Pushing Chord into the Underlay: Scalable Routing for Hybrid MANETs

Thomas Fuhrmann, Pengfei Di, Kendy Kutzner, and Curt Cramer *
System Architecture Group, Universität Karlsruhe (TH), Germany
{fuhrmann|di|kutzner|cramer}@ira.uka.de

Fakultät für Informatik
Technical Report 2006-12

June 21, 2006

Abstract

SCALABLE SOURCE ROUTING is a novel routing approach for large unstructured networks, for example hybrid mobile ad hoc networks (MANETs), mesh networks, or sensor-actuator networks. It is especially suited for organically growing networks of many resource-limited mobile devices supported by a few fixed-wired nodes. SCALABLE SOURCE ROUTING is a full-fledged routing protocol that directly provides the semantics of a structured peer-to-peer overlay. Hence, it can serve as an efficient basis for fully decentralized applications on mobile devices.

SCALABLE SOURCE ROUTING combines source routing in the physical network with Chord-like routing in the virtual ring formed by the address space. Message forwarding greedily decreases the distance in the virtual ring while preferring physically short paths.

Unlike previous approaches, scalability is achieved without imposing artificial hierarchies or assigning location-dependent addresses. SCALABLE SOURCE ROUTING enables any-to-any communication in a flat address space without maintaining any-to-any routes. Each node proactively discovers its virtual vicinity using an iterative process. Additionally, it passively caches a limited amount of additional paths.

By means of extensive simulation, we show that SCALABLE SOURCE ROUTING is resource-efficient and scalable well beyond 10,000 nodes. A MIPS-Linux version demonstrating the real-world feasibility of SCALABLE SOURCE ROUTING is available.

*Supported by Deutsche Forschungsgemeinschaft under grant FU448-1.

1 Introduction

The complexity of planning and managing distributed systems quickly rises with their size and often requires experts. Nonetheless, an increasing number of consumer products are being sold with networking capabilities, be it wired or wireless. Networks installed by consumers should require as little planning and management as possible, or be completely self-organizing. The same requirements apply to disaster recovery scenarios where rapid deployment of a backup communication infrastructure is needed.

We consider a scenario where consumers incrementally create a network of electronic appliances such as portable computers, consumer electronics, embedded controllers, and so forth. The multi-hop network mixes wired and wireless links, as well as fixed and mobile devices. Most devices in such networks are highly available, while others may have volatile connectivity and be prone to failure. For example, the user could program lighting, air conditioning, and security systems using a portable computer. The network enables automation in and across buildings without a structured and managed infrastructure.

A basic functional requirement in a multi-hop computer network is routing. Although ad hoc networks have been extensively researched, and many ad hoc routing protocols have been developed, they are still limited to a few hundred nodes. Furthermore, distributed applications need basic services such as naming and service directories. As a result of their self-organization property, structured peer-to-peer (P2P) overlays such as Chord [35] or Pastry [34] are an attractive choice for the implementation of these basic services. However, running a P2P protocol on top of an ad hoc routing protocol incurs a high cost.

SCALABLE SOURCE ROUTING is an integrated solution to both problems. It creates a Chord-like, ring-

structured P2P network at the network layer. Thereby, SCALABLE SOURCE ROUTING efficiently provides network-layer connectivity, and at the same time the indirect routing primitive of a structured P2P network. In this paper, we demonstrate the feasibility and scalability of SCALABLE SOURCE ROUTING in the scenario outlined above. Compared to the standard ad hoc routing protocols AODV and DSR, SCALABLE SOURCE ROUTING delivers a substantially larger fraction of packets at a lower cost. SCALABLE SOURCE ROUTING can cope with low levels of network mobility and with moderate node failure rates.

The remainder of this paper is structured as follows: The next section gives an overview of SCALABLE SOURCE ROUTING. Its performance is evaluated in Section 3, and Section 4 discusses work related to SCALABLE SOURCE ROUTING. Finally, Section 5 concludes the paper.

2 SCALABLE SOURCE ROUTING Overview

In this section we give an overview of SCALABLE SOURCE ROUTING. We first describe the basics of indirect routing and then briefly explain SCALABLE SOURCE ROUTING's design.

2.1 Indirect Routing

Indirect routing decouples packet addresses from the network nodes. Nodes send packets to abstract destinations that the routing protocol maps to a concrete node. This level of indirection enables data-centric communication where packet addresses identify data objects instead of nodes.

Structured P2P networks like Chord enable indirect routing through an overlay at the application layer. The protocols assume a network-layer routing protocol to provide connectivity among the overlay nodes.

Chord creates a virtual address space that is cooperatively managed by all nodes participating in the overlay network. A node A manages all data objects whose addresses fall between A's address and the address of the node B whose address is the smallest of all nodes with addresses larger than A's. Node B is called A's *successor* and consequently, node A is node B's *predecessor*.

For correct routing in the overlay it is both a sufficient and a necessary condition that each node knows its correct successor. Thus, the structure of the overlay is a ring that is formed by the nodes and their pointers to the respective successors. Nodes route packets by forwarding them in increasing direction of the address space until the distance between the address of the current node and the address of the data object cannot be minimized any further. If the network is inconsistent—i.e., if some nodes do not know their correct successors—requests for data objects may get delivered to nodes that have no knowledge of the requested objects.

Besides the successor, nodes store the addresses of $O(\log n)$ additional nodes at exponentially spaced distances to reduce the average request path length from $O(n)$ to $O(\log n)$, where n is the size of the network. By choosing physically close nodes for the additional state, the total *physical* lookup path length is a constant factor longer than the shortest possible length [14].

2.2 Overview of the Design

SCALABLE SOURCE ROUTING provides indirect routing at the network layer. To this end, it integrates source routing with Chord-like routing in an address space that has the structure of a virtual ring. Nodes have self-assigned addresses that we assume to be unique and preferably uniformly distributed.

In contrast to well-known link-state and distance-vector routing protocols, SCALABLE SOURCE ROUTING does not maintain state information for all destinations in the network. A node's forwarding database is comprised of the node's physical neighbors, the node's virtual predecessor and successor, and opportunistically cached information. The protocol acquires the routes to the predecessor and successor nodes through a simple iterative process that we describe in Section 2.4. SCALABLE SOURCE ROUTING trades off shortest paths for a reduced amount of state information and therefore less maintenance overhead. Depending on the scenario, paths on average are 20%–200% longer than the shortest paths.

The protocol maintains its forwarding database as a route cache. This cache stores source routes using a least-recently-used (LRU) policy (see Section 2.5). It locks the routes to a node's virtual predecessor and successor and thereby avoids flushing them. The next section explains how SCALABLE SOURCE ROUTING routes packets on the basis of the forwarding database.

2.3 Packet Forwarding

SCALABLE SOURCE ROUTING uses two distance metrics for making its forwarding decisions. The physical distance of two nodes A and B is measured in hops, that is the length of a source route between A and B. The virtual distance of two nodes is the absolute value of the numerical difference between the nodes' addresses.

