Mesh-DHT: A Locality-Based Distributed Look-Up Structure for Wireless Mesh Networks

Hanno Wirtz, Tobias Heer, René Hummen, Klaus Wehrle
Chair for Communication and Distributed Systems, RWTH Aachen University
{wirtz, heer, hummen, wehrle}@comsys.rwth-aachen.de

***Abstract*—Distributed Hash Tables (DHTs) offer an elegant and fully distributed solution for reliably storing and retrieving data. Wireless Mesh Networks (WMNs) envision a fully decentralized fashion, and as such require efficient decentralized mechanisms for service discovery, mobility support and data storage and retrieval. Hence, DHTs and WMNs seem to complement each other nicely and even share common traits and challenges, such as multi-path routing and dynamic membership of unreliable nodes. Existing Internet-based DHT approaches are designed to emphasize performance and stability in Internet scenarios and do not consider the special conditions in WMNs. In particular, they do not focus on the impact of the physical neighbor relations of DHT nodes and assume efficient global connectivity. In contrast, in a WMN, locality of communication is essential to avoid unnecessary multi-hop data transmissions and congestion on the wireless link. We present Mesh-DHT, an approach for building a scalable DHT in WMNs that puts special emphasis on the locality of nodes and links. We construct a stable, location-aware overlay network that enables fully distributed organization of information. By design, our DHT geometry is closely aligned to the network topology of the WMN to emphasize local communication. We show that our approach preserves locality in the overlay construction, is robust against node failure, and makes efficient use of local information. These properties make our approach scalable even in the presence of hundreds of mesh nodes.**

I. Introduction

Wireless Mesh Networks (WMNs) provide network access to a large number of users with wide-spread network coverage and support for distributed services. In a WMN, both client-to-router and router-to-router communication occurs exclusively on the wireless medium. The limited wireless communication range introduces a notion of *locality* between physically close routers that can communicate with each other. To reach distant nodes, messages traverse multiple hops, placing additional load on the wireless medium at each hop.

A large-scale WMN requires a look-up structure to support storage, client mobility and generic services. A centralized implementation defeats the inherent robustness and capabilities of the network as this central service provider constitutes a bottleneck in terms of communication, computation and node failure. In the Internet, fully distributed data management has been successfully realized using Distributed Hash Tables (DHT). A DHT spreads the construction and maintenance load of an abstract index infrastructure over all participating nodes, while providing a simple data access interface via *put()* and *get()* calls. For example, mobility support in a WMN can be realized by registering and querying the current location of a mobile device from the DHT.

The design principles of DHTs fit directly to the distributed nature of a WMN. Besides distribution of functionality and responsibility among nodes, both network concepts emphasize redundancy, resiliency, and multi-path routing between nodes. However, constructing and maintaining a stable DHT in a dynamic WMN is challenging. Especially Internet-scale DHTs, such as Chord [1] or CAN [2], fail in the context of WMNs as they are designed for stable physical networks. The major reasons for this failure are: a) message loss over wireless links appears as target node failure, b) high degree of node dynamics due to joining/leaving nodes, and c) very high communication cost to realize communication between close identifiers in the DHT that map to distant physical nodes. Efforts to adapt established designs [3], [4] are limited to patching selected problems but do not meet the above challenges by design. GHT [5], for example, incorporates the geographic proximity of nodes in the DHT operation, but does not introduce an explicit mapping of data items to nodes. Instead, it relies on a hit-and-miss mapping by way of the GPSR perimeter mode.

We propose Mesh-DHT as a DHT concept that exploits the physical locality of nodes to construct an overlay routing geometry using node identifiers that reflect the wireless neighbor relations. To incorporate the locality of nodes in the DHT construction, we base a Voronoi tessellation of the DHT address space on these locality-oriented identifiers. Nodes that are close in the physical network thus have close identifiers and manage nearby regions of the DHT address space. This close coupling allows to minimize the overhead in DHT communication, because nodes mainly talk to physically close nodes, thus requiring fewer transmissions. Hence, communication locality affords scalability and performance as key goals of our approach.

