A Churn and Mobility Resistant Approach for DHTs

Olaf Landsiedel, Stefan Götz, Klaus Wehrle
Distributed Systems Group
RWTH Aachen, Germany
firstname.lastname@cs.rwth-aachen.de

ABSTRACT

Mobile ad-hoc networks (MANETs) and distributed hash-tables (DHTs) share key characteristics in terms of self organization, decentralization, redundancy requirements, and limited infrastructure. However, node mobility and the continually changing physical topology pose a special challenge to scalability and the design of a DHT for mobile ad-hoc network. The mobile hash-table (MHT) [9] addresses this challenge by mapping a data item to a path through the environment. In contrast to existing DHTs, MHT does not to maintain routing tables and thereby can be used in networks with highly dynamic topologies. Thus, in mobile environments it stores data items with low maintenance overhead on the moving nodes and allows the MHT to scale up to several ten thousands of nodes.

This paper addresses the problem of churn in mobile hash tables. Similar to Internet based peer-to-peer systems a deployed mobile hash table suffers from suddenly leaving nodes and the need to recover lost data items. We evaluate how redundancy and recovery technique used in the internet domain can be deployed in the mobile hash table. Furthermore, we show that these redundancy techniques can greatly benefit from the local broadcast properties of typical mobile ad-hoc networks.

Categories and Subject Descriptors

D.2.4 [**Distributed Systems**]: Distributed applications

General Terms

Algorithms

Keywords

Peer-to-peer, distributed hashtable, mobility, ad-hoc network, DHT

1. INTRODUCTION

Peer-to-peer networking has changed the way to store data distributed in a network. Peer-to-peer networks are self maintaining, resilient, and only need limited infrastructure and control. The development of structured peer-to-peer networks, e.g. DHTs, extends these ideas to high scalability, increased resilience and flat hierarchies.

Structured peer-to-peer networking provides a number of key properties enabling efficient data access in mobile ad-hoc networks: (1) commonly, ad-hoc networks have limited or even no infrastructure. Thus, a fully distributed and hierarchy-less substrate – such as a DHT – is required for efficient data storage and access. (2) The high scalability of DHTs enables large mobile crowds. (3) Furthermore, the fragile ad-hoc network can benefit strongly from the redundancy and resilience provided by structured peer-to-peer networks.

However, the random movement of nodes in a mobile ad-hoc network makes it challenging to deploy a structured peer-to-peer network. Thus, until recently the node movement itself was considered churn as the topology and routes of the ad-hoc networks consistently change. The mobile hash table (MHT) addresses these challenges and forms a substrate for scalable mobile peer-to-peer networking.

DHTs map data items, i.e. key-value pairs, on node IDs. Commonly, a key is computed via a hash function from a string describing the corresponding data item. And a node ID is derived from its IP-address [22, 17] or its geographic position [18]. MHT – in contrast – introduces semantics to the keys: it derives a geographic position, direction, and speed from the key of a data item and stores this item on the node which matches these properties best. Thus, a data item is assigned a path along which it moves by being stored on a node moving along a similar path.

Nonetheless, churn, i.e. nodes leaving the network unexpectedly, is still a challenge to the MHT's integrity and resilience. Typically, churn is handled by adding multiple copies of a data item to the DHT. The nodes holding these copies exchange messages at some interval to ensure that the data items are still available. When a node has left the network, i.e. the ping fails, the new node responsible for the data item is determined and a copy placed on it.

In this paper we evaluate how churn can be handled in the mobile environment. The mobility and wireless communication do not allow to adapt the techniques used in the Internet based DHTs blindly. For example, when placing the copies MHT can greatly benefit from the local broadcast properties of the wireless ad-hoc network. The copies are placed in the communication range of the current data item's node. As a result the necessary ping-pong messages between the original data item and its copies are only local broadcasts and do not add an additional load on the routing system of the ad-hoc network.

The remaining paper is structured as follows: Section 2 discusses the limitations of current mobile peer-to-peer technologies and introduces techniques used in Internet based DHTs to handle Churn. Section 3 discusses the MHT design. Section 4 presents the use of redundancy technique in the mobile environment to handle churn. Section 5 evaluates the performance of MHTs under churn and section 6 concludes and discusses ongoing work.

