Christian Gottron, André König, Ralf Steinmetz: A Cluster-Based Locality-Aware Mobile Peer-to-Peer Architecture. In: Proceedings of the 8th International Workshop on Mobile Peer-to-Peer Computing (MP2P 2012), March 2012.

A Cluster-Based Locality-Aware Mobile Peer-to-Peer Architecture

Christian Gottron, André König, Ralf Steinmetz
Multimedia Communications Lab, TU Darmstadt, Darmstadt, Germany
{christian.gottron, andre.koenig, ralf.steinmetz}@kom.tu-darmstadt.de

***Abstract*—Mobile peer-to-peer networks combining a mobile ad hoc network underlay with a peer-to-peer overlay offer a flexible means for providing communication services in absence of a communication infrastructure. However, mobile peer-to-peer networks inherit several challenges from the underlying architectures as, e.g., strongly limited resources and a highly dynamic topology. Due to this, the overlay has to be adapted in order to provide reliable services. We introduce a mobile peer-to-peer architecture that harnesses the location awareness and clusters nodes by their location. By means of simulation, we show that the overhead generated by lookup requests, bootstrapping, and the updating mechanism of the routing tables can be strongly reduced.**

I. Introduction

Mobile peer-to-peer (MP2P) architectures are able to store and retrieve data in a mobile and decentralized network. They are a combination of a *mobile ad hoc network* (MANET) underlay with a *distributed hash table* (DHT) overlay and operate without a pre-deployed communication infrastructure. Due to this, MP2P networks are suitable for multiple environments as, e.g. disaster relief, development aid projects, or tactical scenarios.

Most DHTs require a reliable underlay that provides sufficient resources (as, e.g., the Internet). Due to this, DHTs are not able to handle bandwidth limitations or unstable links properly. Thus, challenges arise when a mobile, wireless underlay as a MANET is used. Therefore, the overlay protocol has to be adapted in order to reduce overhead generated by lookup requests and by the bootstrapping mechanism. Furthermore, due to node mobility and the highly dynamic topology, the overlay's routing table entries become obsolete more frequently. Therefore, the update mechanism has to be revised.

We propose an MP2P architecture that harnesses node locality. Our approach maps physical proximity to overlay identities. Logically close overlay IDs are assigned to physical neighbors. Due to this, physical neighbors are also virtual neighbors in the P2P network. As a result, overhead generated by the lookup mechanism is reduced as lookups are more target-oriented regarding the physical location of the destination. Locality awareness also improves the bootstrapping algorithm as a node can obtain all required information from direct physical neighbors. In order to update the routing tables, we propose to harness information provided by the underlay. Therefore, overlay identifiers and small parts of the routing table are attached to the periodic underlay maintenance messages. Further, periodic updates from physically distant nodes are required as routing table entries would be outdated otherwise quickly due to the highly dynamic topology.

This paper is structured as follows: in the next section, we provide background information on P2P and MANET networks. The third section introduces related work which motivated our work and the fourth section introduces our approach. Thereafter, we discuss the results of our simulation-based evaluation, conclude this paper, and give a brief overview about future works.

II. Background

MANETs are wireless decentralized networks. They operate mostly in the network layer and provide connections between nodes without the requirement of a predefined infrastructure. Whenever the destination is not within the transmission range of the source, intermediate nodes have to provide router functionality. The network topology is highly dynamic due to the characteristics of the wireless links and the node mobility. To cope with these challenges, adapted routing protocols are required. We distinguish between proactive and reactive protocols. Proactive protocols establish and maintain routes to every node in the network. Reactive routing protocols, on the other hand, establish routes on demand only.

The Optimized-Link-State-Routing (OLSR) [1] protocol is a well known proactive routing protocol for MANETs. OLSR is one of the few MANET protocols that has been evaluated in multiple real world testbeds as DUMBO [2] or the Freifunk [3] project. This link state protocol uses periodically sent Hello messages to detect physical neighbors. Topology control (TC) messages distribute the topology information in the whole network. To route packets, each node selects a set of direct neighbors. This set of neighbors called multi point relays (MPR) has to be able to provide a connection to all second hop neighbors of the node. TC messages are only forwarded by MPRs. OLSR relies on a hop metric to estimate the distances between nodes and to calculate the routing tables. However, testbed evaluations of hop-based metrics have shown that they are inefficient [4][5]. Therefore, the *expected transmission count* (ETX) [6] extension of OLSR was introduced. This metric of OLSR also considers the maximum throughput of a link as routing metric and not only the hop distance. Due to this, routes established with the ETX metric are more reliable.