This paper is organized as follows. Section II establishes basic terminology and gives a high level overview of our approach. Section III discusses the specific requirements for establishing a DHT in a WMN. We present our approach to establishing the distributed namespace of the DHT and to constructing the overlay in Section IV. In Section V, we evaluate our approach and show its applicability by providing results from simulated and real-world networks. Section VI presents related work on traditional DHT approaches and DHT designs for multi-hop wireless networks. Section VII concludes the paper.

II. Overview of our Approach

In this section, we give a brief high-level description of our approach and establish a basic terminology. Nodes in the WMN that are in radio range of each other are *physical* *neighbors*. Communication between distant physical nodes requires a routing scheme based on geographic identifiers or IPs. To show the feasibility of our approach, we use a greedy geographic routing scheme on *virtual coordinates* that reflect the locality of nodes. However, underlay routing may be established by any protocol for multi-hop wireless networks, e.g. DYMO, BABEL or OLSR. The set of physical nodes, their wireless links to their neighbors, and the WMN routing form the *underlay* network.

We build a DHT as an overlay index structure on top of the underlay as a routing substrate. We design the system so that the physical location of nodes and the overlay structure are tightly aligned. The overlay address space is a *normalized 2D address space* between $(0, 0)$ and $(1, 1)$ to which identifiers of data items and node locations are mapped. Nodes that are responsible for adjacent regions of the DHT address space are called *overlay neighbors*. The set of overlay node identifiers and overlay neighbor relations constitutes the *overlay* network. The overlay network abstracts from the actual physical topology of the underlay and establishes a consistent store-and-retrieve infrastructure for data items.

III. Solution Characteristics

To be suitable for WMNs, a DHT needs to support the inherent nature of the underlying wireless mesh network (i.e., its dynamics and communication properties). In the following we introduce the main characteristics of a DHT approach for WMNs that traditional approaches *(a,b,c)* and approaches for mobile ad-hoc networks *(d)* do not fully account for:

a) Network Dynamics: WMNs typically expand their coverage by adding nodes at the physical boundaries of the networks. These nodes need to be integrated into the DHT address space. Similarly, leaving of nodes needs to be accounted for. Since both events alter the topology of the network in both the underlay and the overlay, a DHT has to allow for efficient handling of joining or leaving nodes on different scales.

b) Lossy Channels: In contrast to Internet DHTs, all devices share the communication channel in a WMN, resulting in unreliable communication between neighboring nodes.

c) Expensive Wireless Multi-hop Transmissions and Interference: In order to minimize the overall network load, communication should mainly occur locally, i.e. routing between adjacent identifiers in the overlay should strongly relate to routing between physical neighbors.

d) Stable Neighbors: In a WMN, the set of neighbors around a given node will be fix, save for node joins and failures. As all communication of this node will be sent over one of these neighbors, incorporating these node relations in the overlay construction by way of node identifiers keeps overlay *management traffic* local and therefore affordable in terms of transmissions and interference.

IV. A Locality-Based Overlay

Every DHT uses a mapping from a key space to an address space. The challenge for Mesh-DHT is to provide an address space design that not only allows for a consistent mapping

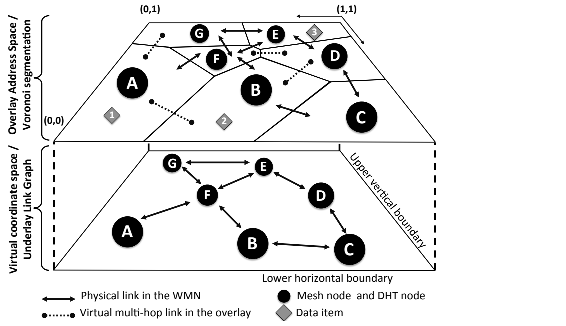

Fig. 1. The location of nodes in the overlay address space is constructed from local information available to the nodes (local link graph). Virtual multi-hop paths connect overlay nodes across voids (e.g., nodes B and D).

of regions of the address space to nodes but also keeps communication between physically close nodes. To this end, we reflect the locality of nodes in their identifiers and build the DHT identifier space on these identifiers so that the physical distance (and closeness) between nodes is accounted for in the overlay. As a general design concept, Mesh-DHT constructs the overlay in close alignment to the underlying physical links of the WMN.

