

T-DHT: Topology Based Distributed Hash-Tables

Abstract—In this paper we introduce T-DHT (Topology based Distributed Hash-Tables) as an infrastructure for data-centric storage, information processing, and routing in ad-hoc and sensor networks. It does not rely on location information and works even in the presence of voids in the network. Using a virtual coordinate system we construct a distributed hash-table, which is strongly oriented to the underlying network topology. Thus, adjunct areas in the hash-table commonly have a direct link in the network. Routing in the T-DHT introduces minimal hop-overhead compared to the shortest path routing.

I. INTRODUCTION

Research advances in highly integrated and low power hardware, wireless communication technology, and highly embedded operating systems enable new types of ad-hoc networks, such as sensor networks. A sensor network may consist of several hundreds of nodes, each with very limited processing, storage, and communication abilities. A sensor network provides large amounts of detailed measurements, which need to be aggregated, evaluated and stored in the network. However, the data is spread across the entire network and so difficult to access. Furthermore, as energy is a scarce resource and communication between nodes requires much energy, effective data propagation is a must.

In a sensor network, the derived data is commonly considered more important than the node, that gathered or stores it. Thus, *data-centric* communication and storage paradigms have been proposed. In such a scheme, data is named, e.g. "average temperature", and communication and storage refers to this name instead of a node's network address. A hash maps the name to a (key,value) pair. The key-space is distributed among the participating nodes. When retrieving and storing data, queries can be directly sent to the node responsible for the corresponding key.

The main contribution of this paper is a scalable and distributed algorithm for the construction of distributed hash-tables, which are strongly oriented to the underlying network topology. As a result, adjunct areas in the hash-table commonly have a direct link in the network. Thus, routing in the hash-table introduces minimal hop-overhead to shortest-path routing. Building a T-DHT consists of two steps: (1) The construction of a virtual coordinate space, which represents the network topology. (2) On top of these coordinates we build a two-dimensional hash-table, where each node maintains the data close to its virtual position.

The remainder of this paper is structured as follows. First, we introduce the lightweight virtual coordinate system in Section II. Section III presents the construction of the topology oriented hash-table. Next, Section IV evaluates the proposed technique. We discuss related work in Section V. Finally, Section VI concludes the paper and presents future work.

II. LIGHTWEIGHT VIRTUAL COORDINATE SYSTEM

In this section we introduce the algorithm to construct a virtual coordinate system. For this, three (or more) reference nodes are randomly selected. Each node triangulates its virtual position from these reference nodes. Thus, the reference nodes flood the network with the distance in hops between each other. Using the hop distance to the reference nodes, each node can determine its position in the virtual coordinate space.

Note that this position only presents the node's position in the network topology and not the physical position. However, we are not interested in the physical node positions. The consistent virtual coordinate system provided by the described algorithm is sufficient to construct a topology oriented hash-table (T-DHT) [1].

For our simulations we focus on two-dimensional coordinate systems and hash-tables. However, the schemes proposed in this paper can be easily adapted to three-dimensional systems.

III. TOPOLOGY ORIENTED HASH-TABLE

Inspired by the Content Addressable Network [1], we next construct a two-dimensional distributed hash-table on top of the virtual coordinate system, which we discuss in this section. The coordinate space is divided among the participating nodes, each node maintains a rectangular area around its position in the virtual coordinate system.

In the hash-table (key,value) pairs can be stored and retrieved deterministically. To store or retrieve data, a hash function maps the key to a position in the virtual coordinate space. Next, the data is routed through the hash-table to the node maintaining the area the key is in.

Each node maintains a routing table containing the path to its neighbors in the coordinate system and the area each maintains. Routing in the 2-dimensional hash-space is intuitively, using its routing table a node forwards a message to its destination by simple greedy forwarding. Additionally, a node may have links to nodes, which are not direct neighbors in the hash-table. Thus, it can use these to "short-cut" the route (see Fig. 1).

Joining the T-DHT consists of three steps: (1) To join, a node must first find a node, which is already in the T-DHT. (2) Via the T-DHT routing it finds the node maintaining the zone of its position in the virtual coordinate system. This zone is equally split between the two nodes. (3) The new member informs its neighbors about its presence.

To bootstrap, the first reference node maintains the whole hash-table at the beginning. When the virtual coordinates are assigned, it announces to its neighbors, that it is a T-DHT member. Next, the neighbors join the distributed hash-table and themselves announce their membership to their neighbors.

