Proximity Neighbor Selection for a DHT in Wireless Multi-Hop Networks

Curt Cramer* Thomas Fuhrmann
System Architecture Group, University of Karlsruhe (TH), Germany

Abstract

A mobile ad hoc network *(MANET) is a multi-hop wireless network having no infrastructure. Thus, the mobile nodes have to perform basic control tasks, such as routing, and higher-level tasks, such as service discovery, in a cooperative and distributed way.*

Originally conceived as a peer-to-peer application for the Internet, distributed hash tables *(DHTs) are data structures offering both, scalable routing and a convenient abstraction for the design of applications in large, dynamic networks. Hence, DHTs and MANETs seem to be a good match, and both have to cope with dynamic, self-organizing networks.*

DHTs form a virtual control structure oblivious to the underlying network. Several techniques to improve the performance of DHTs in wired networks have been established in the literature. A particularly efficient one is proximity neighbor selection *(PNS). PNS has to continuously adapt the virtual network to the physical network, incurring control traffic. The applicability of PNS and DHTs for MANETs commonly is regarded as hard because of this control traffic, the complexity of the adaptation algorithms, and the dynamics of a MANET.*

Using simulations supported by analytical methods, we show that by making a minor addition to PNS, it is also applicable for MANETs. We additionally show that the specifics of a MANET make PNS an easy exercise there. Thus, DHTs deliver good performance in MANETs.

1. Introduction

In a *mobile ad-hoc network* (MANET), mobile nodes communicate over wireless links. Typically, a MANET does not have any infrastructure available. Since communication involves multiple hops, every node has to act as a router. Apart from routing, the nodes also have to cooperatively run higher-level protocols, such as service discovery or naming services.

As others have already noted, MANETs share their basic properties with *structured peer-to-peer (P2P) overlays*: Both have to cope with a dynamic environment and perform their functionality in a hop-by-hop manner [18]. Structured P2P overlays such as Chord [23], Pastry [22], or CAN [21] establish an application-defined address space in a virtual network overlaid onto an existing physical network. To date, however, most of the proposals for P2P overlay protocols are only suited for fixed networks like the Internet.

A well-known application based on the P2P principle is the *distributed hash table* (DHT). DHTs offer data storage in a scalable, distributed, fault-tolerant, and self-organizing manner. They inherit these properties from being implemented as structured P2P overlays.

As in a fixed-network environment, a P2P overlay can serve two purposes in MANETs: First, it can be used to provide scalable routing by implementing it at the network layer. And second, it can be used to provide the basic functionality for the implementation of distributed directory services. Thus, it seems natural to use P2P overlay solutions in order to solve the problems faced in MANETs.

Every routing control structure, that is a routing hierarchy or a structured P2P overlay as discussed here, abstracts from the underlying network graph. Compared to shortest-path routing, this abstraction introduces a performance penalty by lengthening the paths among nodes. Others have shown that this penalty can be made constant (i.e., independent of the number of nodes in a network) for an overlay such as Chord deployed on an Internet-like graph [8, 16, 28]. The technique used for achieving this is *proximity neighbor selection* (PNS). However, PNS only works on graphs having certain properties [28].

To our best knowledge, the performance of PNS in a MANET environment has not been evaluated before. Yet, PNS is performance-critical and a vital component of any scalable protocol implementing structured P2P overlays or DHTs. This especially applies to resource-limited scenarios like wireless MANETs.

Thus, to answer the question whether a DHT can deliver good performance in MANETs, we first have to answer the

*E-mail: cramer@ira.uka.de — Funding was provided by Deutsche Forschungsgemeinschaft under grant FU448-1. The author thanks Tobias Wissmüller for his help preparing the simulations, and the anonymous reviewers for their valuable comments.

question whether PNS is feasible there. (In the remainder of this paper, we will use the terms *DHT* and *structured P2P overlay* interchangeably; they are not the same, since a DHT is an application built on structured P2P overlays. However, both share the same basic structure to which PNS is applied.) This question can be broken down into several individual questions which we answer in this paper:

1. Does PNS work in MANETs at all?
2. If yes, how good is the performance of a DHT with PNS in MANETs?
3. How do we achieve good performance in practice?

In the next section of this paper, we provide the relevant background and define the models used here. In Section 3, we examine the general applicability of PNS to a DHT in the wireless case. We relax some idealized assumptions in Section 4 to examine practical considerations for the implementation of PNS in MANETs. Related work is discussed in Section 5. In the last section, we conclude our findings and give an outlook on directions for further research.

2. Background and Models

In this section, we describe the assumptions made, the background of the protocols, and the models employed for the evaluations presented in this paper. We use the PNS-enabled Chord [23] as a structured P2P overlay, since its structure is both simple and particularly well-suited for PNS [8, 16, 28]. The MANETs are modeled as *unit-disk graphs*.

2.1. Chord

Chord was proposed as a scalable lookup service for large-scale applications. It is a virtual network implemented as a structured P2P overlay. All n nodes in the virtual network have unique, location-independent overlay addresses that are drawn from a uniform random distribution on a large linear integer space $[0,1,2,\dots,s-1]$ ($s=2^m$) which is wrapped around at its borders, making it a ring. In the following, we use i to both denote the individual node, as well as its unique address.