We approach the problem of underlay and overlay alignment in four steps. First, we capture the physical neighbor relations between nodes. Based on these relations, we map node locations onto a two-dimensional plane. Hence, every node in the underlay has a *virtual coordinate* on this plane that reflects the relation between nodes and their neighbors (see Underlay Link Graph in Figure 1). Second, this two-dimensional plane provides the overlay address space for the DHT. Each node's coordinate in the plane maps to a singular point in the DHT address space. Next, we define how data items are mapped to this plane and which node is responsible for the item. To this end, we segment the overlay plane into regions based on the virtual node coordinates (see Overlay Address Space in Figure 1). Finally, we show how our location-aware DHT can deal with topological changes in the underlay and overlay.

A. Virtual Coordinate-based Underlay

Mesh-DHT uses virtual coordinates to represent and determine node locations and distances in the underlay. Every node and data item has a coordinate as tuple (x, y) in an Euclidean plane. The main goal is to represent nodes, that are close in the underlay, by close virtual coordinates. The Euclidean distance in the coordinate plane thus reflects the neighbor relations in the WMN.

In order to determine a node's location in the coordinate space, it assigns itself a coordinate in relation to other nodes in its vicinity. We refrain from using nodes' real-world geographic coordinates (e.g., by using GPS) as mesh nodes rarely possess GPS sensors or other means of identifying their location. Also, geographic locations alone are of limited use if the connectivity between nodes is influenced by obstacles like walls, irrelevant of the geographic vicinity of nodes.

To create a 2D coordinate embedding of the link graph, we follow the approach of Moore and Sekercioglu [6]. Direct physical neighbors attract each other, resulting in similar

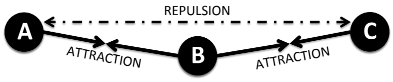

Fig. 2. Physical neighbors (solid line) attract each other, while two-hop neighbors (dotted line) repel each other to straighten the coordinate distribution out.

2D coordinates. Repulsion between two-hop neighbors causes distant coordinates, stretching the coordinate distribution (see Figure 2). Figure 3 shows an example of a 2D coordinate map of our 20-node testbed. As a result of the calculation, each node generates a (x, y) coordinate.

Nodes periodically send beacon messages with their current coordinate and the coordinates of its neighbors, allowing a local adjustment to the 2-hop vicinity. A node that joins the network assigns itself an initial coordinate close to the coordinates of its surrounding nodes and iteratively improves this coordinate. Coordinate changes influence the positions of its neighbors and may cause local topological changes as neighbors align to the new coordinate. However, given sensible thresholds for the repulsion system, changes are kept local and only affect small parts of the network.

B. Establishing a DHT Address Space

An integral part of a DHT is a consistent mapping of keys of data items and node addresses to an address space. The geometry of this address space dictates the routing and the neighbor relations in the DHT. We employ a 2D DHT address space between the low point $(0, 0)$ and the high point $(1, 1)$.

We treat the DHT address space as a rectangle that contains all assigned underlay node coordinates (c.f. Figure 1). This *bounding box* is defined by the maximum and minimum x and y coordinates of any node in the underlay coordinate space (with margins around the edge nodes). The intersection of the two lower boundaries (vertically and horizontally) represents $(0, 0)$ and the intersection of the two higher boundaries marks $(1, 1)$ in the DHT address space. Figure 1 illustrates the relation between the normalized overlay address space and the underlay coordinate space.

By treating the DHT address space as a normalized view of the underlay coordinate space, conversion between both coordinate systems becomes trivial. Nodes calculate their position in the overlay address space (u, v) by computing $(u, v) = (\frac{x-l_x}{|h_x-l_x|}, \frac{y-l_y}{|h_y-l_y|})$ with (x, y) as the node coordinate, h_x, h_y as upper boundaries of the underlay coordinate space and l_x, l_y the lower boundaries. Data items map to a fixed identifier (a, b) in the normalized address space based on their key (e.g, name) as input to a function that is independent from the underlay bounds.