Fig. 1 shows a description of SCALABLE SOURCE ROUTING's packet forwarding in pseudocode. Data packets contain a source address, a destination address, and a source route. The included source route does not have to span the entire path from the packet source to the packet destination. The **route()** function forwards a

```
// At i: get route to next intermediate node towards dest. d
i.get_route_to_next_intermediate_node(d)
  if ( i = d ) return NULL; // destination is reached
  // retrieve nodes that reduce the virtual distance to d
  c_1 = {j ∈ i.route_cache | d_virt(j,d) < d_virt(i,d)};
  // determine smallest physical distance to retrieved nodes
  d_min = min{d_phys(i,j) | j ∈ c_1};
  // retrieve nodes with the smallest physical distance from c_1
  c_2 = {j ∈ c_1 | d_phys(i,j) = d_min};
  // choose node k from c_2 to minimize the virt. distance to d
  k = arg min_{j∈c_2} d_virt(j,d);
  return get_route(k,i.route_cache); // get route to k from cache

i.route(p) // route and forward packet p
  // get next hop in the packet's source route
  n = get_next_hop( p.source_route );
  // has the source route been completely traversed?
  if( n = NULL )
    // yes, so get a route to the next virtual hop
    r = get_route_to_next_intermediate_node( p.destination );
    // append it to the packet's source route
    p.source_route = append( p.source_route, r );
    // get the next hop of the new source route
    n = get_next_hop( p.source_route );
    // could we find a route to forward the packet?
    if( n = NULL )
      // packet reached its dest., deliver to higher layers
      deliver( p )
      return;
  // ... the source route ends at a node ≠ i
  send( n, p ); // forward packet to the next physical hop
```

Figure 1: Pseudocode of packet forwarding in SCALABLE SOURCE ROUTING

packet until the included source route ends. If the intended destination has not been reached thereafter, the last node in the source route tries to append a source route from its local cache. Again, the appended path need not end at the packet's destination, but at a node *virtually closer* to the final destination.

The **get_route_to_closest_node**() function also considers physical node proximity in its selection of the next virtual hop. In particular, if a node with address I_1 does not have a source route to the packet's destination D in its cache, it selects an intermediate destination I_2 from its cache, so that I_2 is virtually closer to D than I_1. Typically, several such nodes exist. If so, I_1 selects the one that is physically closest to I_1. If more than one of these nodes exist, the protocol chooses the virtually closest to D.

For example, in Figure 2, node 1 wants to send a packet to node 42. The packet is first forwarded to node 17 because that node is physically closest to node 1. Node 17 is preferred over node 13 since 17 is virtually closer to 42 than 13 is. For the same reasons, node 17

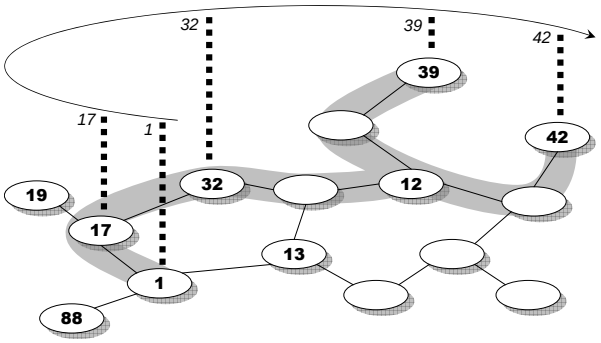

Figure 2: Illustration of the routing process

forwards the packet to node 32. Node 32 forwards the packet to its successor, node 39, which in turn forwards the packet its successor that coincides with the destination, node 42.

As a result of the indirectness of routing, a packet is delivered to the node whose address is virtually closest to the destination address D. If D identifies an existing node, the packet arrives at D.

2.4 State Maintenance

SCALABLE SOURCE ROUTING guarantees consistent routing if and only if all nodes have valid source routes to their respective virtual neighbors, that is its predecessor and successor in the address ring. A node acquires source routes to its virtual neighbors through an iterative process. As a prerequisite for this process, the nodes need to have information about their physical neighborhood. They gather this information from the periodic sending of "hello" beacons. The information is stored in a node's route cache and is locked, so that it is not discarded by the LRU strategy. In the following, we describe the process for discovering a path to a node's successor. Fig. 3 shows pseudocode for SCALABLE SOURCE ROUTING's state maintenance.

1. A node A selects a successor candidate B from its cache. In the beginning, B is one of A's physical neighbors and unlikely to be the globally correct successor of A. A sends B a *successor notification* message. It can do so since by construction the cache contains the source routes to all of its address entries.

2. Let C be the predecessor of B before receiving the *notification* message (**receive_notification**()). Upon reception, B checks whether C's address lies between that of A and B. In this case, C is a better successor candidate for A. Otherwise, A is a better successor candidate for C. In both cases, B sends *update* messages to A and C and inserts the source route from the *notification* message into its cache.

In the trivial case where $A = C$, only one message is sent. This message is called an *acknowledgment* message (see also **receive_acknowledgment**()).

The source routes for the *update* messages are obtained by concatenating B's source routes to A and C.

3. Upon reception of an *update* message (**receive_update**()), A has learned the source route to C, which either is a better successor, or a potentially better predecessor. In the former case, A sends a *successor notification* to C. In the latter case, A either agrees with C being its predecessor or A knows a better successor for C and accordingly sends either a *predecessor notification* or a *successor update* message.

For example, assume that in Fig. 2 node $A = 1$ searches its successor. The node among A's neighbors that is virtually closest to A is $B = 13$. Furthermore assume that B has a path to its correct predecessor $C = 12$. Thus, A sends a *successor notification* to B and since $A < C < B$, node B sends *update* messages to both A and C. As a consequence, A sends its next *successor notification* to C. If C knows of a node D with $A < D < C$, the process continues.

When joining a network (**join**()), a node routes a message to the node virtually closest to its address through its physical neighbors. The resulting source route is used to send an *update* message to the joining node. If the network is in a consistent state, the joining node retrieves its correct virtual neighbors in this step. Otherwise, the *update* message triggers the execution of the iterative discovery process. Therefore, the iterative process is only needed to initially bootstrap a network.

Once started, this process continues until all nodes have mutually correct virtual neighbors. We have shown in ref. [9] that the network always converges into this state within a bounded amount of time, regardless of the initial state.

Mutual correctness does not imply correctness from a global viewpoint, but in many cases the resulting network state is consistent. Global inconsistencies that cannot be locally detected for example occur in disjoint rings. To resolve global inconsistencies, all nodes whose address is larger than the address of their respective successor have to broadcast their address in the network. In each disjoint ring, there is only one such node. Since the number of disjoint rings, if any, typically is small, the broadcasts only amount to a small fraction of the control bandwidth.

To accelerate network convergence, any node forwarding *notification* and *update* messages can check their caches to detect inconsistencies, not only the endpoints of the messages.

```
i.join() // join the network
  // select virtually closest node from physical neighbors
  k = arg min_{j∈i.route_cache} d_virt(i, j);
  send_notification( get_route(i,k) ); // send notification to k

i.receive_notification(route)
  // cache route to originator
  cache_route(route);
  A = route.src;
  B = i;
  C = i.predecessor;
  // is the originator our predecessor?
  if ( A = C )
    send_acknowledgment( reverse_route(route) );
  // is the originator a predecessor of our predecessor?
  else if ( A ≺ C ≺ B )
    send_update( get_route(A), succ=get_route(C) );
      // (pred = NULL)
    send_update( get_route(C), pred=get_route(A) );
      // (succ = NULL)
  // is the originator a better predecessor?
  else if ( C ≺ A ≺ B )
    send_update( get_route(C), succ=get_route(A) );
    send_update( get_route(A), pred=get_route(C) );

i.receive_update(route, succ, pred )
  // cache routes
  cache_route(route);
  cache_route(succ);
  cache_route(pred);
  // update the successor?
  if ( pred = NULL )
    send_notification(get_route(succ.dst));
  // or update the predecessor?
  else if ( succ = NULL )
    // is the originator a better predecessor?
    if ( i.predecessor ≺ pred.dst ≺ i )
      i.predecessor = pred.dst;
      send_notification(get_route(pred.dst));
    // is the originator a better predecessor for our predecessor?
    else
      send_update(get_route(pred.dst),
       succ = get_route(i.predecessor);

i.receive_acknowledgment(route)
  // cache route to originator
  cache_route(route);
  // update our successor
  i.successor = route.src;
```

Figure 3: Pseudocode of state maintenance in SCALABLE SOURCE ROUTING

2.5 Route Cache

The protocol maintains its forwarding database as a route cache that we briefly explain in this section.