2. RELATED WORK

Peer-to-peer communication has had a large impact in the Internet research community. Various structured [3, 5, 17, 22] and unstructured protocols – such as the well known file sharing tools – have been presented. The peer-to-peer paradigm has been extended to ad-hoc networks and even sensor networks [1, 18, 10, 21].

Although their high scalability, resilience, and flat hierarchies make DHTs an interesting substrate for mobile networking, only a very limited number of approaches base on this principle. In this section, we discuss mobile peer-to-peer systems and their shortcomings and compare our work to them.

Most mobile [6, 16, 24], peer-to-peer approaches deploy a structured peer-to-peer network on top a ad-hoc routing protocol, such as AODV [15] or DSR [7]. Thus, these approaches require the underlying ad-hoc routing to frequently set up and maintain routes to all entries in the DHT routing table. The routing protocols use flooding for route discovery, resulting in a limited scalability. Furthermore, the DHT routing is not aware of the underlying topology, resulting in high routing and maintenance overhead. Additionally, all of today's DHTs require frequent discovery messages to test whether their routing entries are still valid. In comparison, our mobile hash table does not depend on this mechanism.

Orion [13] deploys an unstructured peer-to-peer network on top of a ad-hoc routing protocol. Thus, it suffers from the limited scalability of ad-hoc routing protocols and the limited scalability of unstructured networks as both layers strongly rely on information flooding. Additionally, it does not use cross-layer optimization techniques so both layers flood the network without being aware of each other.

Although not built for node mobility, the geographic hash table (GHT) [18] is probably the concept most similar to our approach. GHT maps a key associated with a data item to a geographic location. Geographic routing is used to store and retrieve a the data item at its location. We generalize the ideas presented in this work to support mobile nodes.

The multi-level peer index (MPI) [12] extends the GHT approach from providing a specific geographic location to a spatial area. However, for high node mobility it requires location updates to be distributed in the entire network which severely limits the scalability of this approach.

Recently, Virtual Ring Routing (VRR) [1] has been proposed as a scalable routing protocol and DHT substrate for ad-hoc networks. In contrast to MHT, VRR does not rely on geographical position information. However, VRR needs to maintain and repair the routes to its virtual neighbors making VRR – from our point of view – not a ideal candidate for networks with highly dynamic topologies.

We are not aware of any research that looks at churn in DHTs deployed in mobile environments. However, churn in Internet based peer-to-peer systems – unstructured and structured ones – received strong attention in the recent years. For example, references [2, 4, 11, 14, 20] evaluate the stability of DHTs like Chord [22] and Pastry [3] under churn and present techniques to increase stability. However, this work focuses on the routing tables of DHT which is not applicable to MHT. Due to the geographic placement of data items MHT's routing tables do not suffer from churn. Nonetheless, the availability of data items still suffers from churn in MHTs. Thus, in this paper we evaluate how techniques proposed in the Internet domain for repairing routing tables under churn can be applied to MHT.

3. INTRODUCING MOBILE HASH TABLES

In this section we describe the design of mobile hash tables. The main challenge for structured mobile peer-to-peer networking is to apply a structure to the unstructured and seemingly random node movements. Assuming that each node knows its position, speed, and direction, we show how a structured DHT can be set up.

To map data onto the moving nodes, we use following scheme: for each data item we derive a path from its key. A data item moves along its path and is stored on the node which moves on the most similar path. A path consists of two points, between which a data item moves back and forth, and speed information. As the path of each data item is derived from its key and the path is a loop, one can compute the position of a data item at every point in time. Thus, queries can determine a data item's position and be routed to it. In this paper we use a simple path consisting of two points, e.g. we derive the x and y coordinates of these two points from the key. The approach also allows for more complex mapping functions, e.g. a rectangular path, were the points of the rectangle are derived from the key. Alternatively, a key can denote multiple points in space from which a spline curve can be derived.