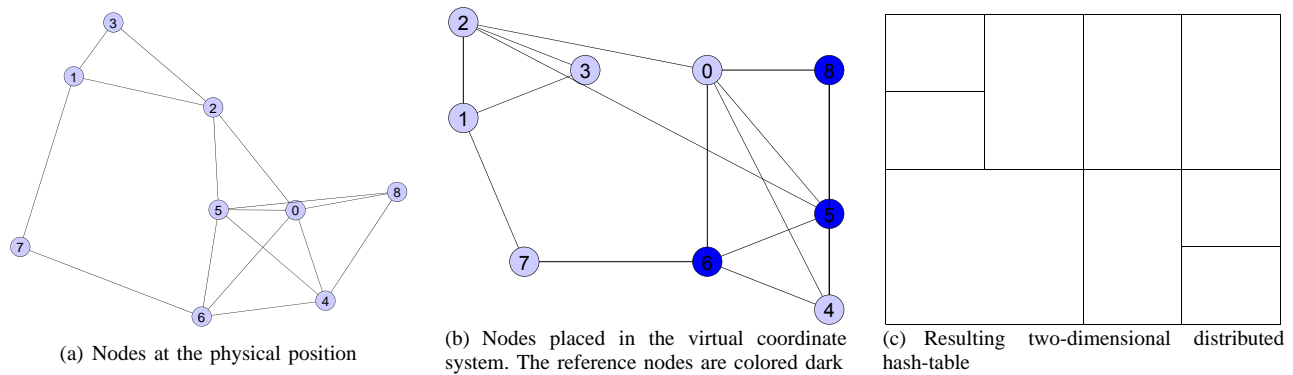


Fig. 1: Building a Topology Based Distributed Hash-Table (T-DHT) for nine nodes.

IV. EVALUATION

In this section we evaluate the performance of the introduced scheme. As pointed out, the proposed T-DHT is strongly oriented to the underlying network topology. Thus, adjacent areas in the hash-table commonly have a direct link in the network, enabling efficient routing.

Figure 2 shows the cumulative distribution function for the number of extra hops T-DHT routing takes over shortest path routing. The simulated network contained 30 nodes distributed over a 50m x 50m area, each with a communication range of 15m. Note, that T-DHT routes nearly 70% of the packets with less than 2 extra hops through the network.

V. RELATED WORK

Recent research provides many distributed hash-table system for usage in the internet: CAN [1], Chord [2], and Pastry [3]. However, they do not take the underlying network topology into account. New approaches focus on building topology aware distributed hash-tables for the internet [4], [5]. However, in the internet nodes are connected on a logic level and not on a physical level as in ad-hoc and sensor networks. Thus, the proposed algorithms do not apply to such a network. Data Centric Storage in sensor networks was first introduced by [6], [7]. However, the proposed technique relies on the fact, that each node knows its physical position. Like our approach GEM [8] proposes an algorithm for data centric storage and

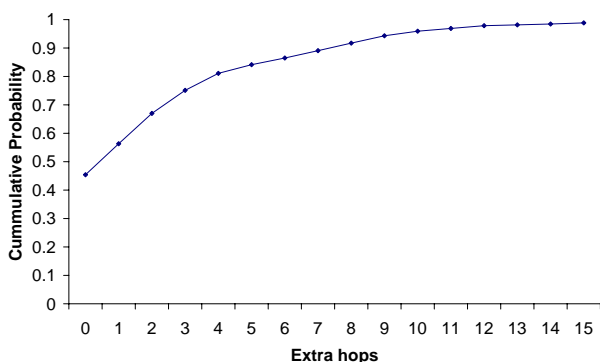


Fig. 2: CDF of extra hops over shortest path routing

routing. They use graph embedding and tree routing, thus it is a more complex algorithm to build the routing system. Their approach maintains a two hop neighborhood, while achieving comparable performance.

VI. CONCLUSION AND FUTURE WORK

In this paper we proposed a topology based distributed hash-table for data-centric routing and storage in ad-hoc and sensor networks. Using a simple virtual coordinate system we construct a distributed hash-table, which strongly oriented to the underlying network topology. Thus, the proposed scheme results in minimal overhead over shortest path routing.

The presented scheme is ongoing work, and many interesting questions are still open. Currently we are evaluating the impact of the positions of the references nodes as well as the number of reference nodes.

REFERENCES

- [1] S. Ratnasamy et al., "A scalable content-addressable network," in *Proc. of ACM SIGCOMM*, September 2001.
- [2] I. Stoica and R. Rinaldi, "Chord: A scalable peer-to-peer lookup service for internet applications," in *Proc. of SIGCOMM*, Aug. 2001.
- [3] A. Rowstron and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems," in *Proc. of IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, Nov. 2001.
- [4] M. Waldvogel and R. Rinaldi, "Efficient Topology-Aware Overlay Network," in *Proc. of HotNets*, Oct. 2002.
- [5] Z. Xu, C. Tang, and Z. Zhang, "Building Topology-Aware Overlays using Global Soft-State," in *Proc. of ICDSC*, May 2003.
- [6] S. Shenker et al., "Data-Centric Storage in Sensornets," in *Proc. HotNets*, Oct. 2002.
- [7] S. Ratnasamy et al., "GHT: A Geographic Hash Table for Data-Centric Storage in SensorNets," in *Proc. of WSNA*, Sep. 2002.
- [8] J. Newsome and D. Song, "Gem: Graph embedding for routing and data-centric storage in sensor networks without geographic information," in *Proc. of SenSys*, Nov. 2003.