To query or store data in the DHT a node issues a query to the 2D coordinate of the identifier of the data. Mesh-DHT routes these packets in a straight line between 2D coordinates in the overlay using multi-hop overlay links and direct physical links. Overlay routing translates to single-hop forwarding for direct physical neighbors and to multi-hop forwarding otherwise. We use a geographic routing protocol similar to GPSR [7], that operates on the virtual coordinates of nodes, in our implementation and evaluation.

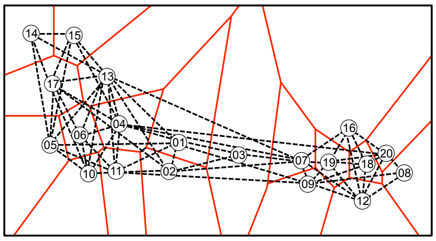

Fig. 3. A 2D underlay coordinate map and Voronoi tessellation of our 20-node physical testbed. The dashed lines are wireless links. The two clusters of nodes are nodes at several floors of the same building.

C. Voronoi Address Space Segmentation

A DHT must consistently distribute the responsibility for the address space among all nodes. We define that a node A is responsible for all data items where the address space identifiers are closer to A's than to any other node identifier in the network. This intuitive rule matches the definition of a Voronoi tessellation [8]. A Voronoi tessellation $V(p)$ is a segmentation of a metric space based on distances between a set of points p, in our case the overlay node locations. Figure 3 shows a Voronoi tessellation of our 20-node testbed.

A node can compute its Voronoi region locally *without* complete knowledge about all other node locations p in the overlay. Below, we present a distributed algorithm that takes a subset $q \subseteq p$ of the nodes in the network (the nodes closest in each direction) to compute the region. In the optimal case, q is the set of physical neighbors of a node. For example, consider node 15 (n_{15}) in Figure 3. The set of Voronoi neighbors of n_{15} $q_{15} = \{n_{13}, n_{14}, n_{17}\}$ matches its set of physical neighbors in the underlay.

However, at voids and next to address space boundaries, nodes can hold adjacent regions in the Voronoi tessellation with more distant nodes. An example for this is node n_{16} in Figure 3. In addition to its underlay neighbors $q'_{16} = \{n_{07}, n_{12}, n_{18}, n_{19}, n_{20}, \}$, it also shares a border to its region with node n_3. It is thus necessary for nodes, especially when joining the network, to consider nodes that are not physical neighbors for the construction of their Voronoi region.

A simple but efficient join procedure safely determines the relevant neighbor set q. Node A starts with a single neighbor n_0 as set q_1 and iteratively extends q_i with further candidate neighbor sets until it reaches a set of neighbors q for which any additional neighbor does not change A's region. In the following, we discuss the algorithm in detail.

We assume that A knows the coordinates of the node n_0 (e.g., due to n_0's beacon), which is responsible for the region containing the address coordinates of A. A now performs the following steps:

1) Begin with a node set only containing A ($q_0 = \{A\}$) and the resulting Voronoi tessellation $V(q_0)$. $V(q_0)$ only consists of a single region belonging to A.
2) Now extend q_0 such that the basis q_1 for the Voronoi calculation in the next step consists of A and its first neighbor set $N_0 = \{n_0\}$: $q_1 = q_0 \cup N_0$.

   The following steps are repeated, beginning with $i = 1$.

3) Compute the Voronoi tessellation $V(q_i)$ based on q_i.

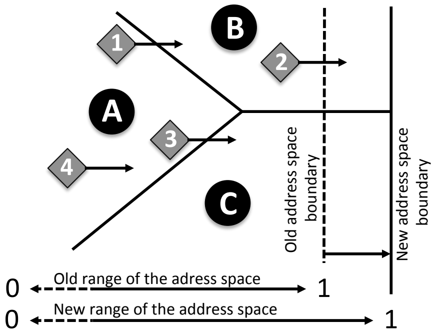

Fig. 4. When boundaries are extended in the underlay, the normalized overlay address space is stretched. This change of boundaries affects the location of nodes in the DHT address space, but not the location of data items. After moving the boundary to the right, the identifiers of data item 1 and 3 are managed by a different node. The items must be moved while data item 2 and 4 stay at the same node.

