over two separate interfaces: the cluster they belong to, and the backbone network. The hosts, on the other hand, only have a single radio for communication within the cluster they belong to. The resulting network is a two-tier hierarchical network structure, with the gateways serving to provide connectivity between members of their cluster and external networks. In the 2, 4, 5, 6, and 8 cluster configurations, 8, 6, 5, 4, and 2 gateways respectively, do not participate in the backbone network but exist to support routing in the clusters they belong to. Thus, they behave like hosts but do not send nor receive any ping messages. In the flat network configuration (one cluster), all hosts are uniformly distributed in the network area and they assume this position throughout the entire simulation duration. In the other cluster configurations, each cluster is assigned a unique area in the network area and the member nodes are also uniformly distributed within their cluster space. Similarly, nodes in their individual clusters assume the same initial position for the entire simulation duration. Simulations were run for 900 simulated seconds. The first 210 seconds was allowed for the routing tables of each node to be populated and for the gateway nodes to build the overlay network and reach stability. The results reported here are the averages over 10 repetitions for each scenario, using different random number seeds each time. Graphs for each metric also show the margin of error by plotting the 95% confidence intervals. At times, these CIs are quite narrow and therefore not easily visible.

Table 1. Simulation parameters, static scenarios.

Parameter	Value
Mobility Model	Stationary, nodes randomly placed
Network area	4000 M × 4000 M
Network architecture	flat and hierarchical
Network scenarios	1, 2, 4, 5, 6, 8 & 10 clusters
Traffic Type	ICMP (ping) traffic
No of nodes	40 (10 gateways, 30 nodes)
No of ping sources	30 nodes
Local-Cluster Ping sources	All nodes in the flat network
	30% of the nodes otherwise
Ping rate	$N(0.3, 0.01)$, ~100 pings/sec
No. simulations	10 runs per scenario
Simulation time	900 seconds
Stabilization time	210 seconds
Measurement time	690 seconds
MAC protocol	IEEE 802.11 g, 11 Mbps

We explored 4 different ping packet sizes, but the results and trends are very similar. So for brevity sake, we report on only one set of results, using a ping packet size of 256 bytes. In all figures, the (F) suffix in the legend refers to the metric value when using Flooding as routing solution in the backbone, similarly (D) refers to the metric value when using the proposed DHT-based approach.

With flooding, as the number of clusters increases, PSR deteriorates sharply (Figure 6(a)). We also observed that the number of packets transmitted in the backbone, as well as packet collisions and packet drops, increase drastically with the number of clusters. The backbone handles more traffic because of the redundant rebroadcasts of each data packet. With multiple gateways re-transmitting a just received packet, there is a high probability that these successive transmissions result in collisions, even when adding random jitter. Overall, as the number of clusters increases, for each ping packet size and across the different ping packet sizes, the frequent rebroadcasts utilize most of the backbone channel resources. Queues start to build up and packets are being dropped, resulting in fewer ping packet exchanges being successful.

For the proposed DHT-based solution, as shown in Figure 6(a), PSR is consistently high for pings that cross the backbone (*i.e.*, sender and receiver are in different clusters), as shown by the blue line. In fact, the success ratio is almost as high as for pings among nodes in the same cluster (the green line). The backbone traffic does grow with the number of clusters, but at a slower rate than was the case for flooding. We also experience almost no collisions and dropped packets. With the DHT-based approach, ping packets are transmitted through a unicast tunnel, so the MAC protocol will retry packet transmissions a certain number of times, ensuring the high PSR across the backbone. Overall, the protocol delivered over 95% of the packets to their destinations, which remains constant for all numbers of clusters and ping packet sizes.

Figure 6(b) shows PRTT as we increase the number of clusters. Both approaches have similar values for the one- and two-cluster scenarios but begin to vary significantly beginning with the four-cluster scenario. The one-cluster scenario has no backbone through which data needs to be routed and is expected to produce similar results. The two-cluster scenario involves only two gateways.