Most P2P networks provide services without requiring a centralized entity. They are built at the application layer on top of an existing network which is called the underlay. The underlay provides the physical connections between the peers. In traditional P2P networks, the peers form a network called overlay on top of the Internet as underlay. Distributed Hashtables (DHT) are a subtype of P2P networks that scales well to the network size regarding both number of required hops and routing table size. A unique ID is assigned to each node that joins the DHT overlay. This ID is used for overlay routing. Virtual neighbors are nodes with a small numerical distance regarding their node IDs. IDs are also assigned to services and objects which are provided by the network. Those objects are administrated by the overlay node with an ID closest to the object's ID.

Pastry [7] is a well known DHT. In order to provide an efficient routing service, Pastry relies on two routing tables. The main routing table provides routes to nodes which are distributed in the ID namespace. It is structured as a tree. Pastry uses a longest prefix match routing algorithm. Due to this, the routing table of a node x provides links to nodes that match up to the $(n-1)$th prefix of x's node ID and differ on the nth digit where n is the row number of the routing table entry. Whenever two nodes match the requirements of a routing table entry, a proximity metric is harnessed to determine which node has to be stored. For this, ping messages are used to determine the round trip time. The proximity metric is used to reduce the routing overhead. The numerical basis of the ID and, thus, the structure of the routing table is defined by the parameter b. The leaf set stores routes to virtual neighbors and has a ring structure. The leaf set is used for the last step of a routing process to deliver the request to the correct node. Furthermore, it is used to determine which node is responsible for an object. The node numerically closest to the overlay ID of the object is responsible for this object.

III. Related Work

Multiple MP2P architectures were introduced in the recent years. Many of them combine a DHT as overlay with a MANET as underlay. In this section, we introduce related work that has motivated our approach.

MADPastry introduced by Zahn et al. [8] combines a Pastry DHT with a MANET underlay. They propose a reactive protocol for underlay routing. MADPastry harnesses node proximity in order to reduce overhead. Clusters are formed around predefined nodes. Nodes within a cluster share the same prefix. Due to the mobility of the predefined nodes, the physical location of the clusters varies over time. The Pastry routing algorithm is modified so that the routing table provides routing information for the first hop only. Thereafter, the leaf set is used to forward the lookup message to the destination. Due to this, multiple overlay hops in the leaf set are required in large scale scenarios. This results in overhead generated by the lookup process.

Beside MADPastry, PeerNet [9] also harnesses the location awareness of nodes to minimize routing overhead. This architecture splits the area of operation in several equal sized zones and assigns a part of the ID space to each of the zones. Each node in the network is assigned to one of these zones, which will be the node's home zone, according to the probability that the node can be found in this zone. Those home zones provide a proxy which stores information about the recent position of each node which belongs to this zone. However, we require proxies which have to be reliable and highly available. Furthermore, overhead generated by routing may be increased whenever a node is not within its home zone, especially when the information stored at the proxy is outdated. It may also be hard to predict the node's location correctly.

Other approaches as the Stealth DHT introduced in [10] harness the heterogeneity of the network. Nodes which are reliable regarding connectivity and provide a high amount of resources are used to offer most of the services as storing objects and forwarding requests. Mobile nodes with strongly limited resources, on the other hand, only have to provide basic services. However, highly available nodes are required and this algorithm can be easily misused by selfish nodes.

Castro et al. proposed an MP2P architecture based on a modified Bamboo architecture. The Bamboo architecture is very similar to Pastry but updates the routing table not on demand. Instead, update messages are sent periodically. By modifying the update interval, Castro et al. reduce overhead generated by the update mechanism. However, this may also influence the stability of the network.