Nodes are sorted on the this ring on the basis of their unique addresses, taking into account the wrap-around. Each node has to maintain a virtual link to its *successor*, that is the node directly following it in the thus ordered node set. With this structure, any node can route messages to any other node simply by each intermediate node forwarding the message to its successor until the destination is reached. This, however, results in path lengths of $O(n)$, rendering routing not scalable.

To make Chord routing scalable, each node i maintains a routing table of m entries where the entry with the index $l \in \{0,1,\dots,m-1\}$ is a virtual link to the *first* node lying in the address range $R_l(i) = [i+2^l, i+2^{l+1}) \pmod m$. (For convenience and legibility reasons, we do not repeat the $\pmod m$ in the following formulae; because of the address space wrap-around, however, any address range given has to be read as $\pmod m$.)

Stoica and others [23] provided a descriptive picture of routing in the thus constructed Chord network: To forward a message, the distance from the current intermediate node to the destination is expressed in binary notation. The l^{th} bit corresponds to the l^{th} entry in the current node's routing table. A bit having the value of 1 has to be "fixed" to a 0. This is greedily done by sending the message over the virtual link which–among all links where the corresponding bit in the distance is set to 1–makes the most progress in the address space, not exceeding the destination.

As a result, in a fully populated address space with each node keeping a state of m virtual links, a message is routed to its destination in $m/2$ virtual hops on average.

In less densely populated networks, each node has to keep $O(\log_2 n)$ state information, and the average path length is $(\log_2 n)/2$. If, as a result of the sparsity, there is no node $j \in R_{l_1}(i)$, the routing table entry with the index l_1 is duplicated from the entry for the next non-empty range $R_{l_2}(i)$ with $l_2 > l_1$.

Since we investigate P2P overlays for implementing basic control structures in MANETs, and since these control structures have to be cooperatively maintained by all nodes, we assume that every node in the MANET participates in the P2P overlay as well.

In the Internet, each virtual link corresponds to a transport connection (i.e., TCP or UDP). Chord has to regularly maintain its virtual links, thus issuing control traffic. In a MANET where bandwidth is scarce, a cross-layer approach where the structured overlay is implemented as a routing protocol at the network layer is more appropriate [6, 18, 20].

The decision of where the overlay is implemented affects the addressing architecture. If the overlay is implemented above the network layer, a node has both a virtual overlay address, and a network address. Then, virtual addresses have to be resolved into network addresses before routing can take place. Typically, a virtual address is generated by hashing the network address.

If the overlay is implemented within the network layer (called *overlay-based routing* by us), the distinction between the overlay address and the network address does not necessarily have to be made. There, a node's network address can also coincide with its virtual address.

We do not assume any particular protocol architecture in this paper, however. The results we derive are based on the graph-theoretical properties of a MANET. They are generally applicable to designs of both the above-mentioned architectural types.

2.2. Proximity Neighbor Selection (PNS)

As was mentioned above, the average length of a path in the Chord overlay is $O(\log_2 n)$ virtual hops. Virtual hops correspond to physical paths in the underlying network. Let Δ be the average length of a shortest path in the underlying network. (The unit of Δ can either be hops or time.) Because of the random address distribution, establishing virtual links in the original way prescribed by Chord results in randomly selected physical paths. Therefore, the average physical length of one virtual hop is Δ, too. In summary, the average concatenated physical path length of a complete virtual path is $O(\Delta \log_2 n)$.

The performance penalty of routing in the overlay over taking the shortest path in the underlying network is quantified by the *stretch*. It is defined as the ratio of the combined lengths of the physical paths traversed for a complete virtual path to the length of the shortest physical path between the source and the destination node. For example, in the unoptimized Chord, the average stretch is $O(\log_2 n)$.

However, the stretch of Chord can be reduced to a constant by relaxing the rules for constructing the overlay. (In the following, we call this relaxation *Randomized Chord (R-Chord)*.) Instead of pointing to the first node in $R_l(i)$, a virtual link can point to *any* node in $R_l(i)$, making it possible to select physically close nodes as virtual neighbors. This is called *proximity neighbor selection* [4, 16]. The modified construction rule does neither alter the required amount of state, nor the average virtual path length. Selecting appropriate neighbors, however, the combined physical length of a virtual path is reduced to $O(\Delta)$, and thus stretch is $O(1)$. Empirically observed by Gummadi and others [16], this result has been formally proven by Zhang and others [28]. Dabek and others [8] estimated the constant factor in the stretch of $O(1)$ to be 2 by assuming a uniform latency distribution and approximating the average latency by the median latency. This roughly coincides with results measured using a more realistic latency distribution [16].

All of these results assume *perfect information*. There, nodes know the overlay addresses of all other nodes, and their distance to all other nodes in the network. This allows them to select the best neighbors for all address ranges and makes it possible to determine the optimum stretch. The practical difficulty of PNS lies in actually *finding* physically nearby neighbors which fall into the respective address ranges.

2.3. Wireless Multi-Hop Networks

We model wireless multi-hop networks as *unit-disk graphs* which are geometric random graphs. The n nodes in the network are placed onto a two-dimensional, quadratic area $A = a^2$ according to a random uniform distribution. An edge between two nodes i and j is added if for their Euclidean distance $d(i, j)$ it holds that $d(i, j) \leq r = 1$. The homogeneous assignment of a single transmission radius r to all nodes does not correctly reflect physical propagation [24], however the model is commonly used since it allows easier analyses.