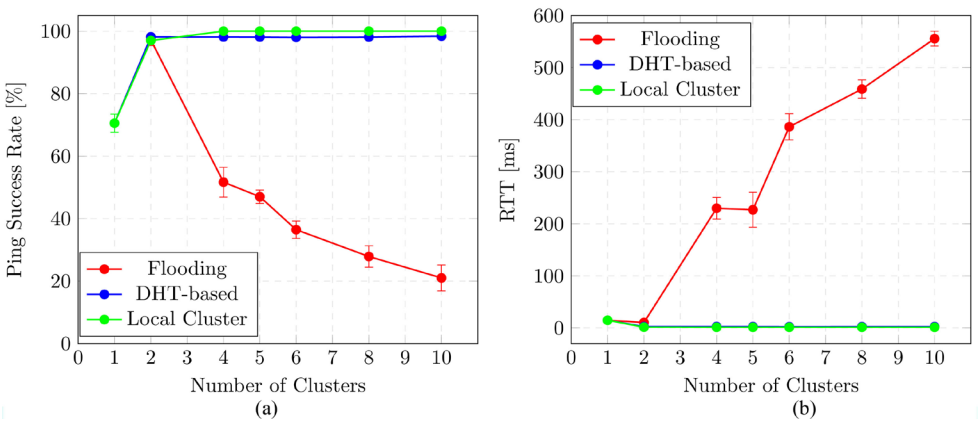

Figure 6. Ping performance, static scenarios. (a) Ping Success Rate; (b) Ping Round-Trip Time.

The flooding solution will behave like a unicast solution since every broadcast packet in the backbone will be directed to just one gateway node, who happens to be the gateway to the destination host. In this case, there will be no redundant broadcasts. However, as we increase the number of clusters further, the round-trip time in the flooding solution (red line) begins to increase. We also note that it gets worse as we increase the ping packet size. Again, the underlying reason is the increased number of (redundant) packet transmissions in the backbone, forcing new ping packets to be queued for longer periods before gaining access to the channel.

The DHT-based solution achieves lower PRTT (blue line, mostly overlapping with the green line), and remains almost constant across all scenarios. For every packet a gateway GW_X is routing for the first time to an external destination, GW_X queries the DHT to obtain the routing information required to route that packet. During this period, the first few packets for which routing information is not available are queued and are forwarded once the gateway successfully retrieves routing information from the DHT. Although there is some delay incurred by this route discovery process, it is in the order of milliseconds. Since the nodes remain in the same cluster for the entire simulation duration, the routing information retrieved is cached and reused for successive packet forwarding in the same direction. In the backbone, which is of more concern as it can become a bottleneck with heavy network loads, packets are forwarded over a unicast shortest-hop route. As discussed earlier, the DHT solution introduces new routing overheads, which are the DHT maintenance messages and the OLSR control messages, but these control messages have fixed intervals of 5 seconds and 2 seconds respectively. This control overhead increases as the number of gateway nodes increases but does not impose as high a load on the backbone as the flooding solution. PRTT in the DHT-based solution remained at about 2.5 ms for all cluster scenarios (and also across all the ping packet sizes). Again the performance of inter-cluster pings, using the DHT-based solution, is very comparable to the performance of local (intra-cluster) pings, which clearly is not the case for flooding.

The routing overhead counts the average number of maintenance messages the protocol incurs in carrying out topology maintenance and routing. As hierarchical network architectures are widely deployed to reduce routing overheads and increase scalability in ultimately large-scale MANETs, this metric gives an insight into the extent to which clustering reduces the overall routing overheads when compared to a flat MANET architecture. In **Figure 7**, the blue line shows the total number of HELLO messages, the green line the total number of TC messages, the orange line the total number of HNA messages and the red line shows the sum of all control messages. In the flat network (single cluster), there are no active gateways. All nodes exchange HELLO messages every 2 seconds and MPRs propagate TC messages every 5 seconds. In the flat network, there are many MPRs because nodes are not within reach of each other. This accounts for the high number of TC messages flooded in the MANET as seen in **Figure 7**.

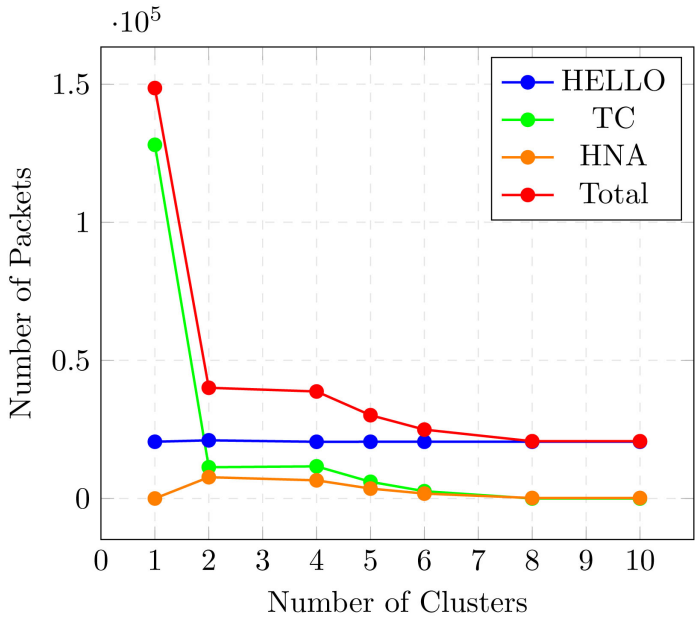

Figure 7. OLSR HELLO, TC and HNA messages.