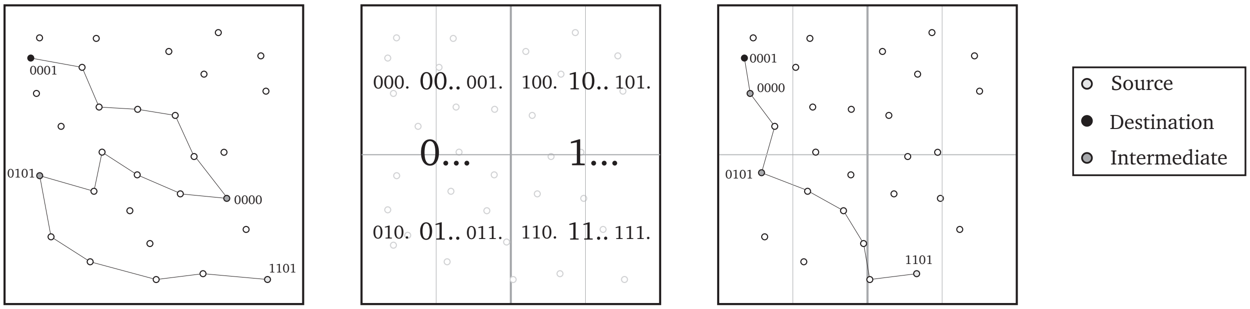
Further information on P2P, MANET, and MP2P architectures can be found in [11].

IV. Architecture

Traditional P2P architectures are developed for scenarios with a very high number of participants. Due to this, the scalability of the architecture is one of the most important challenges. Yet, the number of participants in, e.g., disaster relief scenarios is low compared to traditional P2P scenarios. Therefore, scalability is not anymore the dominant challenge for our MP2P architecture. However, new challenges have to be considered.

A basic challenge for MP2P architectures is to reduce the overhead generated by the lookup algorithm. Whenever a lookup for an object is initiated, each node compares its own ID with the object ID. After calculating the number of matching prefixes of those IDs, a next hop overlay node is selected that matches at least one more digit. This way, the lookup message gets closer regarding the virtual distance to the destination with each hop. However, as virtual neighbors are mostly not also physical neighbors, the lookup message is not routed on the shortest physical path as shown in our example (Figure 1(a)).

We propose a location aware MP2P architecture. Each node has to be aware of its physical location (e.g. by using GPS). Therefore, we combine a Pastry DHT as overlay with an OLSR ETX underlay. The node IDs are not distributed randomly, but are a function of the node's physical position. Therefore, we split the area of operation in several clusters and assign an ID prefix to each of those clusters. The ID of each node that is



(a) Unmodified MP2P (b) Example for a clustered area of operation (c) Clustered MP2P

Fig. 1. An example for a lookup in an unmodified and a clustered P2P network

member of this cluster is defined by this prefix and a random suffix.

Two parameters are used to define the number and structure of the clusters. The clustering level defines the depth of the clustering. The second parameter defines the number of clusters per level. The clustering level also defines the prefix size of the cluster ID. Therefore, the numerical basis of the ID and, thus, the parameter b is a function of the clusters per level. For example, we assume a scenario with two clusters per level ($b = 1$). Due to the the number of clusters per level we have a binary ID. In a small scale scenario, with only a strongly limited number of nodes and a small area of operation, a single clustering level could be sufficient. This results in an overall number of two clusters and a prefix size of one digit. However, larger scenarios require a higher clustering level in order to increase the overall number of clusters and to improve the structure of the routing path. An example of a scenario with a clustering level of three is shown in Figure 1(b). The area of operation is split in $2^{level} = 8$ clusters. Furthermore, the prefix defined by the clusters is increased to three.

As the nodes are mobile, they may leave a cluster. Due to this, a new ID is required with the prefix of the new cluster. This results in an increased churn, as also stored objects have to be redistributed and routing tables have to be updated. However, due to the combination of virtual and physical neighborhood, the overhead generated by churn is low. Furthermore, in order to reduce the overhead generated by nodes that move along the border between two clusters and change their cluster frequently, threshold areas are introduced. When a node leaves a cluster, as shown in Figure 2, it has to enter the threshold area (point a). Even though this node has already left the cluster (point b), the node ID is still the same until this node leaves the threshold area (point c). Thereafter, a new ID is required with the prefix of the new node.

A. Routing Tables and Routing

Similar to Pastry, our approach uses a leaf set and a main routing table. As node IDs depend on the physical location of nodes, the leaf set provides not only links to virtual neighbors but also to physical neighbors.

