Minimizing DHT Routing Stretch in MANETs

Extended Abstract

Marcel C. Castro, Andreas Kassler
Computer Science Department
Karlstad University, Sweden

Gabriel Kliot, Roy Friedman
Computer Science Department
Technion, Israel

Raphaël Kummer, Peter Kropf, Pascal Felber
Computer Science Department
University of Neuchâtel, Switzerland

Since the appearance of Napster in early 1999, peer-to-peer (P2P) networks have experienced tremendous growth. The P2P architectures can be categorized into two main classes: unstructured and structured P2P overlays. Unstructured overlays do not impose a rigid relation between the overlay topology and the indices/resources placement, as flooding or random walks are used to locate resources [3]. In contrast, structured P2P networks tightly control the overlay topology by arranging the nodes in a logical structure and by placing content at specified locations that will make subsequent lookups more efficient. Popular representatives of structured P2P networks are realized through the so called Distributed Hash Tables (DHTs), such as Chord [2].

Mobile Ad-hoc Networks (MANETs) usually do not have a dedicated routing infrastructure and rely on multi-hop communication. Nodes in a MANET cooperatively forward other nodes' data. These networks have a distributed communication architecture, where nodes make individual decisions on routing and medium access. P2P overlay networks in the Internet and MANETs share many key characteristics, such as self-organization and decentralization.

However, current P2P overlay architectures can not be directly used *as is* in MANETs, as it abstract the underlying physical topology during the overlay construction and resource lookup. For example, in Chord, each node has to maintain a set of logical neighbors (successor, predecessor and long-range neighbors) in the virtual identifier space, which requires significant control traffic. Long-range neighbors, also called fingers, are used to quickly route messages to remote locations in the identifier space. Given the limited MANETs bandwidth, the maintenance of logical neighbors can be prohibitively heavy-weight, as the logical neighbors could be located several hops away in the physical wireless topology. While this might be tolerable on the wired Internet with its high bandwidth, it is obviously not feasible for MANETs. Here, the delivery probability of a packet quickly decreases with each physical hop due to factors such as low bandwidth, limited energy, low computation power (of a node), packet collisions, or transmission errors.

Therefore, our envisioned DHT implementation combines a minimalist Chord-like overlay structure which replaces the long-range logical neighbors by physical neighbors. In this ("default") approach, each node only needs to maintain its successor and predecessor in order to guarantee correctness of the routing process, which imposes a limited management overhead. Thus, the lookup requests rely mainly on the physical neighbors. When receiving a lookup request, the node checks among its physical neighbors for the closest node to the requested key. If there is a node closer to the key in its physical neighborhood, such a node will become the next destination. We call this a shortcut link. If not, the request is forwarded to the next node in the logical path. As a result, at every step the lookup request gets closer to the node responsible for the key, until this node is reached.

Having more and diverse possibilities to select shortcut links can greatly improve the lookup efficiency of the overlay. Thus we propose one extension of the above default approach where the visibility of a node can be increased by two hop neighborhood information (which we denote as the neighbor-of-neighbor, "*NoN*") transmitted via hello (beacon) messages, ameliorating then the lookup efficiency while routing it across the MANET. Therefore, as it occurs in the default approach, when routing the request to a destination via the overlay, the node resorts to simple greedy routing by selecting the physical neighbor that makes the most progress in the logical ID space, and then forwarding the packet along the hop-by-hop route.

Another possibility to increase the lookup efficiency is to exploit the lookup request history through the use of a cache scheme. As nodes may participate in the forwarding process of some lookup requests, it should be possible to use this information to augment the ability to find even closer next logical hops for a specific key lookup request than its physical neighbors or neighbor-of-neighbor information. Therefore, each node maintains a "standard" cache that stores, for each observed lookup request, the searched key and the next logical hop neighbor of the lookup request. Thereafter a cache entry can act as a long-range logical neighbor, if it makes more progress to the destination compared to the physical neighbors and neighbors-of-neighbors information. As it is motivated later on through the results presented in Figure 1, by just caching the lookup requests that pass through a node may not be sufficient to guarantee lookup efficiency, we propose a "cross-layer" cache scheme that takes advantage of a 1-hop broadcast communication medium. Then, by also overhearing key lookup requests at the MAC layer, a node can

[1]This work was partially supported by MiNEMA - a European Science Foundation program for Middleware for Network Eccentric and Mobile Applications.