4) Determine the set of A's neighbors N_i in the Voronoi tessellation $V(q_i)$.
5) Determine the set of A's *new* neighbors N_{new} in $V(q_i)$: $N_i \cap N_{i-1}$.
6) Terminate if the set of new neighbors N_{new} is empty.
7) Query the set of new neighbors N_{new} for the neighbors in their region and combine the answers to a set r.
8) Extend the set q_{i+1} to contain the newly acquired node locations in r: $q_{i+1} = q_i \cup r$.
9) Increment i by 1 and continue with step 3.

This algorithm performs an expanding ring search around the new region while iteratively querying only new neighbors. Thus it terminates when the new region is fully enclosed and no new neighbors are found. Starting from the set of direct neighbors, the algorithm transitively expands the neighbor set by querying neighbors' neighbor sets. It thus finds every neighbor via a sequence of queries, assuming a complete segmentation to begin with.

In the example in Figure 1, A first queries F and later queries B and G for their neighbors. Since their neighbors C, D, E do not share a boundary with A, the algorithm terminates. As a worst case, a new node must query all existing nodes. However, in practice, the algorithm converges quickly and only requires querying of distant nodes for nodes next to voids in the underlay or nodes at the edges of the network. A performance evaluation of this algorithm is presented in Section V-c.

D. Dealing with Network Dynamics

In order to support joining and leaving nodes, Mesh-DHT adapts in both the underlay and the overlay to reflect the resulting network topology changes. Once a node joins, it assigns itself a new underlay coordinate. We assume the node to select a location outside the current boundaries to illustrate how Mesh-DHT incorporates topology changes. At the overlay level, the bounding box adjusts the conflicting side(s) of the rectangle to accommodate the location of the new node. This expands the rectangle to provide additional space for the Voronoi region of the new node.

To quickly establish a consistent view on the overlay address space, we flood a notification about this change. With this

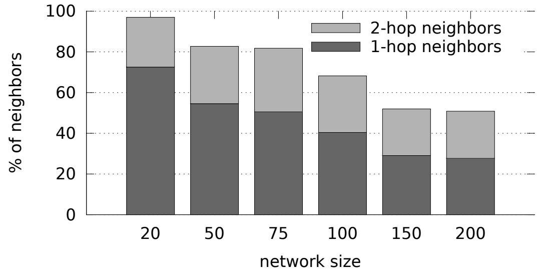

Fig. 5. The fraction of local neighbors in the set of more distant overlay neighbors approaches 50% in large networks with large diameters. Networks with small diameters exhibit a larger fraction of close neighbors.

adjustment, the mapping from the coordinate space to the normalized address space changes, requiring data items to move to new nodes (see Figure 4). However, these changes in the underlay coordinate distribution are locally bounded and only require to move items between local neighbors. We construct new boundaries with an additional margin so that new nodes can join at the edges without further adjustments.

Nodes may fail or leave the network, requiring its managed region of the address space to be re-distributed to other nodes. Once a node A detects the failure of an overlay neighbor B, it queries its own Voronoi neighbors for their neighbor information and re-calculates its address space region. Following the same approach as in the join process, A performs an expanding ring search around the failed region and, with the relevant node locations q, re-calculates its own Voronoi region. As all overlay neighbors of the failed node follow this approach, the vacant address space region is consistently re-distributed.