The leaf set provides links to nodes in the virtual neighborhood of the owner of the routing table. As the IDs are a function of the geographical position of the nodes, each node in the leaf set is also a physical neighbor. As the Pastry leaf set has a ring topology, problems may occur when the virtual neighbor is a member of another cluster. In a scenario as shown in Figure 3, the virtual neighbor of a node with the prefix 00 and the lowest prefix would be located in the cluster 11. Due to this, routing to a virtual neighbor would result in sending a message from one end of the area of operation to the other. Therefore, we modified the structure of the leaf set to ensure that virtual neighbors are always located within the same cluster. Each cluster forms an own leaf set ring as shown in Figure 3. Due to this, the leaf set is used for intra-cluster

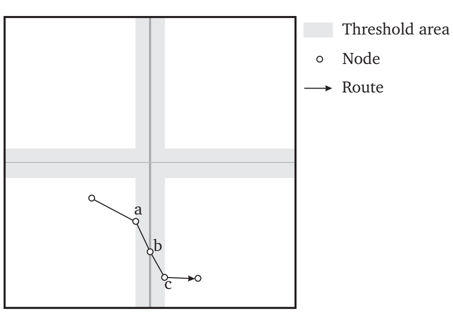


Fig. 2. The effect of thresholds between clusters

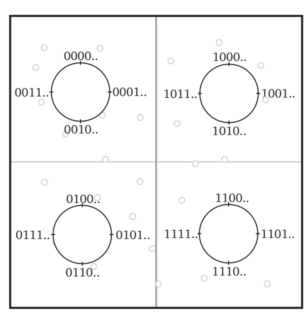


Fig. 3. Structure of the adapted leaf set

routing only. Inter-cluster routing is provided by the routing table only. In contrast to Pastry, the leaf set size is not defined by a fixed parameter, but by the number of nodes in the cluster. The leaf set of our approach provides links to all nodes in the particular cluster.

The main routing table is used to route a lookup message to the target cluster. Similar to Pastry, the routing table is prefix-based and tree structured. Yet, the prefix of a node is not random but depends on the cluster in which the destination ID can be found. The routing table is used to route a message to the correct cluster. Due to this, the number of clustering levels defines the number of rows stored in the routing table. Similar to Pastry, the number of entries is defined by the parameter b. As mentioned in Section II Pastry uses a proximity metric for routing table maintenance. However, no proximity metric is used in the clustered approach as ping messages generate overhead. Furthermore, preferring nodes that are physically close to the owner of the routing table has a drawback. Each node, that is stored in the routing table , is located in an other cluster. When a proximity metric would be used, a node that provides the minimum distance to the owner of the routing table would be preferred. Yet, this closest node would also be the node that is located closest to the cluster boarder to the cluster of the owner of the routing table. This increases the probability, that the preferred node changes the node ID soon. Due to this, such a node may be valid for this specific routing table entry for a strongly limited time only.

The routing mechanism of our clustered approach is similar to Pastry. When receiving a lookup request, we check the cluster of the destination ID and the own cluster. When the destination cluster is reached, the leaf set is used to determine the destination node. Otherwise, the lookup request is forwarded to the next clustering level.

B. Updating Routing Tables

Due to node mobility, node ID adaptation due to clustering, and churn, both the routing table and the leaf set have to be updated frequently. Pastry initiates updates whenever a node does not respond to a request or when a node joins the network. Due to packet loss of the wireless, mobile nodes, an unreplied request may not always be the result of a broken link. Furthermore, an on demand update mechanism may not be sufficient in a scenario with a highly dynamic topology. Nodes that have not sent or forwarded a message during a long period of time may otherwise be disconnected from the overlay due to multiple broken or invalid links. We propose a periodical update algorithm for both, the routing table and the leaf set.