By comparing position, direction, and speed, it is determined which node carries a data item. The scalability of the proposed approach bases on the following observation: the more nodes are in an environment, the higher is the probability that a node with a path and speed similar to the path of the data item exists. Thus, the more nodes, the better matches exist. As result, data needs to be moved between the carrying nodes less frequently.

3.1 Routing in a MHT

Mobile hash tables are built on top of GPSR [8], a geographic routing algorithm for multi-hop wireless networks. We now briefly discuss design features of GPSR relevant for our work and then propose a minor extension to GPSR to integrate the mobile hash tables.

GPSR is highly scalable, as its routing only depends on local knowledge. In GPSR, packets are routed geographically, i.e. based on physical positions of packets and nodes. Packets to be routed are marked with their destination. Furthermore, each node knows its own position and those of its immediate neighbors. GPSR uses this local knowledge to route packets to their final destination. In its default operation mode, GPSR forwards packets greedily. Greedy forwarding fails, when a node has no neighbor closer to the final destination: the packet has reached a local maximum, e.g. a void. In this case, GPSR switches to perimeter forwarding and routes packets with the right-hand-rule around network voids. GPSR returns from perimeter routing to greedy forwarding when it reaches a node closer to the destination than the one at which it switched to perimeter routing (its position was stored in the packet).

3.2 Extension of GPSR

In GPSR, each node knows the position of its neighbors in one-hop distance and uses this information for its local routing decisions. In practice, each node regularly announces its position to the surrounding nodes with local broadcast messages. We extend this announcement by the current speed and the direction the node moves at. These values can be easily derived from GPS position samples. In MHT, we use this extended information to find a node which has a position, speed, and direction similar to the path of a data item.

3.3 Joining and Leaving an MHT

MHT bases on wireless communication so it benefits from its local broadcast properties. There is no need to find the ID space

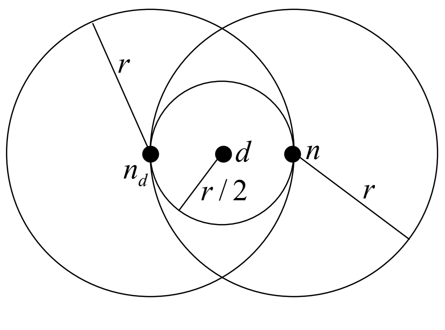


Figure 1: As long as the data item d is stored not farther away from its position than $r/2$, e.g. node n_d in the figure, a node n can reach node n_d when it comes into $r/2$ of the data d.

a node is responsible for via a search in the DHT as commonly in Internet-based systems. For mobile hash tables, it is sufficient that a joining node starts the regular local broadcast of its position, speed, and direction. Thus, its surrounding nodes recognize its existence and will consider this node for their routing and storage decisions.

To leave the system, the leaving node stops sending regular local broadcasts. Consequently, its surrounding nodes stop using it for routing or storage. In such a passive leave, the node does not announce its departure to its surroundings. However, the DHT's routing tables stays consistent after such a passive leave, as all decisions are based on local knowledge. No repair algorithm as in most Internet based DHTs – such as fixing finger tables in Chord – is necessary.

However, nodes may still try to route data via this node until their knowledge about this nodes times out. Furthermore, all data stored on this node and all messages it might have been forwarding at this moment are lost. Thus, next to the passive leave MHT supports a so-called active leave, the node announces its leave to its neighbors. This ensures that the leaving node is not considered for routing and storage anymore. Furthermore, it forwards all pending messages to their next hop and stores all data on the now best matching participant.

3.4 Data Placement

Before discussing lookups and data placement, we explain data movement in MHTs. Thus, for now we assume that a data item d is stored on a node n_d and we discuss how and when it is moved to another node n_{next}. Furthermore, we discuss how this node n_{next} is selected.

For simplicity's sake, we assume a circular communication range, e.g. a unit disk model, and that all nodes have the same communication range[1]. Let r be the communication range of a node, then a data item d needs to be stored on a node n_d not farther away than $r/2$ from the data item's position p_d – the position p_d is determined by the key which describes the data item d. This ensures that a node n in $r/2$ distance from p_d can communicate to n_d and retrieve the data item d (see figure 1). Let p_{n_d} be the position of the node n_d. Thus, when $|p_d - p_{n_d}| > r/2$, the data item d needs to be moved to another node to ensure that queries can successfully find the data item.