As the number of clusters increases, the nodes are limited to smaller areas and more nodes are within reach of each other. The number of MPRs, therefore, begins to reduce and completely vanishes in the ten-cluster scenario. The number of TC message transmissions is about 130,000 messages in the one cluster scenario but drops to about 20,000 messages in the 2 cluster scenario. It continues to reduce as the number of clusters increased. HNA messages introduce additional overhead in the hierarchical design, to enable gateways to advertise reachability to external clusters to nodes within their local cluster. HNA messages are propagated every 5 seconds and forwarded only via the MPRs. This slightly increases the number of control messages. With fewer nodes per cluster as the number of clusters increases, the total number of HNA messages reduces, similar to the number of TC messages. The number of HELLO message transmissions remains constant across all cluster scenarios: every node sends these messages every 2 seconds and these messages are never forwarded. Since the number of nodes remains constant for all scenarios, the number of HELLO messages remains unchanged. Overall, the total OLSR control messages drop significantly going from a flat network (one cluster) to two clusters: from over 150,000 to about 40,000. They drop even further as we increase the number of clusters: from 38,000, 35,000, and 30,000 to 25,000 respectively.

Figure 8 shows the total traffic in the backbone for our two routing solutions. This total traffic includes the ping packets, the OLSR control messages, as well as all traffic attributed to the DHT overlay in case of the DHT routing solution. For the case of a single cluster, there is no backbone, hence both solutions show 0 packets transmitted in this case. The total traffic increases almost linearly when using flooding as the backbone routing solution, even though there is no DHT overlay to build and maintain, and no GET or PUT messages required to perform

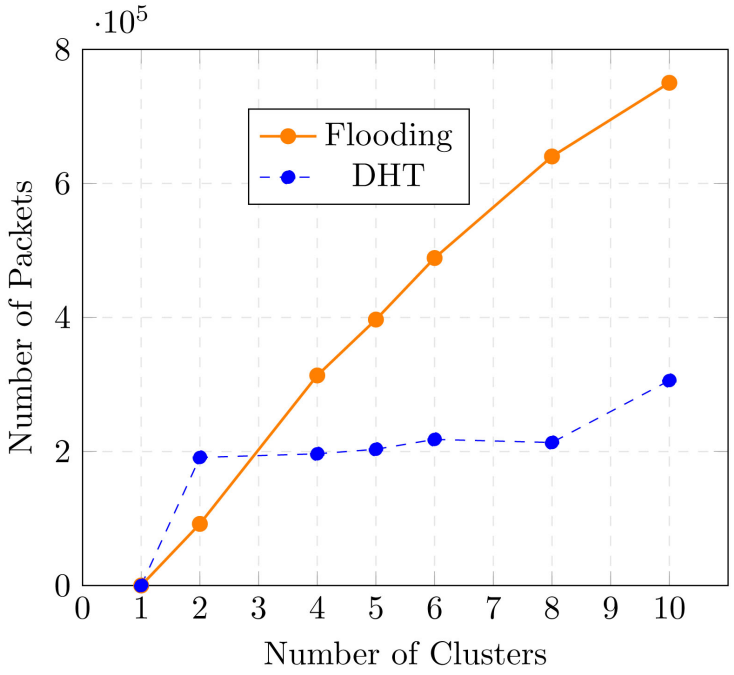

Figure 8. Total packets at the MAC layer, backbone network, static scenarios.

the backbone routing task. The total backbone traffic for the DHT-based backbone shows much less variability with the number of clusters, and is almost constant as we vary the number of clusters. Again, this confirms that the DHT-based solution scales better as we increase the number of clusters.

These reductions in protocol overheads also improve the routing protocol performance within a cluster, with PSR rising from 70% in the case of a single cluster to almost 100% in networks with more clusters (**Figure 6(a)**). Similarly, PRTT drops from about 14 ms in the case of a flat network/single cluster to 1.5 ms as the number of clusters increases (**Figure 6(b)**). This clearly demonstrates that large-scale MANETs benefit from hierarchical architectures: overall routing overhead is reduced and routing performance increases.