V. EVALUATION

To evaluate our approach and to show its scalability, we implemented Mesh-DHT for the OMNeT++ network simulator [9] and for a Linux-based, 20 node real-world 802.11 wireless mesh testbed [10]. The testbed is spread over three floors of our institute building, consists of 20 ALIX 2c2 and ALIX 3c2 wireless routers, and allows us to make statements about the real-world applicability of Mesh-DHT. The OMNeT++ simulation allows us to evaluate large-scale network topologies. The measurements for the 20-node testbed and a 20-node simulation lead to highly similar results at the overlay level. We provide results for both systems. In this section, we evaluate the four main characteristics of Mesh-DHT.

a) Locality of Voronoi Neighbors: The emphasis of Mesh-DHT lies on the preservation of neighbor locality in the overlay. Figure 5 shows how well our approach preserves the underlay neighbor-relation by illustrating the fraction of Voronoi neighbors in the overlay that are physical neighbors or 2-hop neighbors in the underlay.

In small networks of up to 75 nodes, in which the network diameter is small, more than 80% of the overlay neighbors are underlay neighbors. Of these, more than 50% are physical (1-hop) neighbors. With a growing network size, the fraction of distant overlay neighbors approaches 50%. The reason for this decline are voids in the underlay that exist because we placed new nodes at the boundaries of the network instead

TABLE I
VORONOI NEIGHBORS (NODE DEGREE)

Network size		Average	Minimum	Maximum
20	(testbed)	4.61	1	9
20	(simulation)	4.77	2	10
50	(simulation)	5.30	1	10
75	(simulation)	5.53	1	10
100	(simulation)	5.66	1	10
150	(simulation)	5.74	1	11
200	(simulation)	5.75	1	11

TABLE II
NUMBER OF MESSAGES PER NODE JOIN

Network size		Average	Minimum	Maximum
20	(testbed)	1.40	0.17	4.67
20	(simulation)	1.47	0.13	4.50

of creating dense networks. However, a fraction of 50% still represents a balanced mixture between local nodes and distant neighbors. This mixture enables nodes to choose between small routing steps over local neighbors and large routing steps in the overlay over distant neighbors.

b) Number of Voronoi Neighbors: The number of Voronoi neighbors determines the overhead for the establishment and maintenance of a region when nodes join or fail. Table I shows the number of Voronoi neighbors derived from 20 independent simulation runs per network size and 20 testbed measurement runs. The average number of Voronoi neighbors (the DHT node degree) grows very slowly and even stagnates for larger networks. Thus, in networks that grow geographically as well as in the number of nodes, the physical node neighbor degrees as well as the Voronoi neighbor degrees are only slightly affected by new nodes. Maintenance operations that depend on a node's neighbor degree thus cause similar overhead regardless of the network size.

c) Overhead for Joining the Network: A joining node iteratively queries the set of neighbors around its position, expanding this search until all relevant neighbors are found. We show the practicality and efficiency of this algorithm by comparing the number of queries a node sends and the resulting number of *new* neighbors. Table II shows results from our 20 node real-world testbed and a simulation of a 20 node network, indicating that results gathered from simulation are representative for results in real-world networks of similar size. Figure 6 shows the efficiency of our approach with regard to different simulated network sizes. Irrelevant of the network size, the average overhead to join the network correlates nearly 1:1 with the number of resulting overlay neighbors. As the average node degree is almost identical regardless of the network size (cf. Tab. I), this shows the scalability of the overlay operations. We explain the outliers in Figure 6 with regard to the topology of the network: Nodes that query distant nodes to form their Voronoi segmentation, query multiple sets of neighbors during this task.

d) Robustness in Case of Node Failures: Node failures require the remaining nodes to re-distribute the vacant address space region. Figure 7 shows the average number of messages that a remaining node sends to determine its new region after a failure of a neighbor. Again, the number of messages is correlated to the number of previously adjacent nodes.

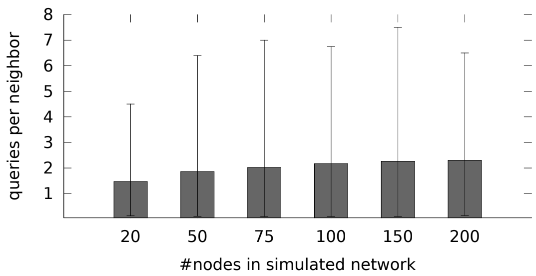

Fig. 6. The overhead for joining the network is strongly correlated with the number of overlay neighbors in the resulting Voronoi region. The gray bars depict the average quotient of the queries a node sends and the resulting number of overlay neighbors.