A fundamental operation implemented by the route cache in SCALABLE SOURCE ROUTING is *source route concatenation*. While forwarding a packet from its source to its destination, the intermediate virtual hops each append route fragments (see Sec. 2.3). In many cases an appended fragment contains a node that already is part of the packet's source route. The source route therefore is pruned to reduce the resulting path length to less than the sum of the individual paths' lengths. In Fig. 2, the source route from node 1 to node 42 can be pruned so that the stub path between node 12 and node 39 is not contained in the resulting path.

In practice the intermediate node performing the concatenation operation is likely to be cut out in this pruning process. For forwarding, packets therefore not only contain one source route but also a stub path that connects the most recent intermediate node to the main source route.

Even when the main route has been reached, the stub needs to be kept in the packet since route caches have to be updated when routes break. Assume that a packet contains a source route which has become invalid, for example because the next hop failed. In this case, the detecting node sends an *update* message back to the most recent intermediate node which can then delete the broken link from its cache. If the route can be salvaged, another *update* will be created to inform that intermediate node of the new route.

Optionally, *update*s may be sent during the regular routing process to inform nodes of the appended route. Assume I_1 appended a route to I_2 and I_2 appended a route to the destination D. Then I_2 can ask D to update I_1 so that in future I_1 has a source route to D.

However, upon reception of an *update*, only the part between the intermediate node and the destination is entered into the cache, since only that part has just been traversed by the packet.

Together with the *update*s indicating broken links, this allows SCALABLE SOURCE ROUTING to work without regular probing. When a source route has become invalid, it remains in the cache until it is used and then causes *update* messages. As a consequence, SCALABLE SOURCE ROUTING does not create any control messages in the absence of payload traffic.

This can force re-bootstrapping of an already deployed network if a network with mobile nodes or node churn remains idle for a long period. The caches become stale and all virtual neighbor source routes must be iteratively rebuilt. Thus, depending on the actual network characteristic, it can be beneficial to periodically probe the virtual neighbors.

3 Evaluation

SCALABLE SOURCE ROUTING provides both, direct any-to-any routing in unstructured networks, and the semantics of a structured peer-to-peer overlay. In the remainder of this section, we report simulation results measuring SCALABLE SOURCE ROUTING's performance for routing in MANETs. In particular, we compare its performance to that of AODV and DSR using their GloMoSim [1] implementation as benchmark.

Comparing SCALABLE SOURCE ROUTING's performance to that of state-of-the-art structured peer-to-peer overlays in MANETs would for example mean to deploy Chord on top of AODV. Obviously, such an overlay approach would increase the delay and network load while, by construction, the performance of SCALABLE SOURCE ROUTING would not differ from the pure routing scenario. We thus restrict this evaluation to the more informative case of routing alone.

Owing to the scenario of embedded devices, the route cache size was limited to 255 nodes only, that is, the union of all nodes in the cached source routes contained at most 255 distinct nodes.

We simulated networks with 50 – 125,000 nodes, where the area is increased likewise so that the networks have a constant density of 50 nodes per square kilometer. With the chosen radio model (see below) this corresponds to 9.8 nodes per radio range disk. With this node density, there is a high probability for the network to be connected with potentially a few unconnected nodes [3, 39]. Other node densities were studied separately (see below).

It is well known that pure ad-hoc networks are limited in their capacity [15]. Thus, in most of the simulations, we equipped 5% of the nodes with additional 1 MBit/s point-to-point links. This models fixed nodes with a wireline communication infrastructure, for example powerline, cable-modem or DSL connections. Links between these nodes are chosen according to the Barabási-Albert model [2], a general model for many computer networks [26, 7]. This is a so-called preferential connectivity model that is known to produce a small-world topology.

Nodes move according to a random direction model with a maximum node velocity of 1 m/s, the velocity of a pedestrian. Other maximum velocities were studied separately (see below).

The traffic model for our simulation is derived from a scenario where SCALABLE SOURCE ROUTING is used for a fully decentralized directory service: Every minute each node sends a 500 byte packet to a randomly selected destination. Other packet sizes were simulated, too, but found to have little influence on the protocol's performance. The results therefore are not reported here. For bootstrapping, all nodes are switched on with an empty cache at equally distributed points in time dur-

ing the first minute of the simulation.

If not otherwise stated, each simulation run spans one hour of simulated time. During the run, we measure the delivery rate, the end-to-end delay for the delivered packets, and the network load created by the payload, overhead, and control messages. The plotted values are averages of one-minute intervals, or, in case of time series, averages of 10 simulation runs. For the analysis of the runs, we delete an initial transient period of 10 min to allow the network to settle. Additional simulations compare the route stretch, which is the average ratio of the achieved path length to the shortest path length, and the traffic distribution.

3.1 Radio- and Link-Layer Model

Packet level simulations of large networks are difficult since the detailed simulation of the entire protocol stack requires sufficiently fast simulation machines. Especially, we found the GloMoSim IEEE 802.11 model to be incapable of simulating sufficiently large networks for our study. Therefore, we resorted to a mixed-mode simulation that models PHY and MAC with the help of a fluid model based on experimental results from the literature:

Kim and others [23] show that PHY and MAC of IEEE 802.11 wireless LANs can be modeled by a sequence of collision, transmission, and idle periods such that the system's throughput is a function of the frame size and the number of nodes M simultaneously attempting to access the medium. Moreover, the authors found the throughput to be almost independent of M when RTS/CTS was applied.

He and others [17] show how the throughput depends on the number of nodes in the transmission range, the carrier sense range of the sender, and the number of hidden nodes. In absence of hidden nodes, the entire throughput of all nodes is independent of the number of nodes attempting to access the medium [23]. In the presence of hidden nodes, the throughput drops when the offered load is increased beyond a certain threshold, where the extent of this drop depends on the node density. However, with the chosen node density, the omission of considering hidden nodes has only a minor effect [17]. Simulation and measurement results are in accordance with these results [8].

Following these results, we based our PHY and MAC simulation on a fluid model using the IEEE 802.11 system parameters from [23]. The reception power threshold was chosen such that the radio range was 250 m.

3.2 Comparison with AODV and DSR

In the following sections, we compare SCALABLE SOURCE ROUTING to AODV and DSR with respect to their scalability and robustness in face of mobility and node churn.

3.2.1 Varying Network Size

Figs. 4–6 show the results for a pure mobile ad-hoc network scenario, a hybrid MANET with 1% fixed-wired nodes, and a hybrid MANET with 5% fixed-wired nodes. Since the GloMoSim implementations of AODV and DSR are not optimized for large networks, they frequently broke down in simulations with more than a few hundred nodes due to internal problems, such as memory leaks and array overflows. However, since we intended to use GloMoSim as a widely accepted benchmark, we refrained from more than basic modifications of the simulation code. Hence, we included those runs that simulated at least 20 minutes, but could not acquire AODV and DSR results for even larger networks.