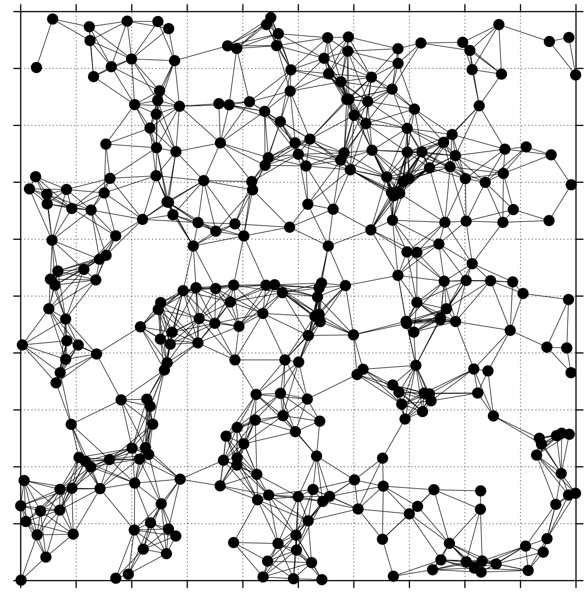

Figure 1. Instance of a unit-disk random graph

Fig. 1 depicts an example instance of a unit-disk graph with $n = 315$, $r = 1.0$, and $a = 10.0$. To avoid artifacts in the results presented here, it was ensured that all generated graphs are connected (i.e., there exists a path from any node to any other node).

A mobility model which–at any instance of time–results in a uniform node distribution as described above is the *random direction* model [2, 3, 15]. In this paper, we do not consider direct effects resulting from node mobility such as link breaks and protocol overhead; we are only interested in statistical properties, that is the lengths of paths between nodes, and the number of nodes falling into certain portions of the overlay address space. We thus assume the MANET to be in a steady state, meaning that its *statistical* properties do not change over time. This enables us to derive results from the respective probability distributions. For comparison, we also obtained values from simulations. If not stated otherwise, each simulation result is the average of ten independent simulation runs. In each run, a new graph is instantiated.

Considering the number of nodes in a given sub-area of the quadratic deployment area, it is difficult to calculate the exact distribution because of the border limiting the deployment area. There, nodes have a different number of one-hop neighbors than in the center of the area. For complexity reasons, we neglect these border effects which nevertheless gives us a very good approximation of the simulated values, as can be seen in the following.

For this case, Bettstetter obtained the result that in the limit, the uniform distribution becomes a homogeneous Poisson point process of density $\lambda = n/A$ if $n \to \infty$, $A \to \infty$, while λ is constant [3]. This stochastic process has the property that the number of nodes in a finite subarea $B \leq A$ is Poisson distributed with mean λB.

3. PNS in Wireless Multi-Hop Networks

3.1. Efficacy

Zhang and others [28] made the fundamental finding that the efficacy of PNS is dependent on the *expansion* property of the underlying network graph. They define $N(x)$ to be the number of nodes within latency x of a node. A graph has *d-power-law latency expansion* if $N(x) \propto x^d$, and *exponential latency expansion* if $N(x) \propto \alpha^x$ with $\alpha > 1$. Their result is that PNS is only effective if a graph has d-power-law expansion. Then, latency stretch of a PNS-enabled R-Chord is a constant. With exponential expansion, latency stretch grows as $O(\log n)$. Note that all studies on PNS in fixed networks employ *latency* as their proximity measure [8, 16, 28].

In a MANET, packet loss and mobility make it difficult to measure the actual round-trip time between two nodes, as was shown in an evaluation of TCP performance by Holland and others [17]. Therefore, we use the hop distance between two nodes as the measure of their proximity. We justify this choice by the fact that most MANET routing protocols also use this measure for calculating paths. To quantify the quality of PNS, we thus have to consider hop stretch instead of latency stretch. (Both measures are the same if we assume unit or constant latency for traversing a single hop.)

Hence, to show the efficacy of PNS in a MANET, we have to determine the hop expansion property of unit-disk graphs. Let $N(k)$ be the number of nodes within k hops of a node. The area covered by k hops equals $\pi(kr)^2$, and the average number of nodes in this area is $N(k) = \lambda \cdot \pi(kr)^2 \propto k^2$ (cf. Section 2.3). This means that a unit-disk graph has 2-power-law hop expansion, affirming the efficacy of PNS in a MANET complying with our model.

3.2. Influencing Variables

Next, we examine the quantitative performance of PNS, that is the hop stretch, in MANETs. Potentially, all model parameters could influence stretch. These are: 1) the length of an address (in bits) m, 2) the number of nodes n, and 3) the network density $\lambda = n/a^2$. The density is varied by keeping n constant while varying a.