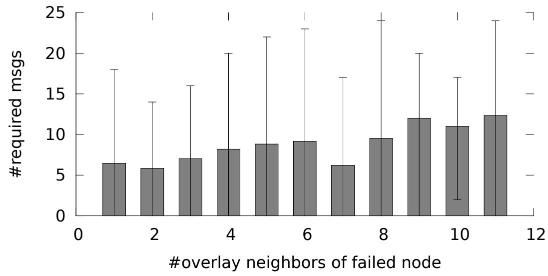

Fig. 7. The average number of messages a node sends to re-distribute the vacant address space region is directly correlated to the number of adjacent nodes of this region.

Regions with a very small number of neighbors (less than four), will likely be distributed among more nodes than before, explaining the relatively high average number of messages in this case. Conversely, redistributing regions with very large numbers of overlay neighbors requires only relatively few messages, because many nodes will notice the failure and redistribute the region among them. The first node to detect the failure of an overlay neighbor sends a disproportionately large number of messages because it notifies the remaining neighbors, which may immediately use this notification to re-calculate their regions. Consequently, nodes that detect the failure later send less messages because of the knowledge gathered from incoming messages. This explains the variance in Figure 7.

VI. RELATED WORK

Approaches related to our work can be grouped into two categories: *i)* Traditional DHT approaches and *ii)* DHT approaches for multi-hop wireless networks.

Traditional DHT approaches, such as Chord [1] and CAN [2] are designed for Internet communication. This design for a reliable and fast delivery of packets makes traditional DHT approaches unsuitable for wireless multi-hop networks. To illustrate, [3] and [11] evaluate Chord in wireless multi-hop networks. Several characteristics distinguish these designs from our approach. The assignment of random identifiers and strict, logical routing schemes and address space segmentation rules prevent the overlay from benefitting from node locality.

DHT approaches for wireless multi-hop networks either place a DHT on top of a given multi-hop routing protocol or integrate the DHT directly into the routing scheme.

In [5], Ratnasamy et al. construct a Geographic Hash Table

(GHT) for wireless sensor networks. Using GPSR as the underlying routing protocol, a node manages the keys of data items that are closer to its position than to any other node. GHT relies on GPSR's *perimeter mode* to to establish an implicit assignments of data items to the closest node. Furthermore, GHT requires nodes to know their exact physical positions, rather than a position relative to their neighbors. We assume WMNs to be deployed in rural scenarios and buildings in which walls or other obstacles influence communication between nodes. In this scenario, nodes may be close to each other but may share a bad link because of these obstacles and attenuation. In such cases, the real-world distance between nodes does not serve well as a routing metric, as a low-quality link between close nodes would be frequently used. Thus, instead of geographic locations, we prefer to use virtual coordinates based on link quality, connectivity, and hop distance as a metric for the distance between two nodes.

MeshCord [3], MADPastry [12] and Ekta [4] integrate traditional DHT approaches into multi-hop wireless networks. MeshChord employs locality of nodes to position nearby nodes close in the Chord identifier ring, requiring each node to know its exact location, e.g. via GPS. Furthermore, "the overhead for overlay maintenance is considerable and, in some cases, can lead to network congestion." [3]. The advantage of MeshCord over pure Chord decreases under network dynamics.

Ekta and MADPastry combine Pastry [13] with Dynamic Source Routing (DSR) and Ad-Hoc on Demand Vector Routing (AODV), respectively. Ekta requires unique IPs to hash to DHT identifiers. This arbitrary placement of identifiers in the DHT address space results in arbitrarily long underlay routes for single overlay hops. MADPastry introduces prefix clustering of nodes in the overlay to group physically close nodes. Next to the additional overhead of cluster and landmark management, MADPastry requires nodes to manage an AODV routing table, leaf and node sets in Pastry and an overlay ID lookup via arbitrary nodes in the network.