4.2. Mobile Scenarios

We assume that clusters are formed by the application requirements, and nodes belonging to a cluster mostly move as a group. We therefore use a group mobility model called Reference Point Group Mobility (RPGM) model [33] to model this behaviour. In RPGM, each group has a "logical center" whose movement defines the behavior of the whole cluster/MANET, including location, speed, direction, acceleration, etc. The nodes are evenly distributed in the geographic scope of a group. Each node in the group follows the movement of the group center, with some degree of freedom to allow for topology changes within a group over time. We also model scenarios where a single node leaves a cluster and joins (temporarily) an alternative cluster. More specifically, we defined and explored the following three mobile scenarios:

- **Intra-Cluster Mobility:** All nodes are mobile within their clusters only. This mobility scenario is a variation of what happens in the static scenarios, with the nodes now moving within their cluster area.
- **Nodes Switching Clusters:** Some selected hosts switch their cluster membership. The remaining nodes stay within their cluster.
- **Whole Clusters Merging and Splitting:** Clusters as a whole will be mobile, with the group of nodes comprising the cluster moving according to the RPGM model (including the gateways). Since nodes within all clusters communicate locally over the same communication channel, when two cluster come into proximity with each other, they overlap physically and logically to form a single cluster.

The traffic and evaluation metrics for the mobility scenarios are the same ones we used in the static scenario. We fix the number of clusters at 4, allowing for enough distinct non-trivial clusters to study merging and splitting. The 6 non-participating gateways serve to support routing in the various cluster where they belong, resulting in clusters of 10 nodes each. BonnMotion [33] generated mobility traces based on RPGM which were used directly in OMNeT++. We varied the average mobility speed from 0 to 20 ms^{-1} in steps of 5. The mobility traces for hosts switching cluster membership were handcrafted. We allowed 210 seconds for network stabilization as was the case in the static scenarios, and the statistics were collected over the next 3390 seconds.

As with the static scenarios, we are particularly interested in the performance of the network protocol deployed in the backbone network. For the intra-cluster routing case, we measure the impact of mobility on the metrics measured, in comparison with the static scenarios discussed above, and we use this as a base case for the other mobility scenarios. Graphs for each metric and all mobility scenarios are then plotted and the margin of error displayed on all graphs represents 95% confidence intervals.

Figure 9 shows the Ping Success Rates when nodes were mobile within their clusters, hosts switched cluster membership and clusters merged/split. The circles represent the intra-cluster ping successes and the squares in Figure 9(a) and Figure 9(c) represent the inter-cluster ping successes. In Figure 9(b), the squares represent PSR when the mobile host was the ping destination and the triangles represent PSR when the mobile host was the ping source. In Figure 9(b) the number of clusters which each host switched to over the entire network lifetime ranges from 0 to 4 in steps of 1. Five (5) nodes switched their cluster membership but at different times from each other with no particular order or interval. However, each node spent about 600 seconds in the first four clusters and 990 seconds in the fifth cluster.

For both approaches, the presence of mobility brought about frequent link breakages which resulted in packet losses and as a result, declining PSR. In the DHT-based approach, PSR across clusters remains almost as high as PSR within a local cluster, with some deterioration when clusters merge and split more often under higher mobility rates (Figure 9(c)). When hosts switch between multiple clusters,