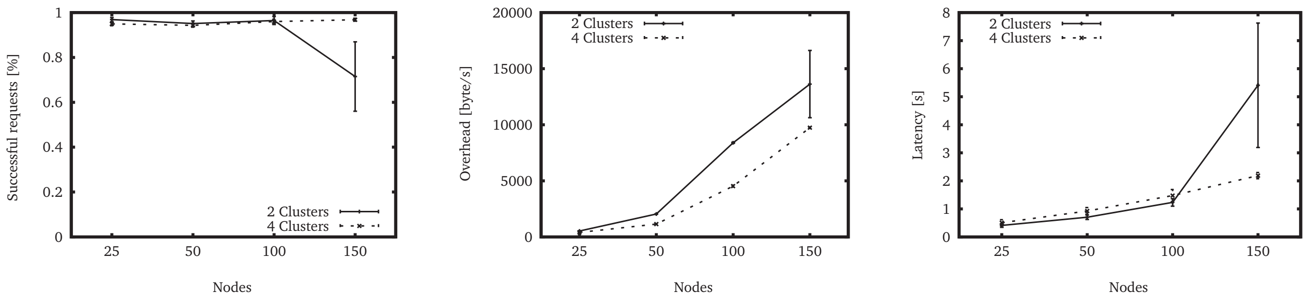
The leaf set is maintained periodically. This is performed efficiently based on a cross-layer information exchange in which the node ID and information about known leaf set nodes is attached to the Hello messages of OLSR. Those Hello messages are sent periodically every two seconds. The leaf set information contains the node ID, the IP address and a time to live field. This information is only broadcasted within the cluster. Due to this, the leaf set is updated periodically. Furthermore, nodes that leave a cluster send a message to the virtual neighbors. This ensures that nodes leaving the cluster are removed from the leaf set quite fast and that objects are replicated correctly. However, even when a node is not able to sent a notification message when leaving the cluster or the network, this node will be removed automatically from the routing tables during the next routing table update.

As nodes may leave a cluster or the network at all, the routing table has to be maintained regularly. Pastry marks an entry as outdated whenever no response is received to a message, that has been sent to the corresponding node. Yet, this is not efficient in the context of a wireless underlay. Messages may get lost even though the destination is still available. Furthermore, we have a dynamic topology due to the mobility of nodes and due to the cluster-based architecture. Therefore, periodic updates of the routing tables are required. For this, each node contacts each routing table entry every 5 seconds. A node that receives such a request checks whether it matches the requested ID. When the ID does not match the requested ID, the routing table is used to check for an entry that matches the requested cluster. If so, a reply is sent including the proposed node. Otherwise, an empty reply is sent. When the node ID matches the requested ID, the node has to check whether it is within the threshold area. In both cases, a reply message is sent. If the node is within the threshold area, a link to another node within the same cluster is added to the reply message. When the requesting node receives a positive reply, the time to live field of the entry is updated. If another node is proposed by the requested node, the routing table has to be updated. When no response at all is received, the routing table of a neighbor node is used to update this routing table entry.

C. Bootstrapping

Each node that joins the network requires a bootstrapping node. From this node, the joining node receives information for building the routing tables. Further, the other nodes in the network, especially the nodes in the leaf set of the new node, are informed that a node has joined the P2P network. When bootstrapping a Pastry network, the joining node has to know the address of a node that is already part of the P2P network. This bootstrapping node routes the join request to the virtual neighbor of the booting node.

As our approach combines the virtual with the physical neighborhood, we simply use physical neighbors to boot nodes. As each node in the DHT adds its node ID and leaf set information to the periodically broadcasted Hello messages of the underlay, neighbors that participate in the DHT can be identified easily. As a result, the virtual neighbor of the new node can be reached within a single hop and no costly routing process is required. Due to this, only little overhead is generated during bootstrapping. Further, overhead generated due to churn is low as objects have to be distributed in the cluster only. Also the number of required hops per bootstrapping can be reduced further by preferring the neighbor with an ID closest to the booting node.



(a) Probability of a successful request

(b) Overhead

(c) Latency

Fig. 4. Results of scenarios with different network size

V. Evaluation

In this section, we introduce implementation details as well as the parameters of our simulation studies. Further, the results of the evaluation are presented and discussed.

A. Implementation

For the evaluation of our MP2P architecture, the discrete event simulator OMNeT++ [12] is used. This open source simulator is well known and provides multiple frameworks. The underlay of our approach is based on the INET framework [13]. This framework provides implementations of MANET routing protocols including OLSR with and without the ETX extension. Further, basic applications as traffic generators, mobility models, models for wireless channels, transport layer, network layer, and link layer protocols are provided. On the other hand, our overlay is based on the OverSim [14] framework. OverSim provides implementations of DHTs as Chord, Pastry, and Kademlia. Traffic generators for uploading and requesting objects in the overlay are also provided. Further, simplified underlays are available. However, no mobile or wireless underlay is provided by the OverSim framework. We implemented our approach by combining the previously mentioned frameworks and modifying the algorithms of both, the Pastry overlay and the OLSR underlay.