As the data item d has to be stored not farther away than $r/2$ from its position p_d, n_d selects the node n_{next} from its surrounding nodes. Since all nodes frequently announce their positions, directions, and speeds, to their neighbors, no explicit communication is necessary; n_d does a lookup in its neighborhood table. Among its neighbors, it selects the one node n_{next}, for which $|p_d - p_{n_{next}}| > r/2$ holds for the longest time. It uses the current speed and direction of data and nodes to predict future positions. When no node in range fullfills these requirements the data item is stored on the node which is closest to the data item's position. Thus, it maybe temporarily not reachable when this node is farther away than $r/2$ from the data item's position. In section 5 we evaluate the probability that a data item is our of place and thereby temporarily not reachable.

After discussing how the node which stores a data item is selected, data placement in the MHT is straightforward. A new data item is forwarded from its source to a node which is not farther away than $r/2$ from the data item's position. This node then determines the node to store the data item the same way that a new node for storage is selected.

3.5 Data Lookup

Data lookup, i.e. queries, use the same technique as the above described data placement. Knowing the data items key, the query compute the item's position. Thus, the query is forwarded from its source to a node which is not farther away than $r/2$ from the data item's position. A local broadcast from this node reaches the node carrying the requested data item, as the item itself is always on a node in $r/2$ or less distance from its position.

Although the data item can be found easily, sending the reply back to its source is not as trivial because the source moves along its own path. To find the source, the query and its reply store the position, direction, speed and ID of the source. Thus, nodes forwarding the reply can determine the source position at any time. However, the source might change its direction or speed at any time. When this happens, the source places a new (temporary) data item – a buoy – in the MHT. It is placed at the current position of the source and moves with the old direction and old speed of the source. Furthermore, it stores the new direction and speed of the source. Thus, the reply reaches the buoy instead of the the source and retrieves the new direction and speed of the source. Multiple direction changes are handled by chains of buoy.

In contrast to existing internet based or ad-hoc network DHTs (compare section 2) MHT does not maintain a routing or finger table to nodes which are multiple hops away. This property – next to the mapping of data items on paths – enables MHT's scalability in highly mobile scenarios.

3.6 Realistic Mobility

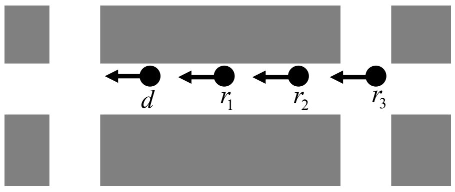
MHT focuses on systems with a high degree of node mobility. In the real world, nodes move along a limited set of paths, e.g. the roads of city. Thus, instead of arbitrary paths, we derive paths along the roads of a city from the data item's keys. In our work, we use the Manhattan grid as an example. As a result, the space and direction where nodes and data items move become strongly correlated. Thus, the chance of matching data and node paths increases significantly and MHT provides better performance.

4. CHURN AND LOAD BALANCING

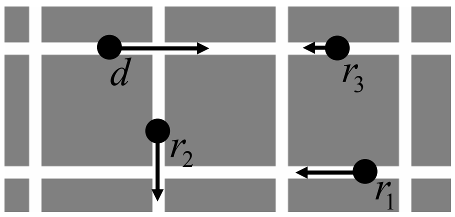
When a node unexpectedly leaves the network all data items stored on it get lost. The standard technique to address this churn is to store copies of a data item on various nodes in the peer-to-peer network. These data items frequently exchange ping-pong messages to test for their for availability.

Obviously, such an approach sounds promising for MANETs, too. However, the local broadcast properties of the wireless com-

[1]Please note that by introducing a factor α to the communication range, we can easily model heterogeneous communication ranges and non unit-disk models.



(a) Local Redundancy: adding replicas with the same path, but slightly different starting points.