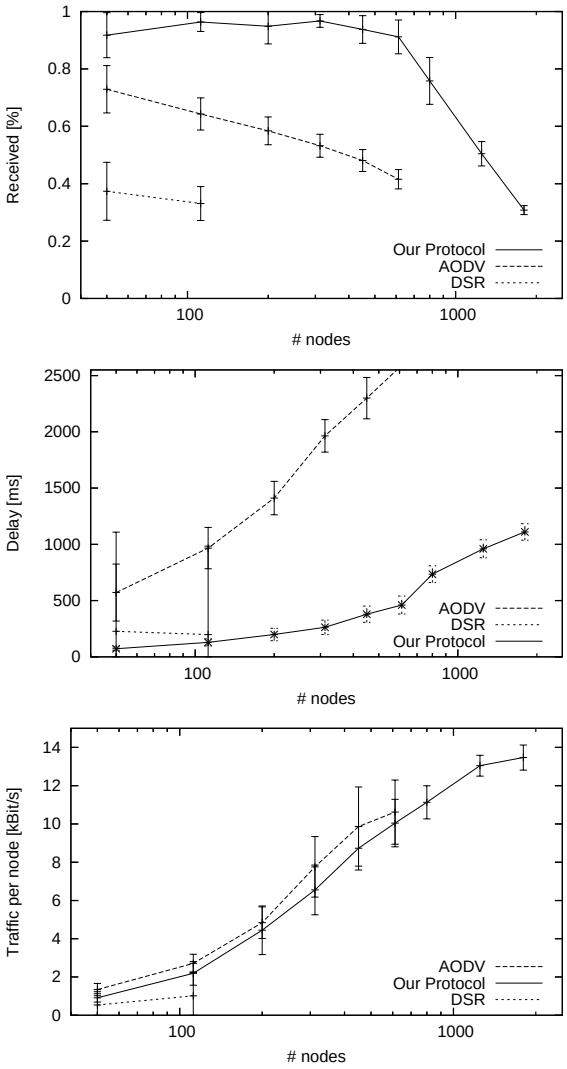

Figure 4: Pure MANETs

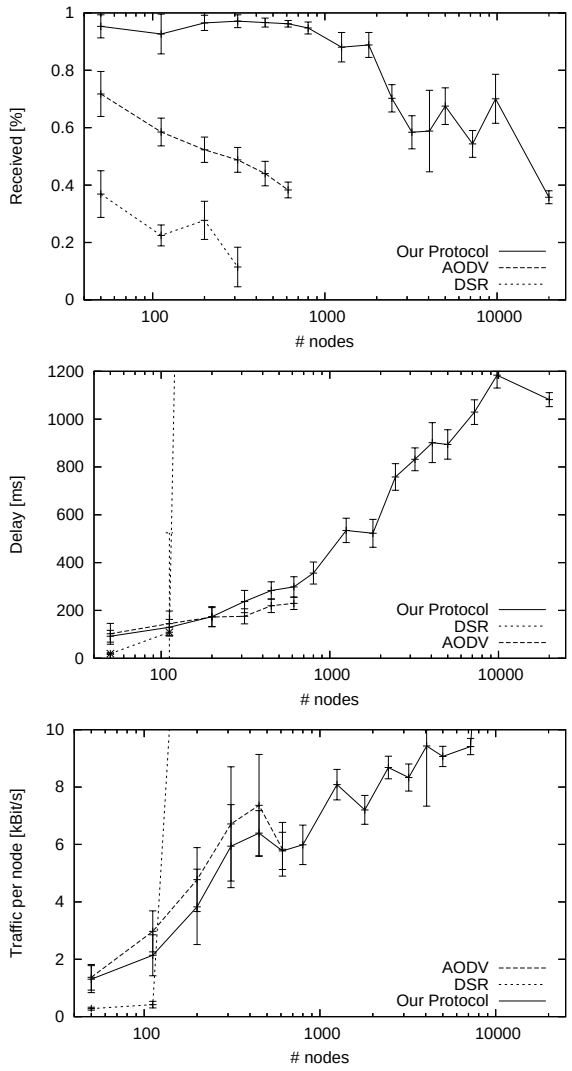

Figure 5: Hybrid MANETs with 1% fixed-wired nodes

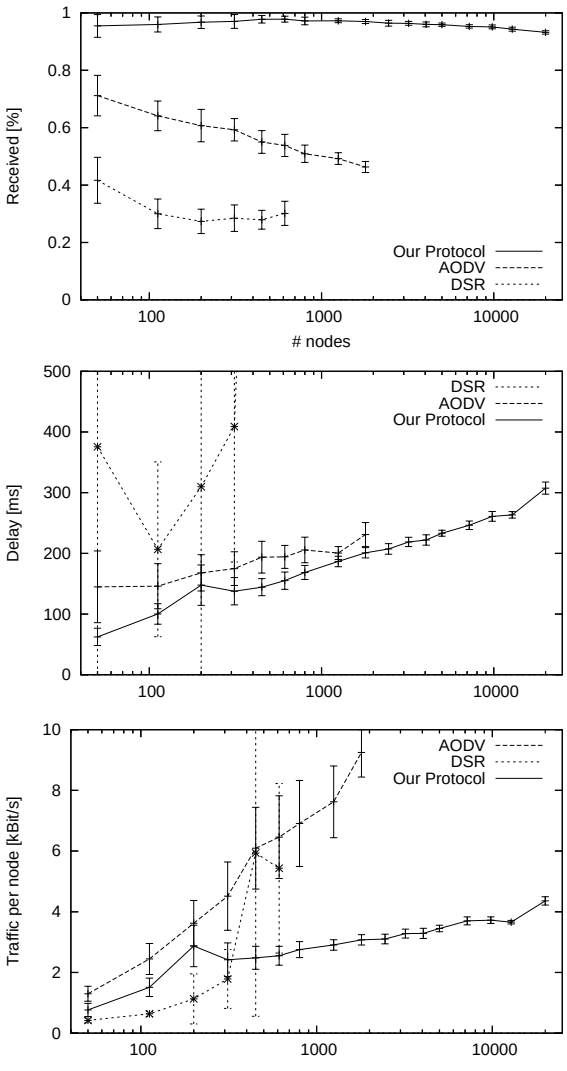

Figure 6: Hybrid MANETs with 5% fixed-wired nodes

Nevertheless, the results already show that SCALABLE SOURCE ROUTING is capable of routing a large fraction of the packets in large networks, while AODV and DSR are not. In the pure MANET scenario, increasing the network size up to 600 nodes does not have an effect on the delivery ratio of SCALABLE SOURCE ROUTING, which successfully routes about 90% - 95% of all messages (cf. Fig. 4). AODV and DSR, in contrast, have delivery ratios that continuously drop to less than 50%. In the hybrid MANET scenarios, the delivery ratios of AODV and DSR are comparable to those of the pure MANET scenario (cf. Figs. 5 and 6). SCALABLE SOURCE ROUTING, however, can maintain a high delivery ratio for much larger networks, namely up to 1200 nodes in the scenario with 1% fixed nodes and up to more than 12,000 nodes with 5% fixed nodes. (See below for even larger networks.)

Figs. 4–6 (middle) show the average end-to-end delays for the three scenarios. Note that the drop in delivery ratios for SCALABLE SOURCE ROUTING coincides with a rise in the round trip time. Higher round trip times indicate contention on the wireless links. Both, the higher round trip times and the packet loss caused by overflowing queues in a congested network affect the ability of SCALABLE SOURCE ROUTING to quickly adopt to changes in the network. In the pure MANET case, SCALABLE SOURCE ROUTING can keep the delay below 400 ms for networks of up to 600 nodes. AODV and DSR have significantly larger delays. With DSR, the delays are also erratic, leading to huge error bars.

The reason for the erratic delays is that the distribution of delays is heavy-tailed. Fig. 7 shows the distribution of the delays for the 5% fixed nodes scenario and a total of 612 nodes. While SCALABLE SOURCE ROUTING follows a power-law distribution with a single distribution parameter, AODV and DSR exhibit a slightly

different behavior: both follow two different distributions, one for values below about 40 ms and a different one for the larger delays.

The reason for the two different parameters is that AODV and DSR need to acquire or produce state in the network before they can route the actual payload packet. Thus we can distinguish the case where that state already exists from the case where the state needs to be established. In the former case, AODV and DSR lead to a lower delay than SCALABLE SOURCE ROUTING. In the latter case, they lead to a much higher delay. The different cases correspond to the different slopes in the AODV and DSR delay distribution in the double-logarithmic plot in Fig. 7.

Coming back to the relation of delay and delivery ratios, we see that in all cases SCALABLE SOURCE ROUTING can maintain its high delivery ratio as long as the delay stays below 500 ms. We also see that the delay increases roughly logarithmically with the network size. This is caused by the growing path lengths (not shown). Such a scaling behavior of average node distances is typical for small-world networks [38, 4].