Virtual Ring Routing (VRR) [14] establishes a DHT substrate by randomly placing nodes in a Chord-like ring overlay. Multi-hop routing is done by forwarding to the overlay neighbor with an identifier closest to the destination, irrelevant of the physical proximity of this node. Thus, one hop in the overlay may result in long routes in the underlay. Furthermore, the random nature of identifiers does not allow for provision of services that exploit the locality of clients and routers as locality is not reflected in the overlay.

In the Virtual Cord Protocol [15], node identifiers in a normalized $[0, 1]$ identifier space are assigned based on the proximity of nodes. The first two nodes occupy the upper and lower ends of the address space, every new node positions itself between the node positions in its proximity. This results in an unbalanced distribution of the address space if many nodes join close to each other.

VII. Conclusion

In this paper, we presented the design of the overlay address space and overlay segmentation in Mesh-DHT as an approach to provide a fully decentralized look-up structure for wireless mesh networks. This look-up structure provides a means of handling client mobility, data management, name resolution and service discovery. It realizes the look-up functionality in a fully decentralized manner and thus scales well for large network sizes. Mesh-DHT preserves the physical location and relation between nodes in the WMN in the DHT overlay address structure. This reduces multi-hop transmissions needed for periodic maintenance and provides directed routing towards data items. We thereby reduce the overall number of transmissions and load on the wireless medium, reserving the medium for client and application traffic. We implemented Mesh-DHT for our real-world IEEE 802.11 mesh network and for the OMNeT++ network simulator to show the performance of the system under realistic conditions and in large-scale scenarios. Mesh-DHT can be extended to provide additional services on top of the basic index structure, for example resilient data storage by using redundant storage locations, or local service provision by encoding location information into the key mapping.

References

[1] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," in *Proceedings of the ACM SIGCOMM '01*, San Diego, 2001.

[2] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content-addressable network," in *Proceedings of the ACM SIGCOMM '01 Conference*, San Diego, 2001.

[3] S. Burresi, C. Canali, M. E. Renda, and P. Santi, "Meshchord: A location-aware, cross-layer specialization of chord for wireless mesh networks," in *PerCom*, 2008.

[4] H. Pucha, S. M. Das, and Y. C. Hu, "Ekta: An efficient dht substrate for distributed applications in mobile ad hoc networks," *IEEE Workshop on Mobile Computing Systems and Applications*, vol. 0, 2004.

[5] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker, "Ght: a geographic hash table for data-centric storage," in *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, ser. WSNA '02, 2002.

[6] N. Moore, A. Sekercioglu, and G. K. Egan, "Virtual localization for mesh network routing," in *Proc. Network and Communication Systems (NCS 2005)*, 2005.

[7] B. Karp and H. T. Kung, "Gpsr: Greedy perimeter stateless routing for wireless networks," in *Proc. 6th Annual International Conference on Mobile Computing and Networking (MobiCom 2000)*, 2000.

[8] F. Aurenhammer, "Voronoi diagrams, a survey of a fundamental geometric data structure," *ACM Comput. Surv.*, 1991.

[9] A. Varga and R. Hornig, "An overview of the omnet++ simulation environment." in *SimuTools*, 2008.

[10] "Umic-mesh network," last visited on 24.04.2011. [Online]. Available: http://www.umic-mesh.net

[11] C. Cramer and T. Fuhrmann, "Performance evaluation of chord in mobile ad hoc networks," in *Proceedings of the 1st international workshop on Decentralized resource sharing in mobile computing and networking*, ser. MobiShare '06, 2006, pp. 48–53.

[12] T. Zahn and J. Schiller, "MADPastry: A DHT Substrate for Practicably Sized MANETs," in *Proc. of 5th Workshop on Applications and Services in Wireless Networks (ASWN2005)*, 2005.

[13] A. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems," in *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, 2001.

[14] M. Caesar, M. Castro, E. B. Nightingale, G. O'Shea, and A. Rowstron, "Virtual ring routing: Network routing inspired by dhts," *SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 4, 2006.

[15] A. Awad, C. Sommer, R. German, and F. Dressler, "Virtual cord protocol (VCP): A flexible DHT-like routing service for sensor networks," in *Proceedings of the 5th IEEE International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, 2008.