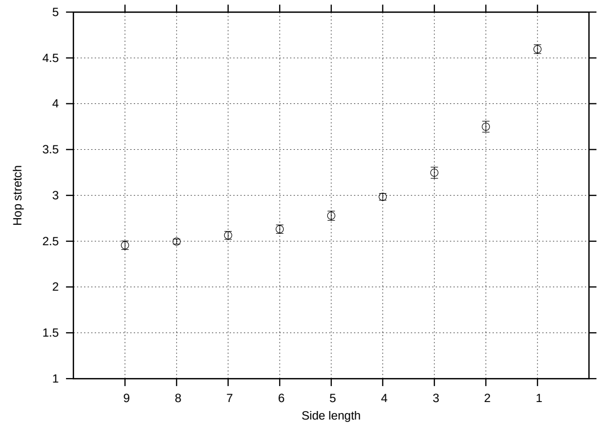

Figure 2. Effect of network density on hop stretch (500 nodes)

3.3. Performance Evaluation

As stated before, we ensured the networks to be connected. Starting with the lowest possible density, the side length a was decreased in integer steps down to 1. For a 500-node network, this means that $a \in \{9, 8, \dots, 1\}$.

Each data point is the average of ten independent simulation runs. In each run, 1 000 messages were sent from random sources to random destinations, both drawn from a uniform distribution. The figures show approximate 95%-confidence intervals calculated using Student's t distribution with 9 degrees of freedom.

The address length m was varied from 9 to 31 bits for a 500-node network with $a = 9$. No significant effect on stretch, which on average was 2.37, could be found. The same observation applies to varying n from 100 to 2 000, where network density was chosen to be the lowest possible for each n, respectively. We believe that the deviation from the theoretical value of 2 published elsewhere (cf. Section 2.2, and [8]) results from the path lengths not being uniformly distributed, and from calculating average values instead of median values.

As can be seen from Figure 2, network density has a significant effect on hop stretch. Stretch rises steeply as the network gets more dense. The reason for this is that with $a \to 1$ and thus $\lambda \to n$, the average path length Δ converges to the minimum value of 1. This can also be seen in Figure 3 which depicts the empirical cumulative distribution functions for different network densities.

Hence, in dense networks, it does not make a difference whether PNS is used or not, since all shortest physical path lengths are close to 1, coinciding with Δ. Therefore, PNS becomes ineffective, as stretch then equals the average virtual path length of $(\log_2 n)/2$. This can also be seen in Figure 4 where the stretch value is shown for $n \in [100, 2\,000]$ and $\lambda = n$. The measured values perfectly agree with the expected theoretical values.

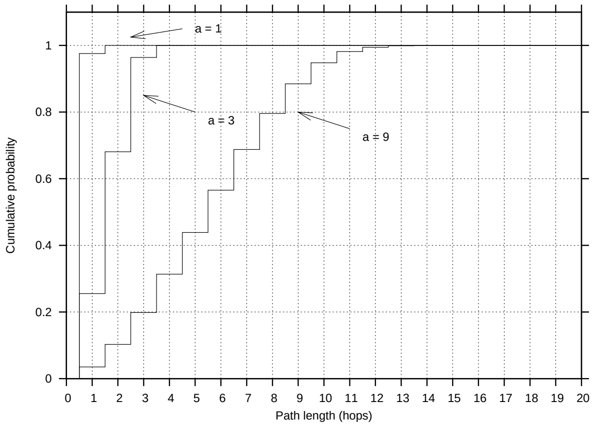

Figure 3. Empirical path length CDFs (500 nodes, $a \in \{9, 3, 1\}$)

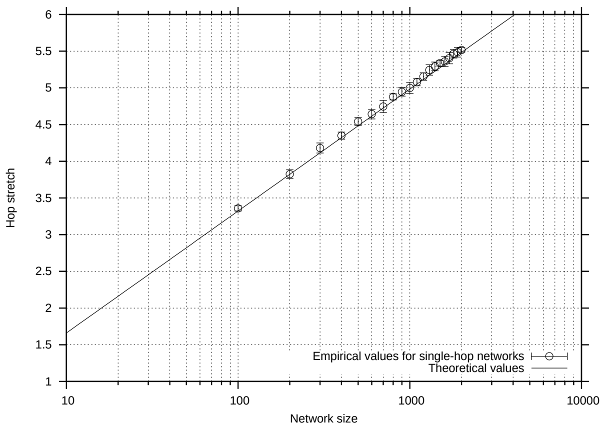

Figure 4. Hop stretch with PNS in single-hop networks

To improve the performance of PNS in dense networks, nodes should maintain additional information about the other nodes in their physical neighborhood. By caching the overlay addresses of the nodes in the *k*-hop neighborhood, the overlay is augmented with more links. The greedy routing procedure of Chord remains unchanged with each node forwarding the message to its virtual neighbor which reduces the distance towards the destination the most.

The overlay protocol instance on a node does not necessarily know the addresses of the node's physical neighbors. However, information about the 1-hop neighborhood can be easily provided to the overlay protocol by an underlying routing protocol, or by each node periodically emitting *hello* packets containing its address via radio broadcast. (As discussed in Section 2.1, the type of the address depends on the protocol architecture. It can either be an overlay address or a network address which is to be hashed into the corresponding overlay address.) If each node additionally appends the addresses of its currently known 1-hop neighbors to this packet, all nodes get to know their 2-hop neighborhood.