(b) Global Redundancy: adding replicas with uncorrelated paths.

Figure 2: Local vs. global replication in the Manhattan grid.

munication open up two interesting design choices: (1) local replication and (2) global replication:

4.1 Handling churn with local replication

MHT uses local replication to handle Churn. It places replicas of each data item close to their positions. Each replica is placed with a constant offset *o* to the original data item in the system. This offset is added to the start and end point – or the center point in case of traffic adaptation – of the data item. Thus, replicas and original data items move on the same path, i.e. the same direction and speed, just slightly apart from each other (see figure 2(a)).

When selecting the offset, one has to take care of two factors: (1) the offset shall be large enough to ensure that the two copies of the data item are not placed on the same node. (2) When the offset is too large, the distance between two copies becomes increases and the ”still-alive” messages need to be routed over multiple hops. The result is communication overhead. Thus, we select the offset so that a replica can allways contact the previous and the next copy directly.

When an item is lost, a copy with the corresponding path information is created and placed in the hash table. As all items are close to each other, the “still alive” messages result in low overhead as they only need to be transmitted via a limited number of hops. For the same reason, updates to a data item have limited overhead. Local replication allows MHT to efficiently deal with sudden node death or departure.

4.2 Load balancing with global replication

Nonetheless, local replication has some limitations. When a data item is very popular, queries can cause a high load on the routes to the data item. Global replication addresses this problem by placing a data item at various unrelated places in the network (see figure 2(b)). This ensures better load balancing and furthermore ensures that the network stays alive even when parts get disconnected. For global replication however, “still alive” messages need to be sent across large parts of the network and so increase the message overhead drastically.

Concluding, global both replication techniques have different trade-offs, which depend on the deployment scenario. In this paper we focus on local replication to handle churn.

4.3 Options for local replication

When adding local replication to mobile hash-tables we identify two import factors next to the offset itself:

- Level of redundancy: Obviously, adding more copies to the system increases the redundancy. Although the nodes that carry the replicas are in one hop distance from each other and therefore ”still alive” messages are one hop messages, the number of exchanged messages increases with a higher redundancy level. Basically, these messages add a constant load to the communication system. Thus, the number of replicas impacts the query and data item maintenance capabilities of the MHT. Although the level of redundancy depends on highly on the application scenario, we want evaluate the trade offs of the different redundancy levels in this paper.

- ”Still alive” message frequency: The second important factor is the frequency of the ”still alive” messages. Obviously, while one copy of a data item is lost (and not restored yet) queries to this copy will fail. Furthermore, when all copies are lost before recovery, the data item will not be available anymore. Exchanging these ”still alive” messages frequently decreases the chance of loosing a data item from the system but increases the message load. Thus, next to the redundancy level we evaluate the frequency of the ”still alive” messages in this paper.

5. EVALUATION

After discussing the MHT design and its features to handle churn, we evaluate the proposed technique. First we present our simulation setup and then discuss simulation results.

5.1 Simulation Model

We implemented the mobile hash table in the OmNet++ simulator [23] to evaluate the MHT performance and scalability. The mobility scenarios are based on the “random waypoint” model [7, 19]. When not denoted differently, between 5000 and 100000 nodes move with a speed uniformly distributed between 10 and 15 m/s in an area of 2000m x 2000m for the scalability simulations and an area of 1000m x 1000m for the simulations regarding churn. The wireless radio has a transmission range of 100m and its propagation bases on the unit-disk model. The simulation duration is 1000 to 3600s and results are averaged over three runs. The offset for replication is set to 50m.