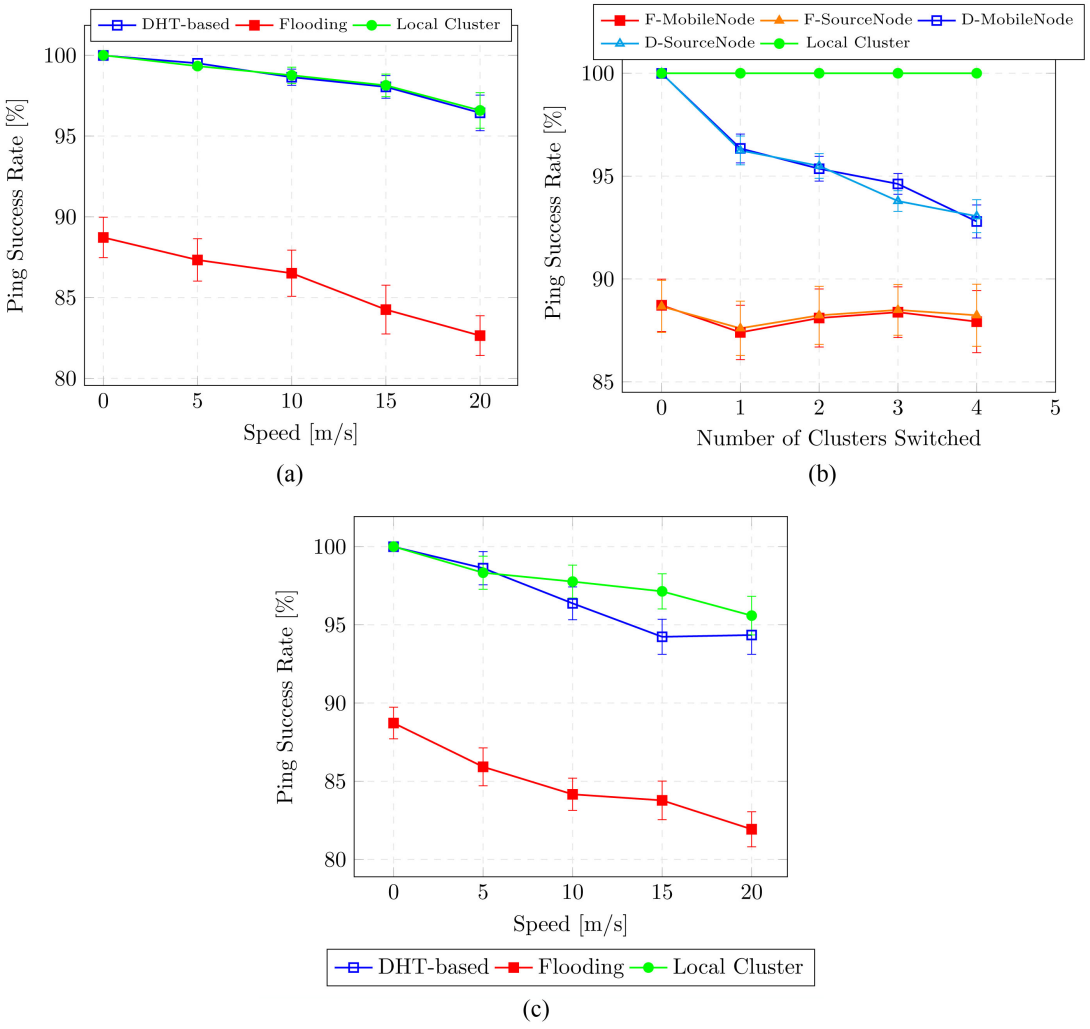

Figure 9. Ping success rates, mobile scenarios. (a) Intra-Cluster Mobility; (b) Nodes Switching Clusters; (c) Clusters Merging/Splitting.

routing information becomes stale more frequently, contributing to the declining ping success rates. Figure 9(b) also shows that PSR is, in general, independent of whether the fixed node or the mobile node initiates the ping exchange. This is somewhat expected, as we need, in both cases, a forward and a reverse route to complete the ping exchange.

With flooding, in addition to the link breaks, which become more frequent as mobility increases, the performance deteriorates for the intra-clusters mobility model and when clusters merged/split because of the high traffic load and frequent collisions in the backbone as discussed earlier. We therefore conclude that our design supports routing efficiently under different forms of mobility and outperforms the flooding approach. Mobility has a negative impact on PSR, and this is true for any routing protocol: When nodes move, they have to be rediscovered in order to be reached. Thus, messages exchanged during this rediscovery phase will be lost (either dropped, as no route is yet available, or sent to the old cluster due to stale routing information).

Figure 10 shows the Ping Round Trip Time for all mobile scenarios. Pings delivered within the clusters experienced low PRTT (green line) because the nodes are within reach of each other most of the time. For flooding, PRTT is again higher than for the DHT-based approach, with the same explanation we gave earlier. In the DHT-based solution, packets are only queued for the initial route discovery phase of our protocol and packets are delivered with a unicast protocol in single hops through the backbone for most cases. When hosts are mobile within their clusters and when clusters merged/split as seen in Figure 10(a) and Figure 10(c), the round trip times are similar for all mobility speeds and the confidence intervals show that the results overlap. The gap is a bit wider for the case of single nodes switching cluster, but overall, PRTT with the DHT-based approach is comparable to the intra-cluster PRTT and is consistently and significantly lower than that of the flooding approach.