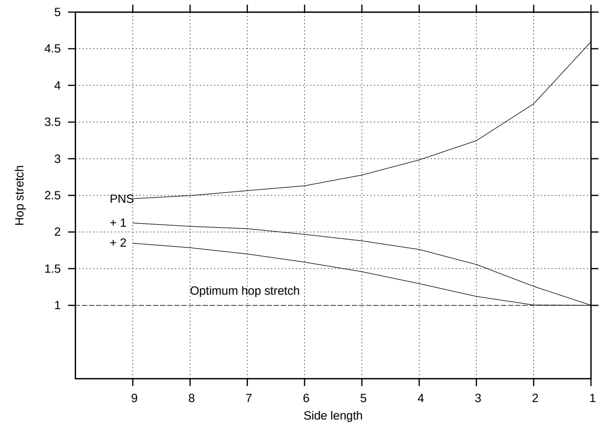

Figure 5. PNS improvement by using neighborhood information (500 nodes)

We evaluated the effect of augmenting the basic Chord network with the virtual links obtained from the nodes' 1- and 2-hop neighborhoods, respectively. Figure 5 compares the hop stretch of plain PNS with that of the PNS using 1-hop information (labeled "+ 1"), and 2-hop information (labeled "+ 2"). It can be seen that this simple and–in terms of per-node control packet overhead–cheap modification of PNS yields a significant performance improvement. This is not only the case for dense networks, where stretch now is close to its optimum value of 1, but even for sparse networks.

4. Practical Considerations for PNS

To achieve good performance for routing in a structured P2P network, PNS has to select physically close nodes for the virtual address ranges of a node. In a practical setting, the perfect information assumption made before (cf. Section 2.2) does not hold. Getting perfect information about the distances to all other nodes in the network requires continuously updated measurements of these distances. The cost of perfect information becomes prohibitively expensive for larger networks. Thus, we have to relax this assumption.

The question of how PNS can achieve good performance in practice therefore translates into the following question: *How easily can we find good virtual neighbors?* We examine this question in the remainder of this section.

4.1. Sampling

To establish the virtual overlay network, each node i has to find other nodes falling into its address ranges $R_l(i)$ where $l \in \{0, 1, \ldots, m-1\}$. (In the following, we call l the *range index* of the corresponding range.) The probability that a node falls within $R_l(i) = [i + 2^l, i + 2^{l+1})$ is $p_l = 2^l / 2^m = 2^{l-m}$. Thus, the expected number of nodes in $R_l(i)$ is $n \cdot 2^{l-m}$. This limits the number of actually required neighbors, as for $n \ll 2^m$, many ranges will not contain any node.

The largest required index is $m-1$. The smallest required index is determined by a node's successor, since by definition, there does not exist any other node whose address lies between the address of a node and that of its successor. As there is exactly one successor per node, for the expected index l_s of a successor, it holds that the expected number of nodes in R_{l_s} is $n \cdot 2^{l_s - m} = 1$. Thus, $l_s = m - \log_2 n$.

Node i does not only have to find a neighbor for each required l ($l_s \le l \le m-1$). For PNS to work properly, i has to find good, that is nearby neighbors. The process of finding a nearby neighbor for a range among all eligible neighbors is called *sampling*. Sampling is easy to model in our MANET scenario. To find a good neighbor, node i has to search for it in an increasingly expanding neighborhood around it. (Note that this does not imply any protocol behavior; as in Section 3.3, node i could have cached information about its k-hop neighborhood in which it searches.) The number of nodes that have to be inspected, that is their virtual addresses have to be known to i, until the first node falling into the respective address range is found follows a geometric distribution with mean $1/p_l = 2^{m-l}$.

We can now equate this mean with the average number of nodes within the k-hop distance of i to get the expected minimum number of hops towards a good node for range index l. This yields $k = \sqrt{2^{m-l}/(\lambda\pi)}$, making the distance of good neighbors for l only dependent on the network density λ, since m is fixed for all nodes. The maximum hop distance $k_{\max}$ to search in for all virtual neighbors is determined by l_s. It follows that $k_{\max} = \sqrt{n/(\lambda\pi)} = a/\sqrt{\pi}$. (Note that a cannot be chosen independently of n, since network connectivity requires a minimum network density [3]. If the density λ is kept fixed while varying n, then $k_{\max}$ is in $\Theta(\sqrt{n})$.)

4.2. Performance Evaluation

We evaluated the theoretical results of the last section by comparing them with simulation results. To account for the maximum search distance $k_{\max}$ which is determined by l_s, we calculate k as follows:

$$k(l) = \begin{cases} \sqrt{2^{m-l}/(\lambda\pi)}, & \text{if } l > \lfloor l_s \rfloor; \\ \sqrt{n/(\lambda\pi)} = a/\sqrt{\pi} = k_{\max}, & \text{otherwise.} \end{cases}$$

(Since messages can only be sent over complete hops, in our evaluation, we round k to the next greater integer.) In a network with $n = 500$, $a = 9$, and $m = 31$, for each range index, we determined 1) the average minimum distance, 2) the average distance, and 3) the average maximum distance to nodes falling into the corresponding address range. The result is depicted in Figure 6.