Besides its high delivery ratio, SCALABLE SOURCE ROUTING also keeps the network traffic low. Figs. 4–6 (bottom) show the average traffic per node. As can be best seen in the pure MANET case, in small networks SCALABLE SOURCE ROUTING produces a similar traffic volume as AODV. DSR achieves slightly lower traffic volumes. In the hybrid cases, however, SCALABLE SOURCE ROUTING is able to keep the traffic volume low while the AODV and DSR traffic quickly drive the network into congestion, presumably because of their intensive use of flooding.

3.2.2 Varying Node Density

In order to complete the comparison, we have also studied the effect of varying node densities and maximum node velocities. Simulations varying the message size did not yield any particularly remarkable effect and are thus not reported here. Due to the frequent problems with the DSR simulation, we compared SCALABLE SOURCE ROUTING only to AODV.

Fig. 8 shows the delivery rate in a 450 node pure MANET where the size of the network has been varied to yield node densities between 5 and 22 nodes per radio disk. As expected, the probability of successful delivery drops rapidly when the node density falls below 10 nodes since the network partitions [39]. Conversely, higher node densities show no effects on the delivery ratio. We did however not explore very high node densities since with our traffic model they would immediately cause congestion in the network.

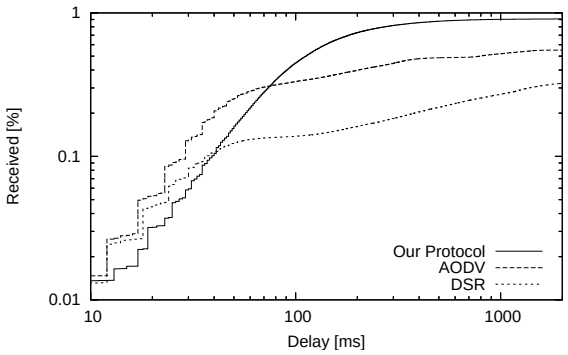

Figure 7: CDF of the end-to-end delays

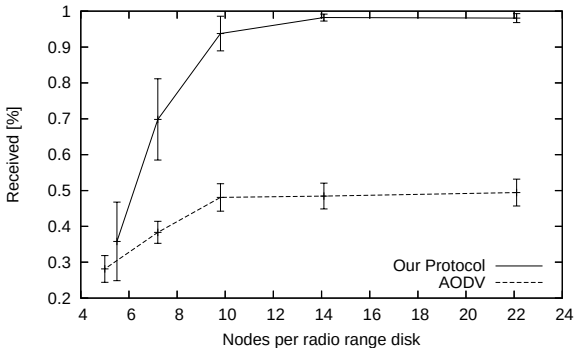

Figure 8: Varying the node density, max. velocity 1 m/s

3.2.3 Varying Node Mobility

Figs. 9 show the effects of varying the maximum node velocity in our 450 node pure MANET model. As can be seen, SCALABLE SOURCE ROUTING is most effective at low node velocities. Up to about 1 m/s, the velocity of a pedestrian, the delivery rate can be sustained at more than 90%. When the node speeds are further increased, the delivery rate significantly drops. For maximum node velocities of more than about 2 m/s, SCALABLE SOURCE ROUTING performs worse than AODV. The reason is that the proactive aspects of SCALABLE SOURCE ROUTING cannot cope with the speed of route changes while the reactive AODV protocol can obtain routes, albeit at the expense of a high delay. This is different from the congestion collapse in a large pure MANET, as can be seen from the comparatively low delay for SCALABLE SOURCE ROUTING and the low traffic load (cf. Figs. 9 mid and bottom).

Finally, we have compared the paths lengths produced by SCALABLE SOURCE ROUTING and AODV to the shortest paths as they could be produced by an ideal omniscient routing instance (Fig. 10). As expected, SCALABLE SOURCE ROUTING does not yield shortest paths.

Nevertheless, this comparison yields some remarkable insights, too. Since AODV discovers routes by

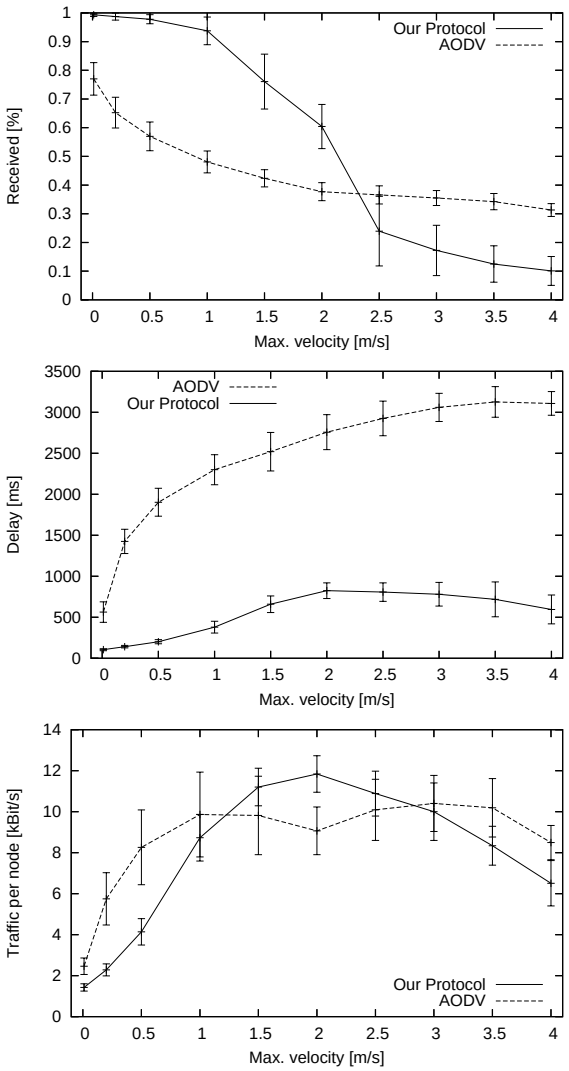

Figure 9: Varying node mobility in a pure MANET

means of flooding, the resulting paths are shortest paths, or, if repair mechanisms apply, close to shortest paths. The larger the distance, the higher the probability that AODV fails to deliver the path. If in the shown simulation, SCALABLE SOURCE ROUTING would give up after a maximum of eight hops, the corresponding line would level with AODV's line.

3.3 Challenging Scalability: Bootstrapping and Churn Resistance

So far, we have compared SCALABLE SOURCE ROUTING to AODV and DSR. The simulation results demonstrated that in large MANETs SCALABLE SOURCE ROUTING can route any-to-any traffic significantly better, provided a few additional fixed-wired links resolve the general problem of a low scalability bisection bandwidth. In this section, we further challenge SCALABLE

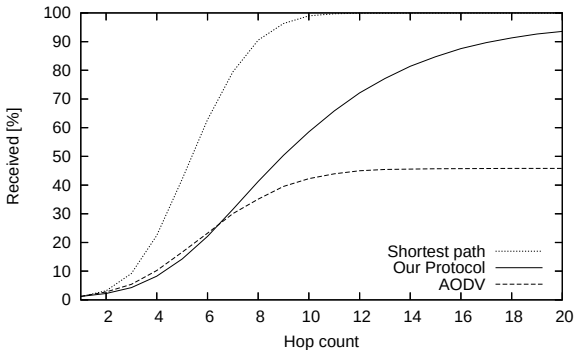

Figure 10: Paths lengths for an 800 node hybrid MANET with 5% fixed-wired nodes

SOURCE ROUTING . In particular, we evaluate three important issues: the potential creation of traffic hot spots, the bootstrapping performance, and SCALABLE SOURCE ROUTING's resistance against node churn. These issues are less significant with protocols like AODV and DSR that discover state on demand by flooding. Since SCALABLE SOURCE ROUTING proactively builds up state in the individual nodes, these questions are important here.