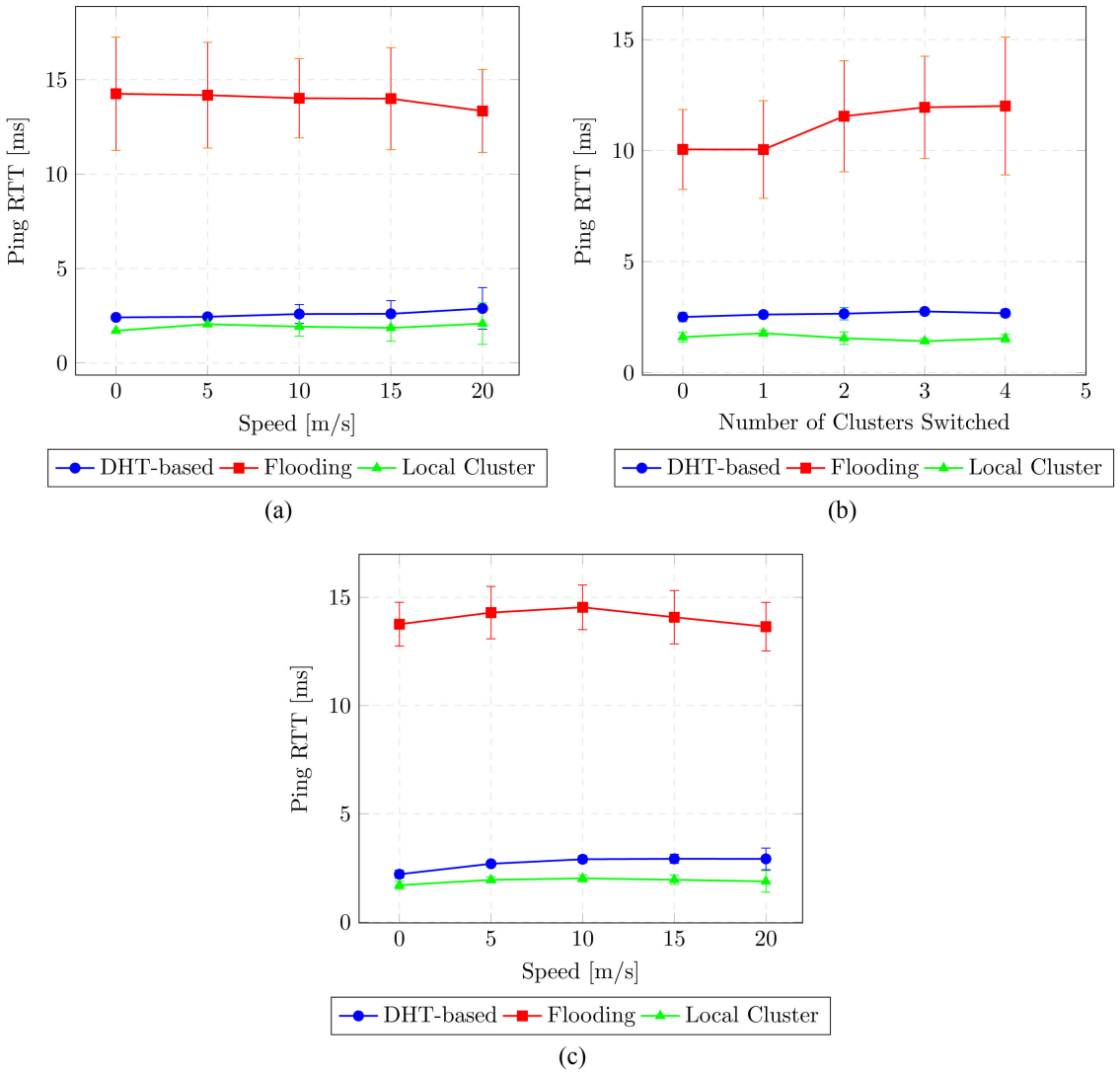

Figure 10. Ping round trip time, mobile scenarios. (a) Intra-Cluster Mobility; (b) Nodes Switching Clusters; (c) Clusters Merging/Splitting.

5. Conclusions and Future Work

In this paper, we explored ways to provide a scalable and efficient routing solution in the backbone of hierarchical MANET. Different from existing prior work, we envision that P2P applications may become popular for such inherent infrastructure-less networks that have a need for self-organization and self-healing. We therefore propose a routing solution that exploits the capabilities of a generic structured P2P overlay, based on the notion of a Distributed Hash Table.