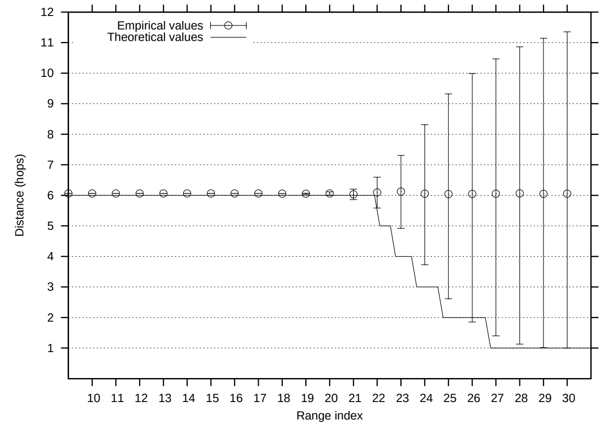

Figure 6. Min., average, and max. distances to virtual neighbors (500 nodes)

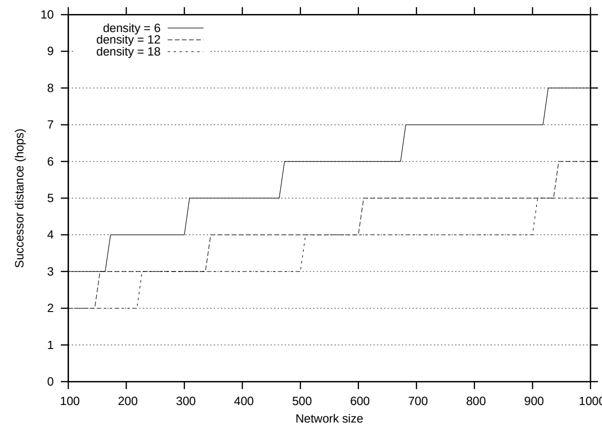

Figure 7. Expected successor distance $\lceil k_{max} \rceil$

The theoretical result for k gives a good approximation of the measured average minimum distances to the virtual neighbors. As mentioned above, the maximum hop distance to search in is determined by the length of the path from a node to its successor. For a fixed network density λ, the average length of this path is in $\Theta(\sqrt{n})$. This can be seen in Figure 7 which shows the expected distance $\lceil k_{\max} \rceil$ to a node's successor in networks of 100–1 000 nodes with densities $\lambda \in \{6, 12, 18\}$.

As was described in Section 2.1, a virtual address size m of 31 bits does not necessarily imply that a node has to maintain 31 different virtual neighbors. As a result of the sparsely populated virtual address space, in the example of Figure 6, all routing table entries below the index of the successor ($\approx$ 21–22) are filled with the address of the successor. That is, each node only has to maintain $\approx$ 9–10 different virtual neighbors.

It can be seen that it is particularly easy to find neighbors for the higher range indexes. Fifty per cent of all required nodes are situated within 2 hops of a node, and sixty per cent are within 3 hops, even though the network is sparse.

We obtained similar results for a sparse network of 1 000 nodes. This can be explained as follows: For fixed k, the number of range indexes for which virtual neighbors are found in the physical k-hop neighborhood of a node only depends on λ. Thus, if λ and k are fixed, the number of virtual neighbors found is constant. The range index l_s of the successor is a logarithmic function of the network size n. As only the neighbors with indexes $\geq l_s$ and $\leq m-1$ are required, the number of virtual neighbors per node is also a logarithmic function of n. Therefore, the percentage of neighbors found within k hops among all required neighbors only diminishes slowly with n.

4.3. Discussion

The results presented in the last section also show that a large number of physically close virtual neighbors can be easily found in the presence of mobility. Statistically, there is always a sufficient number of appropriate nodes available in the physical neighborhood of a node. If a node has current knowledge about its k-hop neighborhood, it will also find suitable virtual neighbors with high probability.

Since with probability $1/2$ each range is used for forwarding a message, nodes have to find good neighbors for all of their ranges in order to make PNS effective. Finding neighbors for l gets increasingly harder as l gets smaller. Thus, for this case, more efficient approaches than searching in the k-hop neighborhood of a node have to be used. Otherwise, if k gets too large, this corresponds to a flooding-based solution which limits network scalability.

Therefore, a hybrid solution for discovering all virtual neighbors seems to be appropriate. For larger range indexes, for which nodes can be found within few physical hops, a proactive approach for continuously maintaining information about the k-hop neighborhood of a node (cf. Sec. 3.3) can be used. This has the two-fold benefit that the average hop stretch of PNS is reduced (cf. Sec. 3.3), and that a large fraction of the required virtual neighbors is found (cf. Sec. 4.2).

For smaller range indexes, neighbors should be discovered more efficiently than with a flooding-based approach. This can be done by exploiting the structure of the P2P overlay. For example, ISPRP [6, 7] demonstrates how to efficiently find paths to the successor nodes in our scenario.

Neighbors for the remaining, intermediate range indexes can be obtained from querying the overlay. Although this selection does not necessarily imply that the best nodes are chosen for these ranges, it can be a good solution for two reasons: First, the number of nodes appropriate for these ranges is limited. And second, the differences in the distances to the eligible nodes do not vary that much as for the higher range indexes. However, this approach has yet to be evaluated. The cost for discovering and maintaining these remaining virtual neighbors might outweigh their actual performance benefit.