The iterative process of appending and pruning source routes might be thought to lead to the traffic following well-worn paths. This would lead to a traffic concentration at a few hot-spot nodes. To clarify this question, we measured the traffic distribution over the nodes for a 9,800 node hybrid MANET with 5% fixed-wired nodes. Fig. 11 shows rank plots of the traffic distribution on the nodes, averaged over 10 min at the end of a run simulating one hour. The lower axis enumerates all nodes with a wireless interface only. The upper axis enumerates the nodes that also have additional fixed-wired interfaces.

The measurement leads to two observations:

1. Nodes with additional fixed-wired connections attract a significantly higher fraction of the entire traffic than the mobile nodes that only have a wireless interface. This is a favorable feature, since it shows that SCALABLE SOURCE ROUTING can effectively use the additional bandwidth provided by the fixed-wired links.

2. Within the two node classes, the traffic is distributed quite evenly. 80% of the mobile nodes have traffic loads between 0.533 and 3.995 kBit/s, compared with 3.632 and 71.239 kBit/s for the fixed-wired nodes. This was calculated by excluding the top and bottom 10 percentiles. The larger span for fixed-wired nodes is due to the fact that nodes with many fixed-wired links bear a higher traffic share. If we break the traffic down to the individual link, the distribution gets more uniform.

Fig. 12 shows a measurement comparing SCALABLE SOURCE ROUTING to a shortest path routing algorithm. We simulated a large backbone with 16,000 fixed-wired

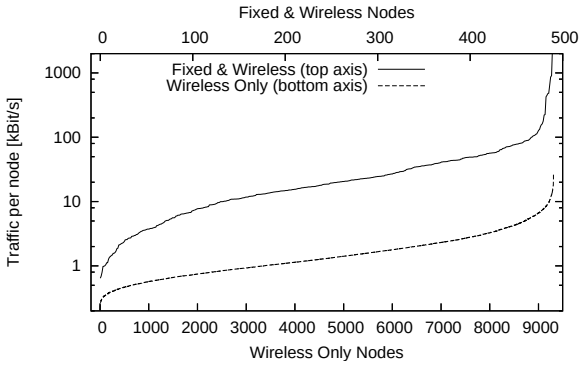

Figure 11: Rank plot of traffic distribution

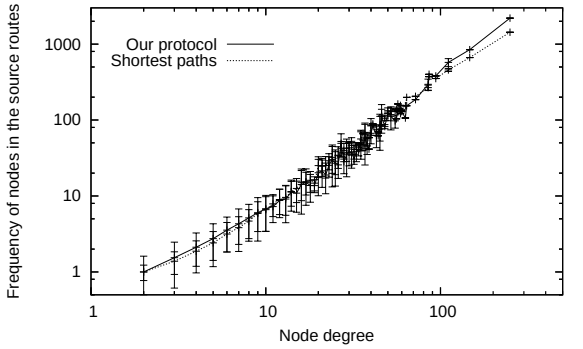

Figure 12: Node frequency

nodes and compared the frequence of the nodes in the resulting paths with that of shortest paths. As can be seen from the figure, SCALABLE SOURCE ROUTING does not create hot-spots other than those caused by the actual network topology.

Figs. 13 show the behavior of SCALABLE SOURCE ROUTING under node churn. The figures show the delivery rate, delay and traffic for a 9,800 node hybrid MANET with 5% fixed-wired nodes and a maximum node velocity of 1 m/s. Each plotted value is the average of 10 simulation runs. In addition to the random direction mobility model, the network was subject to node churn during the time 0–30 minutes and 60–90 minutes. This means that during these periods, nodes died according to a Poisson process with half-life times of 5, 10 and 20 minutes respectively. In order to keep the total number of nodes constant, nodes were immediately reincarnated, that is they reappeared with a new address and empty state at a random position in the network. If the respective node was fixed-wired, it became again fixed-wired with the same degree as the node that died, but of course the links connected to other nodes. Thus, unlike the Barabási-Albert graph, the resulting fixed-wire network not necessarily is connected.

The assumption that all nodes are subject to the same node churn is atypical. In practice, most of the nodes will be stable for long periods of time while a few nodes join the network for a short period of time. In our model, the life times of the nodes are averaged to be identical.

Fig. 13 (top) shows the delivery rate. For a moderate node churn rate with a half-life time of 20 min the network converges within 5–10 minutes to a delivery rate of about 60%. With extremely high churn rates, that is when the half-life time is only 10 or 5 minutes, the delivery rate is smaller. But when the churn is switched off, the network quickly recovers in all cases. After about 10 min, it is back to its normal state with about 95% delivery rate.

Close inspection of the first churn period shows that the delivery rate for the 20 min case peaks at the end of the bootstrapping process and then slightly and slowly drops again. The reason is that nodes do not immediately detect when source routes break. In the beginning, the nodes are concerned with the discovery of their virtual neighbors anyway. Thus all virtual neighbor routes are fresh. Once the nodes have found their consistent state, the control message traffic caused by the neighbor discovery process drops and route breaks have a higher chance to be unnoticed for a while.

Fig. 13 (bottom) shows the traffic per node. In the low and medium churn case, the traffic pattern is identical within the error bars. Astonishingly at first sight, the high churn rate case creates less traffic. The reason is that here packets have to be discarded relatively quickly. Accordingly, the delay for the successfully delivered packets is lower than in the medium and high churn cases (cf. Fig. 13 middle).

Finally, we have made a few simulations with a 125,000 node hybrid MANET to explore the protocol's behavior in very large networks. Figs. 14 shows the delivery rate (top) and the delay (bottom) for two different runs with differing line speed on the fixed-wired links. In the 1 MBit/s case, the network gets congested, as can be seen from the extremely high delay. In that case, convergence stops at about 80%. If the bandwidth on the fixed-wired links is increased to 10 MBit/s, the delays peak at only about 1 sec during the initial convergence of the virtual ring. Afterwards, they drop to 250 ms and the delivery rate reaches 90%.

Since SCALABLE SOURCE ROUTING caches routes that have been created by the iterative process, all but the first packet of a connection travel on a significantly shorter path. Figs. 14 (bottom) also shows the delays for a second packet that is sent to the just discovered destination.

3.4 Real-World Evaluation

For all new protocols, real-world world experience is highly desired. Therefore, we implemented SCALABLE