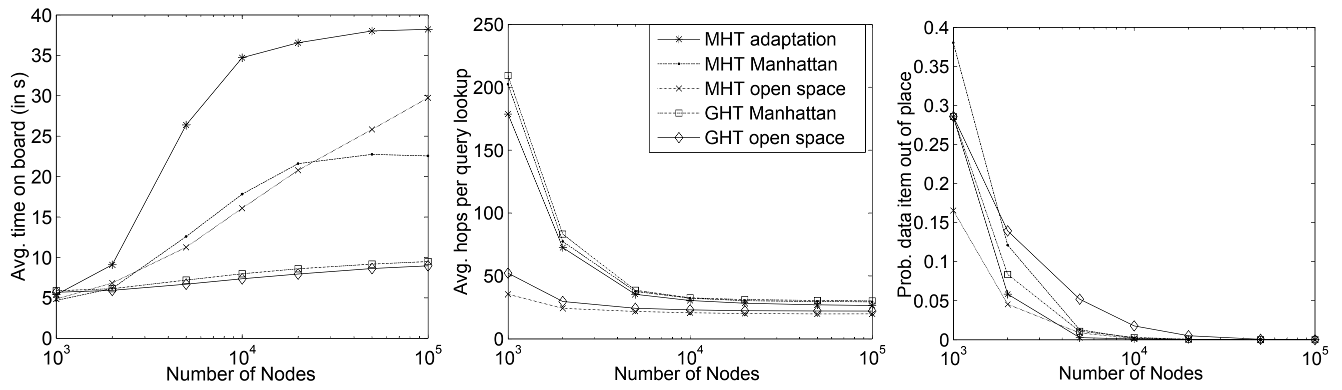
Our simulation model ignores the capacity of, and the congestion in the network and packet loss. While these assumptions are obviously unrealistic, they allow the simulator to scale to tens of thousands of nodes and us to evaluate the scalability of the proposed approach.

We evaluate the following performance metrics:

- Maintenance overhead: this metric evaluates how long a data item is stored on a node until their paths do not match anymore and the data item is moved to another node. This metric is the key metric of MHT, as it describes its scalability.

- Path length: this metric evaluates the hops it takes to resolve a query in the MHT.

- Data item out of place: when a data item needs to move to another node and there is no applicable node with similar path properties in range, the data item can move away from its path. Thus, queries fail until the data item is back on an appropriate route.



(a) Maintenance overhead: average duration a data item is stored on a node.

(b) Query Lookup: number of hops to resolve a query.

(c) Probability that a data item is out of place when queried.

Figure 3: Evaluating the performance of MHT for varying node numbers. For comparison GHT performance is depicted, too. The playground has a size of 2000m x 2000m. Please note the logarithmic scale.

- Loss rate of data items: when a node leaves the network unexpectedly, i.e. churn, all data items stored on it get lost. Redundant storage of items in the DHT can drastically reduce the impact of churn.

5.2 Performance Results

Scalability – Varying Number of Nodes

The results for varying the number of nodes in the system are depicted in figure 3. With the raising number of nodes the node density increases. Thus, the probability increases that a data item finds a node with similar speed and direction to be stored on. As figure 3(a) depicts, the average time a MHT data item is stored on a node raises with the increasing number of nodes. Consequently, the more nodes participate in a system, the lower the maintenance overhead is. This is a very interesting system property, as MHT – in contrast to most other systems, including GHT – scales inverse with the number of nodes. GHT does not show these scaling properties as it does not benefit from node movement.

Furthermore, figure 3(b) shows that the MHT approach – and particularly not the adaptation mode – does not impact data lookup. The average number of hops to resolve a query of MHT is nearly equal to the hops of GHT. Figure 3(c) depicts the probability that a data item is stored farther away from its position than half the transmission range when queried and so cannot be retrieved via lookups. The figure shows that commonly for MHT this probability is lower than for GHT. GHT addresses this problem, by storing the data item on all nodes surrounding the respective area.

Churn resilience

The results for the MHT's churn resilience are depicted in figure 4. High churn rates, i.e. a low average duration of participation in the network, results in high loss rates of data items. Figure 4(a) depicts that for high churn rates a high level of data replication can reduce the loss rate of data items drastically. Additionally, figure 4(b) depicts that short intervals for the exchange of "still alive" messages can further reduce the impact of churn. Figure 4(c) shows that a high frequency of "still alive" messages and redundancy can reduce the data loss to roughly 100 data items per hour, i.e. less than 0.3% per hour. Concluding the evaluation of MHT's churn resilience, it can be said that a high ping frequency and a high number of replicas reduce the chance for data loss drastically, especially when nodes join the network just for short durations. Nonetheless, both approaches introduce maintenance overhead and thereby additional message load. However, due to local replication, the messages are only exchanged between physical neighbors.