This hybrid approach as discussed above differs from previously published, non-hierarchical MANET routing protocols in that for a uniform traffic pattern, a solution based on structured P2P overlays only has to maintain $O(\log n)$ paths per node instead of $O(n)$ paths.

5. Related Work

A number of publications has been devoted to bringing P2P protocols into ad-hoc networks. Many of the publications are mainly conceptual, presenting architecture proposals but not evaluating them (cf. [5, 9, 11, 13, 18, 26]).

Hu and others [18] proposed DPSR, a cross-layer approach for using the structured P2P overlay Pastry as a network-layer routing protocol by integrating it with the MANET routing protocol DSR [19]. No performance evaluation of DPSR is available.

Ekta [20], which is based on DPSR, is a DHT for MANETs. It is also based on the proximity-aware Pastry overlay and DSR. The details of the proximity adaptation in the MANET scenario have not been explicitly evaluated by the authors. Yet, they claim that Chord cannot be adapted to the physical network as flexible as Pastry; the results published earlier by Gummadi and others [16] disprove this claim.

ISPRP [6, 7] is a self-stabilizing protocol that creates the basic ring structure of a Chord overlay for overlay-based routing at the network layer of an ad-hoc network. It does not employ flooding for discovering the routes to the successor nodes. However, it does not handle node mobility at present.

CrossROAD [10] is also based on Pastry; the evaluation was made in a network of only eight nodes.

Eberspächer and others [13], and Gruber and others [14] argue that structured P2P overlays cannot be adapted to the underlying physical network, and thus are infeasible in MANETs, ignoring the previously published results on PNS for structured P2P overlays in the seminal works by Castro and others [4], and by Gummadi and others [16].

A different strategy than PNS for adapting a virtual network to its underlying network is *proximity identifier selection* (PIS) [16] where a node's overlay address is chosen according to the node's current location (cf. [12,25,27]). To enable location-independent addressing for routing, in this case, a level of indirection has to be introduced for mapping a location-independent address to its current location-dependent address. Maintaining and querying this directory service incurs additional communication overhead.

Unstructured P2P networks have less rigid construction rules than structured P2P overlays do. By this, they can be adapted to the physical network with more flexibility. Therefore, some authors advocate the use of unstructured

P2P overlays for MANETs (cf. [1, 14]). However, these overlays cannot provide bounded path lengths or lookup guarantees as structured P2P overlays can, and in general require flooding within the overlay for sending messages, making them less scalable.

6. Conclusions and Future Work

The performance of structured P2P overlays and their applications, such as DHTs, strongly depends on the quality of the adaptation of their virtual network to the underlying physical network. A well-known technique for this adaptation, where node addresses remain location-independent, is PNS. With PNS, the performance penalty paid for routing in the virtual network compared to routing in the underlying network is constant for certain classes of networks. However, in the literature, PNS has only been considered for Internet-like graphs.

In this paper, we affirmatively answered the question whether PNS is also applicable to MANETs. Using Chord as a basis, we presented simulation results to quantify the quality of PNS in MANETs. For sparsely connected networks, it is comparable to the case of fixed networks. However, the performance of PNS was shown to be strongly dependent on the network density, degrading with increasing density. We have shown that it is significantly improved by augmenting the virtual network with additional information about a node's physical neighborhood.

Using simulations and analytical models, we have shown that sampling in a MANET, that is the process of selecting proximate virtual neighbors for adapting the overlay to the physical network, is easy even for sparse networks of 1 000 nodes: a large fraction of the required virtual neighbors is found within 2–3 hops. This result is important regarding node mobility, since it also implies that for being able to forward messages, nodes need not continuously maintain their virtual links to these neighbors; with high probability, there is always a sufficient number of suitable virtual neighbors available in the physical neighborhood of a node.

It remains an open and important research question how large the savings in protocol overhead of a structured P2P overlay actually get by leveraging the results presented here. We are currently developing a network-layer routing protocol based on a structured P2P overlay to answer this question.

References

[1] L. Barbosa e Oliveira, I. G. Siqueira, and A. A. F. Loureiro. Evaluation of Ad-Hoc Routing Protocols under a Peer-to-Peer Application. In *Proc. IEEE WCNC 2003*, pp. 1143–1148, New Orleans, LA, USA, Mar. 2003.

[2] C. Bettstetter. Mobility Modeling in Wireless Networks: Categorization, Smooth Movement, and Border Effects. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(3):55–66, 2001.

[3] C. Bettstetter. On the Connectivity of Ad Hoc Networks. *The Computer Journal*, 47(4):432–447, July 2004.

[4] M. Castro, P. Druschel, Y. C. Hu, and A. Rowstron. Topology-aware routing in strucutured peer-to-peer overlay networks. Technical Report MSR-TR-2002-82, Microsoft Research, Redmond, WA, USA, 2002.

[5] M. Conti, E. Gregori, and G. Turi. Towards Scalable P2P Computing for Mobile Ad Hoc Networks. In *Proc. of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops (PERCOMW '04)*, pp. 109–113, Orlando, FL, USA, Mar. 2004.

[6] C. Cramer and T. Fuhrmann. ISPRP: A Message-Efficient Protocol for Initializing Structured P2P Networks. In *Proc. of the 24th IEEE International, Performance, Computing, and Communications Conference (IPCCC)*, pp. 365–370, Phoenix, AZ, USA, Apr. 2005.