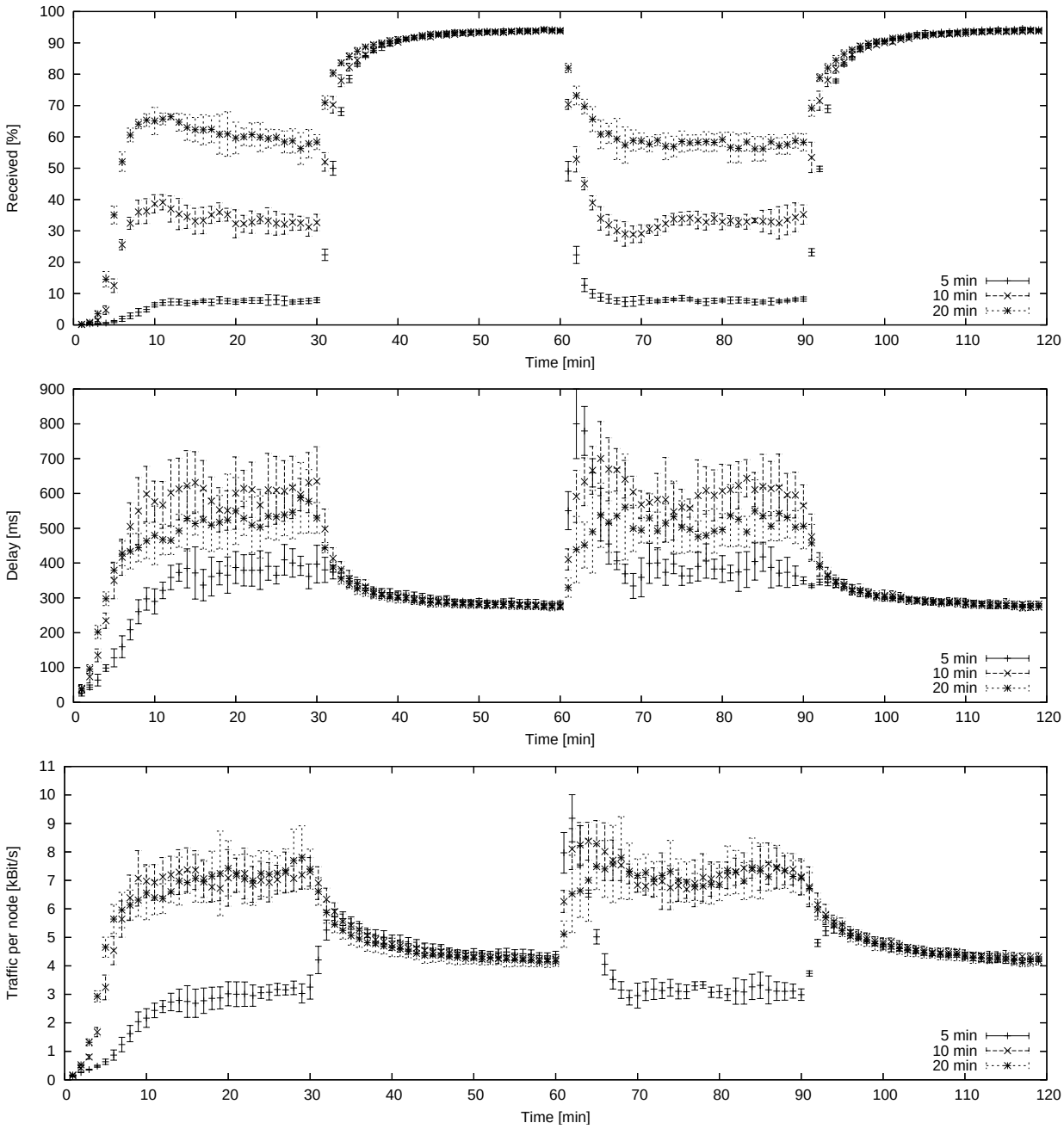

Figure 13: 9800 node hybrid MANET with churn periods

SOURCE ROUTING in a way that it can run in the simulator as well as in real systems. This implementation was ported to general Linux systems. We tested it on a network of popular embedded MIPS IEEE 802.11 routers. These routers do not only provide the fixed-wired access for machines running SCALABLE SOURCE ROUTING, but can also act as a gateway to unmodified IPv6 hosts.

In all tested cases, SCALABLE SOURCE ROUTING performed as expected and was able to handle realistic workloads, even when the topology of the network changed. After this qualitative evaluation, the next steps are to scale up our testbed in order to show the scalability of our approach not only by simulation but also in real-world systems.

4 Related Work

Related work can be broadly classified into routing protocols for self-organizing systems (that is, future architectures of the Internet, mobile ad hoc networks, and

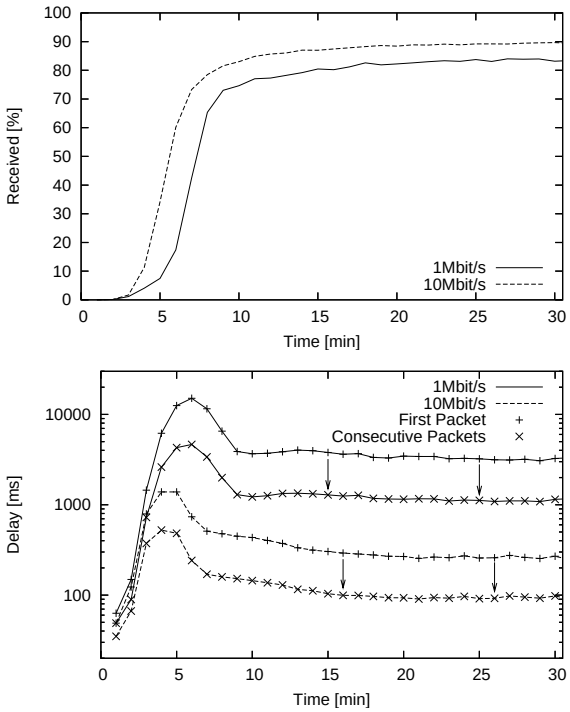

Figure 14: Variation of line speed for fixed-wired links

sensor networks) and protocols for indirect routing in MANETs.

In the current architecture of the Internet, addresses are dependent on the physical location of the nodes. This prohibits users from running long-lived application connections while roaming between different subnetworks. For example, TCP connections break if the address of an endpoint changes. In addition to that, since addresses cannot be used to identify nodes, they also cannot be used to locate a mobile node.

Mobile IP [29] was designed to enable roaming in the Internet. A *mobile node* is assigned a permanent *home address* by its *home agent*, that is a router in the mobile node's home network. When roaming to a different network, the mobile node temporarily gets assigned a *care-of address* by a *foreign agent*. For packet delivery, the mobile node is identified by its home address. Packets are transparently tunneled from the home agent to the foreign agent. By sending packets over sub-optimal routes, tunneling increases both end-to-end delays and the network load.

Newer approaches therefore employ a different form of indirection for handling roaming. Instead of tunneling, they use dynamic location services. There, fixed node identifiers are transparently resolved into location-dependent addresses. For scalability, the directory storing the identifier-to-address mappings has to be decentralized. A number of proposed approaches use structured P2P protocols for implementing their directories and their routing primitives [11, 37]. Location directories have to be updated to provide correct mappings, and the resolution of identifiers into addresses incurs a delay. While SCALABLE SOURCE ROUTING is based on structured P2P principles and employs fixed node identifiers, it eliminates the need for a separate location service. The node identifiers are also used as addresses. Nonetheless, the locality-awareness of the network established by SCALABLE SOURCE ROUTING diminishes the cost of location-independent addressing. The UIP [13] also is independent of a location service, but the authors do not address and evaluate the aspect of network dynamics.

Mobile ad hoc networks form a class of self-organizing networks which are deployed without a supporting infrastructure. Nodes in a MANET communicate wirelessly and act as routers for the other nodes. The first routing protocols for these networks were simple extensions of fixed-network routing protocols. They assume flat routing where all nodes are grouped in one logical sub-network with no hierarchy. *Destination-Sequenced Distance Vector* (DSDV) routing [30] enables distance-vector routing in mobile environments. With DSDV, every node proactively maintains routes to all other nodes on the network. Route changes due to mobility are announced through network-wide broadcasts. The high amount of $O(N)$ routes maintained per node and the high overhead generated by DSDV limit the protocol's applicability to small networks. SCALABLE SOURCE ROUTING proactively maintains only $O(1)$ routes per node.

Johnson proposed to reduce the amount of state for routing in a MANET by only considering currently needed routes [20]. The idea was extended to *Dynamic Source Routing* (DSR) [21]. DSR separates routing in a discovery and a maintenance phase. Routes are stored as source routes and they are discovered by flooding. The inherent scalability limitation through flooding is slightly alleviated by the extensive use of caching. Similar to DSR, Perkins and others extended DSDV for on-demand routing [31]. Unlike a proactive protocol, on-demand routing incurs a delay before two nodes can communicate. If the communication pattern consists of many short-lived connections, this delay is disadvantageous. Also, the performance of on-demand protocols degrades in scenarios where many routes need to be simultaneously maintained.

A variety of other approaches to reducing the amount of protocol state and overhead of flat routing has been proposed. For example, with *Fisheye State Routing* [27], routes to nearby nodes are updated more frequently than routes to distant nodes. Haas [16] proposed a hybrid routing protocol, the *Zone Routing Protocol* (ZRP). ZRP proactively maintains routes to nearby nodes, and discovers routes to distant nodes on demand. The *Opti-*

mized Link State Routing (OLSR) [19] protocol reduces the overhead of link-state routing through multipoint relaying. Only a subset of all links is advertised, and the efficiency of flooding is increased by only a subset of all nodes retransmitting messages.