6. CONCLUSION

MHT addresses two key problems of mobile peer-to-peer networking: efficient data lookup and scalable routing in a mobile environment. Our simulation results validate the scalability of the design – in a network with 100000 nodes it supports efficient data lookup and has low DHT maintenance overhead. Furthermore, it shows that MHTs can efficiently use local replication to handle churn.

Currently, we are implementing MHT on top of a more detailed simulation models to add packets loss and channel capacity to our evaluation. Furthermore, we evaluate load balancing with global replication and how it can interact with local replication.

In this paper, we presented a scalable approach to structured peer-to-peer networking in mobile environments and evaluated its churn resilience.

7. REFERENCES

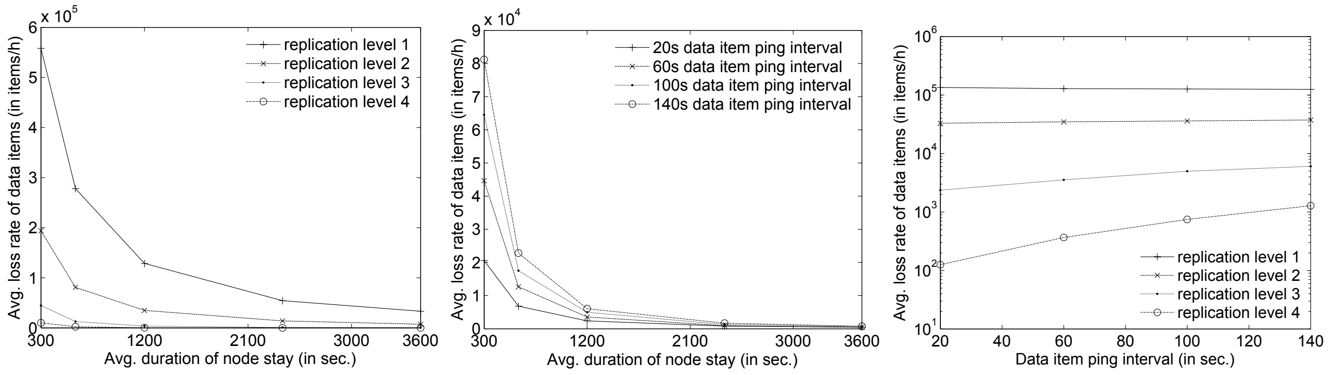
[1] M. Caesar, M. Castro, E. Nightingale, G. O'Shea, and A. Rowstron. Virtual Ring Routing: Network routing inspired by DHTs. In *Proc. of the conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM)*, 2006.

[2] M. Castro, M. Costa, and A. Rowstron. Performance and Dependability of Structured Peer-to-Peer Overlays. *dsn*, 2004.

[3] M. Castro, P. Druschel, Y. C. Hu, and A. Rowstron. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *Proc. of IFIP/ACM International Confernence on Distributed Systems Platforms*, November 2001.

[4] K. Gummadi, R. Gummadi, S. Gribble, S. Ratnasamy, S. Shenker, and I.Stoica. The impact of DHT routing geometry on resilience and proximity. In *Proc. of the conference on Applications, Technologies, Architectures, and protocols for computer communications (SIGCOMM)*, 2003.

[5] N. Harvey, M. B. Jones, S. Saroiu, M. Theimer, and A. Wolman. Skipnet: A scalable overlay network with



(a) Churn resilience for various replication levels and node participation in the network. The "still alive" message interval is kept constant at 60s.

(b) Churn resilience for various "still alive" message intervals and node participation in the network. The redundancy level is kept constant at 3.

(c) Churn resilience for various replication levels and "still alive" message intervals. The avg. node participation is kept constant at 1200s.

Figure 4: Evaluating churn resilience. The number of nodes in simulation is 5000 and it contains 50000 data items, nodes move in the Manhattan grid.

practical locality properties. In *Proc. USENIX Symposium on Internet Technologies and Systems (USITS)*, March 2003.