We evaluated the proposed routing approach, discussed in Section 3 by implementing it, as well as a simple flooding-based approach, in OMNeT++. Running a number of different experiments, we confirmed that, as we expected, the flooding approach performed poorly compared with our solution. Our solution, in contrast, provides a scalable and efficient routing scheme for the different scenarios investigated here. In both the static scenario and when nodes were mobile, our solution incurred at most half of the total routing traffic the flooding approach incurred in the backbone network. Success rates remained stable above 90% in our solution. The round trip times for messages delivered to destinations in different clusters were comparable to those for messages delivered locally within the clusters, which was not the case with the flooding approach. We conclude that our DHT-based solution provides an efficient routing solution in both the static case and when mobility is involved.

As future work, our DHT-based solution could be tested with much larger MANETs, *i.e.*, increasing the total number of nodes N and splitting them into different number of clusters while maintaining the total number of nodes N in the network, to evaluate its effectiveness under the group mobility model we used in our study and other mobility models as well.

In our implementation, gateways schedule DHT queries every 30 seconds to obtain up-to-date route advertised from the DHT. As future work, the implementation could be modified such that ICMP Route Error messages will be returned to a source gateway GW_X from a destination gateway GW_Y for messages directed at a host Y who is no longer a member of that cluster, or whenever an outdated DHT entry for a node Y is used, which will then trigger a route update from the DHT for destination Y.

Finally, our DHT solution is designed to work with any routing and P2P protocol. Its scalability depends strongly on the scalability of these two key components as a function of the backbone size/number of gateways in the backbone. We only experimented with OLSR and Chord, but it would be very interesting to see how the performance of this solution changes when used with DHTs that are particularly well suited for MANETs such as OneHopOverlay4MANETs [12]. The use of a hierarchical network where the routing protocol and the DHT work together in a cross-layer fashion seems potentially well-suited for this task, especially for larger networks. Where Chord, for example requires a complex protocol to build and maintain an overlay structure among N nodes that allows GETs and PUTs to be resolved by traversing at most $\log(N)$ overlay hops, OneHopOverlay4MANETs uses information collected from the underlay routing protocol

without exchanging control messages of its own. For most cases, GETs and PUTs can then be resolved following one logical hop, which corresponds to routing requests over the shortest path among two gateways. This will reduce protocol overhead in two ways: the absence of maintenance messages to build and maintain an overlay structure, as well as a much improved performance when issuing GETs and PUTs. Such an overlay is also more successful when the topology changes [34], supporting routing more efficiently in the presence of inter-cluster mobility.

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] 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. https://doi.org/10.1109/PERCOM.2009.4912791

[2] 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

[3] 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

[4] 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

[5] Furness, J.R. (2014) Optimising Structured P2P Networks for Complex Queries.

[6] 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. https://doi.org/10.1145/2632296

[7] 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.

[8] 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*, **11**, 14-17.

[9] 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

[10] 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

[11] Zheng, X.H. and Oleshchuk, V. (2010) A Survey on Peer-to-Peer Sip Based Communication Systems. *Peer-to-Peer Networking and Applications*, **3**, 257-264. https://doi.org/10.1007/s12083-009-0064-4

[12] Al Mojamed, M. and Kolberg, M. (2017) Design and Evaluation of a Peer-to-Peer MANET Crosslayer Approach: OneHopOverlay4MANET. *Peer-to-Peer Networking and Applications*, **10**, 138-155. https://doi.org/10.1007/s12083-015-0413-4

[13] Ripeanu, M. (2001) Peer-to-Peer Architecture Case Study: Gnutella Network. *First International Conference on Peer-to-Peer Computing*, Linkoping, 27-29 August 2001, 99-100.

[14] Ou, Z.H. (2010) Structured Peer-to-Peer Networks: Hierarchical Architecture and Performance Evaluation. Dissertation.

[15] Lee, S.-H., Wong, S.H.Y., Chau, C.-K., Lee, K.-W., Crowcroft, J. and Gerla, M. (2010) InterMR: Inter-MANET Routing in Heterogeneous MANETs. *IEEE 7th International Conference on Mobile Ad Hoc and Sensor Systems*, San Francisco, 8-12 November 2010, 372-381.