B. Setting

In order to adjust the area of operation and the node density, the *Ad-Hoc Network Simulator* (ANSim) [15] has been used. This tool simulates node mobility and derives the connectivity between nodes within the network. As a dense network is assumed due to the scenario, a setting is chosen where a connected network is typically achieved, i.e., a route between an arbitrarily chosen pair of nodes can be established with a high probability. Each node uses WiFi according to the 802.11g standard and an average transmission range of about 200m. The resulting area of operation according to ANSim is shown in Table I. For the underlay routing, the modified OLSR ETX protocol as described in Section IV has been used. For the overlay lookup functionality, the clustered Pastry approach is used. The parameter b is set to 1 and the cluster $level$ is either 1 or 2. Due to this, scenarios with two and four clusters were simulated during the evaluation. A random way point model is used for mobility. This model has been implemented as part of the INET framework. Each node moves with a speed up to 1 m/s and changes its speed and direction after a time period of 10s to 60s.

Nodes	area of operation
25	500m x 500m
50	800m x 800m
100	1100m x 1100m
150	1400m x 1400m

TABLE I
Number of nodes and area of operation

Up to 50 objects are provided by nodes within the DHT. Those nodes are selected randomly. Each object has a size of 2 kbytes and a lifetime of 250s. Outdated objects are discarded and replace by a new one. Two replicas of each object are generated and stored by the virtual neighbors. After the booting phase, lookup requests are sent every 10 seconds on average in the network. Each simulation runs for 30 simulated minutes. Further, for more reliable results, each setting is simulated at least ten times.

C. Results

In order to evaluate the scalability of the clustered approach, scenarios that differ in network size are simulated. Three metrics are used to determine the efficiency of the clustered MP2P network. The fraction of successful lookup operations is a metric to determine how efficient the MP2P network operates regarding the retrieval and availability of data objects. This is very important, as an MP2P network, where either objects can not be accessed in a reliable way or are not stored correctly, is not usable. As mentioned before, bandwidth is strongly limited. Therefore, overhead generated by the MP2P algorithm has to be minimized. Updating the leaf set generates the highest overhead and, therefore, has to be monitored. The overall traffic has to be low in order to enable a reliable lookup operation and to reduce the packet loss due to congestion and collisions. Furthermore, other applications as text or voice chat may be required in our scenario, and, therefore, additional traffic may be generated by those applications. As the time required for a lookup request may be important regarding usability, the latency is used as third metric. Therefore, the overall time between sending the request and receiving the object is measured.

In Figure 4(a) the fraction of successful lookup requests is shown as a function of the network size and the clustering level. Most results of the clustered approach provide a success rate of about 95%. However, in small scale scenarios, the scenario with four clusters provides worse results compared to the two cluster scenarios. As a consequence of the increased number of clusters, nodes have to changes their ID more often. Therefore, routes become invalid more often and this may lead to incorrectly routed and, therefore, lost lookup requests. Furthermore, due to node mobility, network partitioning may result in an increased fraction of failed lookup requests. As mentioned in Section V-B, a dense network is simulated. This reduces the probability of a network partitioning strongly. Yet, single nodes or small groups may leave the network. Except from object replication nothing else can be done when nodes are not anymore within transmission-range of the MP2P network. However, due to the increased size of the area of operation in scenarios with a high number of nodes, the influence of those effects is decreased. In the scenario with 150 nodes, the fraction of successful lookup requests in scenarios with two clusters is decreased strongly due to high congestion. By increasing the clustering level, the traffic can be reduced and a reliable lookup functionality can be still provided. Therefore, an increased number of clusters is required in scenarios with more than 100 nodes.

Most overhead is generated by the leaf set update mechanism. Due to this, we focus on this update traffic to determine the generated overhead. As shown in Figure 4(b), the overhead generated each second in the network increases strongly as a function of the network size. By increasing the number of clusters, the overhead can be reduced significantly. However, even in our worst case scenario with 150 nodes, the overall overhead due to the leaf set update mechanism is only about 200 kbytes in the whole network.

The latency is used to derive the delay generated due to the lookup and request mechanism. The approaches with a higher clustering level increase the number of overall hops. Due to this, also the latency is increased. However, compared to the overall latency, the difference between two approaches with different clustering levels is quite low. Further, in the scenario with 150 nodes and two clusters, the latency is increased strongly due to congestion.