[7] C. Cramer and T. Fuhrmann. Self-Stabilizing Ring Networks on Connected Graphs. Technical Report 2005-5, Fakultät für Informatik, Universität Karlsruhe (TH), Germany, 2005.

[8] F. Dabek, J. Li, E. Sit, J. Robertson, M. F. Kaashoek, and R. Morris. Designing a DHT for Low Latency and High Throughput. In *Proc. NSDI '04*, pp. 85–98, San Francisco, CA, USA, Mar. 2004.

[9] A. Datta. MobiGrid: Peer-to-Peer Overlay and Mobile Ad-Hoc Network Rendezvous - a Data Management Perspective. In *Proc. of the CAiSE 2003 Doctoral Symposium*, Klagenfurt, Austria, June 2003.

[10] F. Delmastro. From Pastry to CrossROAD: CROSS-Layer Ring Overlay for AD Hoc Networks. In *Proc. of the 3rd IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom 2005 Workshops)*, pp. 60–64, Kauai Island, HI, USA, Mar. 2005.

[11] G. Ding and B. Bhargava. Peer-to-peer File-sharing over Mobile Ad hoc Networks. In *Proc. of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops (PERCOMW '04)*, pp. 104–108, Orlando, FL, USA, Mar. 2004.

[12] S. Du, A. Khan, S. PalChaudhuri, A. Post, A. K. Saha, P. Druschel, D. B. Johnson, and R. Riedi. Self-Organizing Hierarchical Routing for Scalable Ad Hoc Networking. Technical Report TR04-433, Department of Computer Science, Rice University, Houston, TX, USA, 2004.

[13] J. Eberspächer, R. Schollmeier, S. Zöls, and G. Kunzmann. Structured P2P Networks in Mobile and Fixed Environments. In *Proc. of the International Working Conference on Performance Modeling and Evaluation of Heterogeneous Networks (HET-NETs '04)*, Ilkley, West Yorkshire, UK, July 2004.

[14] I. Gruber, R. Schollmeier, and W. Kellerer. Performance Evaluation of the Mobile Peer-to-Peer Service. In *Proc. of the IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2004)*, pp. 363–371, Chicago, IL, USA, Apr. 2004.

[15] R. A. Guérin. Channel Occupancy Time Distribution in a Cellular Radio System. *IEEE Transactions on Vehicular Technology*, 35(3):89–99, Aug. 1987.

[16] 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. ACM SIGCOMM 2003 Conf.*, pp. 381–394, Karlsruhe, Germany, 2003.

[17] G. Holland and N. Vaidya. Analysis of TCP Performance over Mobile Ad Hoc Networks. In *Proc. ACM/IEEE MobiCom '99*, pp. 219–230, Seattle, WA, USA, 1999. ACM Press.

[18] Y. C. Hu, S. M. Das, and H. Pucha. Exploiting the Synergy between Peer-to-Peer and Mobile Ad Hoc Networks. In *Proc. HotOS IX Workshop*, pp. 37–42, Lihue, HI, USA, 2003.

[19] D. B. Johnson and D. A. Maltz. Dynamic Source Routing in Ad Hoc Wireless Networks. *Mobile Computing*, 353:153–181, Feb. 1996.

[20] 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 the 6th IEEE Workshop on Mobile Computing Systems and Applications (WMCSA 2004)*, English Lake District, UK, Dec. 2004.

[21] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A Scalable Content-Addressable Network. In *Proc. ACM SIGCOMM 2001 Conf.*, pp. 161–172, San Diego, CA, USA, 2001. ACM Press.

[22] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *Proc. IFIP/ACM Middleware Conf. 2001*, Heidelberg, Germany, Nov. 2001.

[23] 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. ACM SIGCOMM Conf. 2001*, pp. 149–160, San Diego, CA, USA, 2001.

[24] I. Stojmenovic, A. Nayak, and J. Kuruvila. Design Guidelines for Routing Protocols in Ad Hoc and Sensor Networks with a Realistic Physical Layer. *IEEE Communications Magazine*, 43(3):101–106, Mar. 2005.

[25] A. C. Viana, M. Dias de Amorim, S. Fdida, and J. Ferreira de Rezende. An Underlay Strategy for Indirect Routing. *Wirless Networks*, 10(6):747–758, 2004.

[26] L. Yan, K. Sere, and X. Zhou. Towards an Integrated Architecture for Peer-to-Peer and Ad Hoc Overlay Network Applications. In *Proc. of the 10th IEEE International Workshop on Future Trends of Distributed Computing Systems (FTDCS '04)*, pp. 312–318, Suzhou, China, May 2004.

[27] T. Zahn, R. Winter, and J. Schiller. Simple, efficient peer-to-peer overlay clustering in mobile, ad hoc networks. In *Proc. IEEE ICON 2004*, pp. 520–524, Singapore, Nov. 2004.

[28] H. Zhang, A. Goel, and R. Govindan. Incrementally Improving Lookup Latency in Distributed Hash Table Systems. In *Proc. ACM SIGMETRICS '03 Conf.*, pp. 114–125, San Diego, CA, USA, 2003.