[16] Anupama, M. and Sathyanarayana, B. (2011) Survey of Cluster Based Routing Protocols in Mobile Adhoc Networks. *International Journal of Computer Theory and Engineering*, **3**, 806. https://doi.org/10.7763/IJCTE.2011.V3.414

[17] Bentaleb, A., Boubetra, A. and Harous, S. (2013) Survey of Clustering Schemes in Mobile Ad Hoc Networks. *Communications and Network*, **5**, 8. https://doi.org/10.4236/cn.2013.52B002

[18] Bein, D., Datta, A.K., Jagganagari, C.R. and Villain, V. (2005) A Self-Stabilizing Link-Cluster Algorithm in Mobile Ad Hoc Networks. *8th International Symposium on Parallel Architectures, Algorithms and Networks*, Las Vegas, 7-9 December 2005, 6.

[19] Chiang, C.-C., Wu, H.-K., Liu, W. and Gerla, M. (1997) Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel. *Proceedings of IEEE SICON*, Volume 97, 197-211.

[20] Basagni, S. (1999) Distributed Clustering for Ad Hoc Networks. *Fourth International Symposium on Parallel Architectures, Algorithms, and Networks*, Perth, 23-25 June 1999, 310-315.

[21] Hussein, A.R., Yousef, S., Al-Khayatt, S. and Arabeyyat, O.S. (2010) An Efficient Weighted Distributed Clustering Algorithm for Mobile Ad Hoc Networks. *International Conference on Computer Engineering and Systems*, Cairo, 30 November-2 December 2010, 221-228. https://doi.org/10.1109/ICCES.2010.5674857

[22] Sivakumar, R., Sinha, P. and Bharghavan, V. (1999) CEDAR: A Core Extraction Distributed Ad Hoc Routing Algorithm. *IEEE Journal on Selected Areas in Communications*, **17**, 1454-1465. https://doi.org/10.1109/49.779926

[23] Kleinrock, L. and Kamoun, F. (1977) Hierarchical Routing for Large Networks Performance Evaluation and Optimization. *Computer Networks* (1976), **1**, 155-174. https://doi.org/10.1016/0376-5075(77)90002-2

[24] Özdamar, L. and Demir, O. (2012) A Hierarchical Clustering and Routing Procedure for Large Scale Disaster Relief Logistics Planning. *Transportation Research Part E*: *Logistics and Transportation Review*, **48**, 591-602. https://doi.org/10.1016/j.tre.2011.11.003

[25] Tsuchiya, P.F. (1988) The Landmark Hierarchy: A New Hierarchy for Routing in Very Large Networks. *ACM SIGCOMM Computer Communication Review*, **18**, 35-42. https://doi.org/10.1145/52325.52329

[26] Boukerche, A., Turgut, B., Aydin, N., Ahmad, M.Z., Bölöni, L. and Turgut, D. (2011) Routing Protocols in Ad Hoc Networks: A Survey. *Computer Networks*, **55**, 3032-3080. https://doi.org/10.1016/j.comnet.2011.05.010

[27] Stoica, I., Morris, R., Liben-Nowell, D., Karger, D.R., Frans Kaashoek, M., Dabek, F. and Balakrishnan, H. (2003) Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications. *IEEE/ACM Transactions on Networking* (*TON*), **11**, 17-32. https://doi.org/10.1109/TNET.2002.808407

[28] Clausen, T. and Jacquet, P. (2003) RFC 3626. Optimized Link State Routing Protocol (OLSR). https://doi.org/10.17487/rfc3626

[29] Varga, A. (2019) Omnet++ Simulation Manual. https://omnetpp.org/

[30] INET Framework User's Guide, January 2019. https://inet.omnetpp.org

[31] OverSim the Overlay Simulation Framework, January 2019. https://inet.omnetpp.org

[32] Echegini, N. (2018) A DHT-Based Routing Solution for Hierarchical MANETs. Master's Thesis, Carleton University, Ottawa.

[33] Aschenbruck, N., Munjal, A. and Camp, T. (2011) Trace-Based Mobility Modeling for Multi-Hop Wireless Networks. *Computer Communications*, **34**, 704-714. https://doi.org/10.1016/j.comcom.2010.11.002

[34] Kunz, T., Esfandiari, B. and Ockenfeld, F. (2017) Efficient Routing in Mobile Ad-Hoc Social Networks. 2017 *IEEE International Conference on Internet of Things* (*iThings*) *and IEEE Green Computing and Communications* (*GreenCom*) *and IEEE Cyber*, *Physical and Social Computing* (*CPSCom*) *and IEEE Smart Data* (*SmartData*), Exeter, 21-23 June 2017, 216-222. https://doi.org/10.1109/iThings-GreenCom-CPSCom-SmartData.2017.37