D. Discussion

As shown in the evaluation section, small scale scenarios with four clusters have a lower probability for a successful lookup process compared to scenarios with only two clusters. Yet, with increasing network size, scenarios with a higher clustering level provide equal or even better results. Furthermore, overhead generated in those scenarios is strongly reduced compared to similar scenarios with two clusters as the number of nodes in the leaf set is reduced and, therefore, the size of the leaf set update messages that have to be broadcast frequently is decreased. However, a mechanism that increases the clustering level automatically as a function of the network size would most probably be able to provide better results especially in scenarios where the number of nodes is not fixed. This mechanism could be harnessed to split a cluster into two sub-clusters whenever the number of nodes within this cluster exceeds a specific threshold. Furthermore, two clusters can be merged whenever the number of nodes falls below a specific threshold. As only the routing tables of those nodes have to be adapted that are within the clusters that have to be modified, this would not have a strong effect on the rest of the network.

VI. Conclusions

We have introduced a new cluster-based architecture for an MP2P network that is based on a combination of a MANET and a DHT. This architecture is able to provide a reliable lookup functionality even though the nodes are mobile and the topology is dynamic. Furthermore, we compared several scenarios with different network sizes and a different clustering level. We have shown that the overhead can be strongly reduced by adapting the clustering level without any drawbacks in larger scale scenarios.

In our future work, we plan to consider adapting the cluster size automatically. We assume that this improves both the efficiency and reliability in the architecture as the overhead can be decreased. Further, we plan to increase the robustness of the network against both malicious behavior and packet loss due to the underlying network. For this, existing mechanisms out of the field of MANETs and DHTs have to be evaluated. When required, we want to develop an adapted replication mechanism in order to increase the availability of the stored objects.

References

[1] T. Clausen and P. Jacquet, *RFC 3626: Optimized Link State Routing Protocol (OLSR)*, Network Working Group Std., 2003. [Online]. Available: http://tools.ietf.org/html/rfc3626

[2] K. Kanchanasut et al. "A Multimedia Communication System for Collaborative Emergency Response Operation in Disaster-affected Areas," intERLab, AIT, Tech. Rep. TR 2007-1, 2007.

[3] Förderverein Freie Netzwerke e.V. [Online]. Available: http://wiki.freifunk.net/Kategorie:English

[4] H. Lundgren et al., "Coping with Communication Gray Zones in IEEE 802.11b based Ad hoc Networks," in *Proc. of the ACM WOWMOM*, 2002.

[5] C. Gottron et al., "The Rise and Fall of the AODV Protocol: A Testbed Study on Practical Routing Attacks," in *Proc. of the IEEE LCN*, 2010.

[6] D. De Couto et al., "A High-Throughput Path Metric for Multi-Hop Wireless Routing," *Wireless Networks*, vol. 11, 2005.

[7] A. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems," in *Proc. of the IFIP/ACM Middleware*, 2001.

[8] T. Zahn and J. Schiller, "MADPastry: A DHT Substrate for Practicably Sized MANETs," in *Proc. of the IEEE ASWN*, 2005.

[9] A. Gopalan and T. Znati, "PeerNet: A Peer-to-Peer Framework for Service and Application Deployment in MANETs," in *Proc. of the ISWPC*, 2006.

[10] A. MacQuire et al., "Performance Analysis of Stealth DHT with Mobile Nodes," in *Proc. of the IEEE PERCOMW*, 2006.

[11] C. Gottron et al., "A Survey on Security in Mobile Peer-to-Peer Architectures Overlay-Based vs. Underlay-Based Approaches," *Future Internet*, vol. 2, no. 4, 2010.

[12] A. Varga, "OMNeT++," in *Modeling and Tools for Network Simulation*, K. Wehrle, M. M. Günes, and J. Gross, Eds. Springer Berlin Heidelberg, 2010.

[13] INET Framework. [Online]. Available: http://inet.omnetpp.org

[14] I. Baumgart et al., "OverSim: A Flexible Overlay Network Simulation Framework," in *Proc. of IEEE GI*, 2007.

[15] H. Hellbrück. (2011) ANSim - Ad-Hoc Network Simulation Tool. [Online]. Available: http://www.ansim.info