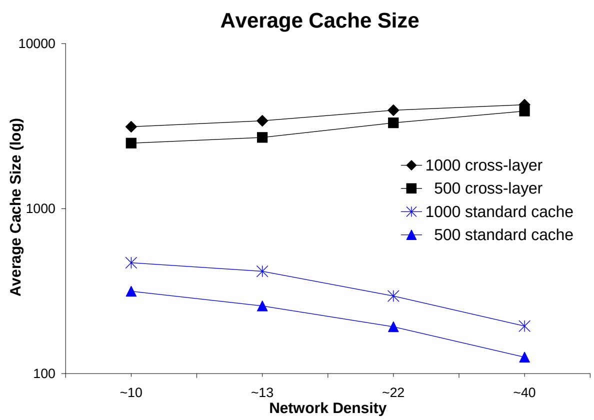

Fig. 1. Average cache size versus network density

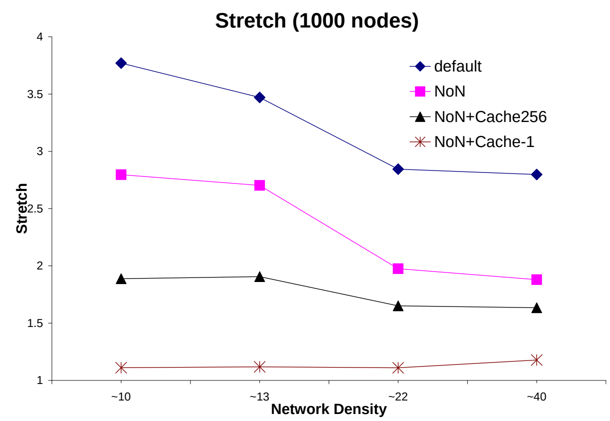

Fig. 2. Routing stretch versus network density

enrich its cache entries. This solution is particularly adapted to MANETs as no additional transmissions are required. This however comes at an increased energy cost, since nodes that constantly overhear all transmissions can not remain in the idle power-low state.

In our preliminary results, we consider a mobile ad-hoc network of 500 and 1000 nodes, uniformly distributed at random in a rectangular space. Different node densities are used in order to control the number of physical neighbors a node has in the system. Two nodes are considered neighbors if their distance is smaller than the transmission range limit of 100 meters. We remark that our current evaluation does not target mobility scenarios, thus nodes are stationary along the simulation. The simulation executes a warm-up phase to reach a steady state prior to issuing the key lookups. In the warm-up phase, each node generates 20 lookups to randomly selected keys. We use standard AODV as the routing protocol, which also routes key lookup requests between logical neighbors inside the MANET. For the cross-layer cache scheme, we consider two cases: nodes with limited cache size of 256 entries (*Cache256*) and nodes with infinite cache size (*Cache-1*). The limited cache size is implemented based on the Least Recent Used (LRU) scheme. To populate the cache, we used two strategies. First, only the requests passing through a node were memorized (standard cache). Then this caching policy has been extended by listening and memorizing request passing nearby the node (cross-layer). The performance metric under investigation is the DHT routing stretch of key lookups. The *routing stretch* is defined as the ratio between the length of the actual route of a key lookup used by our algorithm compared to the optimal shortest path routing [1].

Figure 1 motivates the use of ”cross-layer” information to enrich the caching scheme. In this figure we plot the average cache size per node (log scale) versus network density, for 500 and 1000 nodes topology. It is important to observe that we used infinite cache size per node in order to evaluate the maximum cache population that a node can acquire in such scenarios. It can be noted that by just caching lookups which pass through the node (standard cache) does not guarantee a high number of cache entries. Moreover, the number of entries for the standard cache decreases for higher network densities (e.g. 22 and 40 node density) as the number of logical nodes traversed by a key lookup reduces in denser networks. In contrast to that, by taking the advantage of 1-hop broadcast communication, the cross-layer caching enables the MANET’s nodes with the possibility to overhear key lookup packets and therefore acquire a higher (average) cache size per node.

In Figure 2, we compare an average DHT routing stretch of our DHT implementation as a function of different network densities. We compare four different approaches (*default*, *NoN*, *Cache256*, and *Cache-1*). Each result was obtained by averaging over 100 random key lookups performed after the warm-up period. As expected, the use of neighbor-of-neighbor (*NoN*) information compared with one-hop information (*default*) ameliorates the DHT routing stretch, specially in higher density topologies. This is because via neighbor-of-neighbor information each node can acquire a greater view of the network topology and therefore select better next logical neighbors. By using caching size of 256 entries allied to neighbor-of-neighbor information (*NoN*+*Cache256*), the DHT routing stretch can be reduced even further. In this case, the knowledge acquired by the cache during the warm-up phase and the overheard key lookups help in the selection of physical nodes closer to the destination, while routing through the logical neighbors. We also verified that using an infinite cache size leads to a DHT routing stretch of approximately one, which means that the selected route via the overlay is as efficient as the shortest path route. This is however not surprising, since using an infinite cache and infinite lifetime for entries means that after enough lookup requests a node will store all possible node identifiers, thus having the full network membership.

One problem with limited cache is that the cache may contain many entries which are not necessarily significant, as the cache entries may not be evenly mapped along the

[2]Routing messages could also be used to enrich the cross-layer caching scheme, however we do not consider it since we decide to design it independently of the routing protocol used.

logical space. Based on our results, one important point that should be analyzed further is how to better select and manage the cache entries to improve the lookup efficiency and consequently reduce its average path cost, while maintaining a viable number of cache entries. Possible caching strategies include (but are not limited to) caching according to harmonic distribution (inversely proportional to the logical distance on the ring), trade off between the logical quality of the shortcut and its physical cost, attaching confidence probabilities to the cache entries based on the last validation of the link. Therefore, by carefully selecting the cache entries that must be kept by a node we can 1) have a good coverage of the logical space, 2) cheap shortcut relying on physical nodes and 3) limited memory consumption while keeping only the best entries.

REFERENCES

[1] C. Cramer and T. Fuhrmann. Proximity Neighbor Selection for a DHT in Wireless Multi-Hop Networks. In *P2P '05: Proceedings of the Fifth IEEE International Conference on Peer-to-Peer Computing*, pages 3–10, Washington, DC, USA, 2005. IEEE Computer Society.

[2] I. Stoica and R. Morris and D. Karger and M. Frans Kaashoek and H. Balakrishnan. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In *Proc. of ACM SIGCOMM*, San Diego, USA, August 2001.

[3] C. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker. Search and Replication in Unstructured Peer-to-Peer Networks. In *Proc. of the 16th International Conference on Supercomputing (ICS)*, pages 84–95, 2002.