[6] Y. C. Hu, H. Pucha, and S. M. Das. Exploiting the Synergy between Peer-to-Peer and Mobile Ad Hoc Networks. In *Proc. of HotOS-IX: Ninth Workshop on Hot Topics in Operating Systems*, May 2004.

[7] D. B. Johnson and D. A. Maltz. Dynamic Source Routing in Ad Hoc Wireless Networks. In *Mobile Computing*, volume 353. Kluwer Academic Publishers, 1996.

[8] B. Karp and H. T. Kung. GPSR: greedy perimeter stateless routing for wireless networks. In *Proc. of ACM International Conference on Mobile Computing and Networking (MobiCom)*, August 2000.

[9] O. Landsiedel, S. Götz, and K. Wehrle. Towards Scalable Mobility in Distributed Hash Tables. In *Proc. of the 6th IEEE International Conference on Peer-To-Peer Computing (P2P)*, 2006.

[10] O. Landsiedel, K. A. Lehmann, and K. Wehrle. T-DHT: Topology-Based Distributed Hash Tables. In *Proc. of 5th IEEE Conference on Peer-to-Peer Computing (P2P)*, August 2005.

[11] J. Li, J. Stribling, T. M. Gil, R. Morris, and M. F. Kaashoek. Comparing the performance of distributed hash tables under churn. In *Proc. of the 3rd International Workshop on Peer-to-Peer Systems (IPTPS04)*, February 2004.

[12] M. Li, W.-C. Lee, and A. Sivasubramaniam. Efficient peer to peer information sharing over mobile ad hoc networks. In *Proc. of Second WWW Workshop on Emerging Applications for Wireless and Mobile Access (MobEA04)*, May 2004.

[13] C. Lindemann and O. Waldhorst. Exploiting Epidemic Data Dissemination for Consistent Lookup Operations in Mobile Applications. *ACM Mobile Computing and Communication Review (MC2R)*, 2004.

[14] R. Mahajan, M. Castro, and A. I. T. Rowstron. Controlling the Cost of Reliability in Peer-to-Peer Overlays. In *In Proc. of the Second International Workshop on Peer-to-Peer Systems (IPTPS)*, 2003.

[15] C. E. Perkins and E. M. Royer. Ad hoc On-Demand Distance Vector Routing. In *Proc. of the 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA)*, February 1999.

[16] H. Pucha, S. M. Das, and Y. C. Hu. Ekta: An Efficient DHT Substrate for Distributed Applications in Mobile Ad Hoc Networks. In *Proc. of 6th IEEE Workshop on Mobile Computing Systems and Applications (WMCSA)*, December 2004.

[17] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In *Proc. of ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM)*, September 2001.

[18] S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin, and F. Yu. GHT: A Geographic Hash Table for Data-Centric Storage in SensorNets. In *Proc. of ACM Workshop on Wireless Sensor Networks and Applications (WSNA)*, September 2002.

[19] G. Resta and P. Santi. An analysis of the node spatial distribution of the random waypoint model for Ad Hoc networks. In *Proc. of ACM Workshop on Principles of Mobile Computing (POMC)*, October 2002.

[20] S. Rhea, D. Geels, T. Roscoe, and J. Kubiatowicz. Handling churn in a dht. In *Proc. of USENIX Technical Conference (USENIX)*, June 2004.

[21] S. Shenker, S. Ratnasamy, B. Karp, R. Govindan, and D. Estrin. Data-Centric Storage in Sensornets. In *Proc. ACM Workshop on Hot Topics in Networks (HotNets)*, October 2002.

[22] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proc. of ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM)*, August 2001.

[23] A. Varga. The OMNeT++ Discrete Event Simulation System. In *Proc. of the European Simulation Multiconference (ESM)*, June 2001.

[24] T. Zahn and J. Schiller. MADPastry: A DHT Substrate for Practicably Sized MANETs. In *5th Workshop on Applications and Services in Wireless Networks (ASWN)*, 2005.