As an alternative to flat routing, Kleinrock and others [24] analyzed hierarchical routing for large networks. By grouping nodes into areas and assigning addresses to this area hierarchy, the size of routing tables can be reduced. The price of state reduction is an increase in the average path length. Tsuchiya [36] extended the concept of an area hierarchy to the landmark hierarchy. There, node addresses are assigned on the basis of relative positions to a set of landmark nodes. A landmark hierarchy is easier to dynamically configure and maintain. The routing performance of an area hierarchy is sensitive to the selection of areas, whereas the landmark hierarchy is less sensitive to the placement of landmarks. In addition to that, it is more complex to maintain an area of nodes instead of a landmark. LANMAR [28] and L^+ [6] are examples of landmark routing protocols for MANETs. Unlike SCALABLE SOURCE ROUTING, landmark routing protocols make dynamic address assignments and therefore need an additional location service.

The Safari protocol [10] is closely related to landmark routing. The protocol maintains a self-organized hierarchy, and routing is done by a combination of proactive and reactive mechanisms. The hierarchy is maintained by periodically sending beacons, and packets are sent along reverse beacon paths towards their destination. Routes within a destination cell are discovered by flooding. Safari uses distributed hashing for its location service. The mappings from node identifiers to node addresses are replicated in the address space of the self-organized hierarchy.

With *geographic routing*, no routes have to be maintained at all. Therefore, it has ideal scaling properties. Nodes only need to know the physical locations of their neighbors, and packets are greedily forwarded into the direction of the destination's physical location. However, for enabling geographic routing, all nodes have to be equipped with satellite navigation receivers (e.g., GPS). An example of geographic routing protocols is *Greedy Perimeter Stateless Routing* (GPSR) [22]. For practical purposes, geographic routing requires a location service for mapping the location-independent address of a node to its current geographic coordinate. The *Grid Location Service* [25] is an example of a scalable, distributed location service. Greedy geographic routing requires a planarized network graph. Therefore, it is unsuitable for our scenario of hybrid MANETs where additional wired links provide shortcuts. The Geographic Hash Table (GHT) [33] enables indirect routing on the basis of GPSR. As a result of that, the GHT requires the nodes to access to physical location information.

Wireless sensor networks (WSNs) are large-scale deployments of resource-constrained nodes. As a result of their size and deployment scenarios, WSNs need to be self-organizing. Protocol designs have to address the limited communication and computational capabilities as well as the limited memory and battery capacity of nodes. In general, sensor nodes are assumed to be stationary, but to be prone to failure. Early applications of WSNs were concerned with environmental data collection. These applications require data-centric routing primitives which support aggregation. Recent proposals also necessitate point-to-point routing. To this end, Fonseca and others [12] proposed *Beacon Vector Routing* (BVR). BVR combines randomly selected landmarks to infer virtual addresses with geographic forwarding in the virtual coordinate space, and therefore requires a location service. Geographic routing is augmented with scoped flooding to ensure packet delivery in situations where the greedy strategy is insufficient.

The idea of using structured P2P protocols for routing in a MANET has independently been proposed by several groups. DPSR [18] is a cross-layer approach for using the structured P2P overlay Pastry as a network-layer routing protocol. It integrates Pastry with the MANET routing protocol DSR [21]. DPSR addresses nodes by fixed identifiers. The Ekta protocol [32] enhances DPSR with an indirect routing primitive. In contrast to SCALABLE SOURCE ROUTING, both DPSR and Ekta use flooding to discover routes.

MADPastry [40] integrates Pastry with AODV for indirect routing. For adapting the overlay to the underlying network graph, the protocol uses location-dependent addressing. The addresses are assigned using a mechanism called random landmarking. Landmarks have a guiding function in routing, but the protocol reverts to flooding if the destination node cannot be found. As a result of the location-dependency of addresses, key-to-node mappings frequently change due to mobility. This causes additional overhead which SCALABLE SOURCE ROUTING avoids by using fixed identifiers.

CrossROAD [5] provides an indirect routing primitive through a simple extension of OLSR. With OLSR, each node maintains routes to all other nodes in the network. The CrossROAD protocol hashes the IP addresses in the routing table to create a full mesh overlay topology. It has the same scalability limitations as OLSR and is only suitable for small networks. The authors only evaluated their approach in a small test-bed of eight nodes.

5 Conclusion

Scalable routing in unmanaged, unstructured, and self-organizing networks is a basic requirement for scenar-

ios where consumers use many resource-limited devices that communicate with each other. We have argued that, although ad hoc networks have been extensively researched, in the scenarios that we envision, MANET protocols are not practical due to their scalability problems.

We have proposed SCALABLE SOURCE ROUTING, an indirect routing scheme based on ideas from the Chord overlay network. SCALABLE SOURCE ROUTING provides scalable any-to-any communication in large unstructured networks as well as the semantics of a structured peer-to-peer overlay. Thus, SCALABLE SOURCE ROUTING can serve as a basis for distributed applications in the scenario sketched above.

We have evaluated SCALABLE SOURCE ROUTING by means of simulations and have found that it delivers a substantially larger fraction of packets than AODV and DSR, while achieving a lower average delay. In hybrid scenarios, SCALABLE SOURCE ROUTING can efficiently use fixed-wired links and thereby avoid congestion collapse. With low node mobility and low node churn, SCALABLE SOURCE ROUTING achieves delivery rates of more than 90% in networks with up to 125,000 nodes. The code used in the simulations has also been ported to a popular IEEE 802.11 MIPS-Linux router creating a working real-world version of SCALABLE SOURCE ROUTING .

References

[1] L. Bajaj, M. Takai, R. Ahuja, K. Tang, R. Bagrodia, and M. Gerla. GloMoSim: A Scalable Network Simulation Environment. Technical Report 990027, UCLA, Computer Science Department, May 1999.

[2] A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286:509–512, October 1999.

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

[4] B. Bollobas. *Random Graphs*. Cambridge University Press, 2001.

[5] E. Borgia, M. Conti, F. Delmastro, and E. Gregori. Experimental Comparison of Routing and Middleware Solutions for Mobile Ad Hoc Networks: Legacy vs Cross-Layer Approach. In *Proceedings of the ACM SIGCOMM '05 Workshops*, pages 82–87, Philadelphia, PA, USA, Aug. 2005.

[6] B. Chen and R. Morris. L^+: Scalable Landmark Routing and Address Lookup for Multi-hop Wireless Networks. Technical Report 837, MIT LCS, Cambridge, MA, USA, Aug. 2002.

[7] Q. Chen, H. Chang, R. Govindan, S. Jamin, S. J. Shenker, and W. Willinger. The Origin of Power Laws in Internet Topologies Revisited. In *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, volume 2, pages 608 – 617, June 2002.

[8] S. Choi, K. Park, and C. kwon Kim. On the performance characteristics of WLANs: revisited. In *Proceedings of the ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 97 – 108, Banff, Alberta, Canada, June 2005.

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

[10] 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.

[11] J. Eriksson, M. Faloutsos, and S. Krishnamurty. PeerNet: Pushing Peer-to-Peer Down the Stack. In *Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS '03)*, Claremont Hotel, Berkeley, CA, USA, Feb. 2001. Springer Verlag.

[12] R. Fonseca, S. Ratnasamy, J. Zhao, C. T. Ee, D. Culler, S. Shenker, and I. Stoica. Beacon Vector Routing: Scalable Point-to-Point Routing in Wireless Sensornets. In *Proceedings of 2nd Symposium on Networked Systems Design and Implementation*, Boston, MA, U.S., May 2005.

[13] B. Ford. Unmanaged Internet Protocol. *ACM SIGCOMM Computer Communications Review*, 34(1):93–98, Jan. 2004.

[14] K. Gummadi, R. Gummadi, S. Gribble, S. Ratnasamy, S. Shenker, and I. Stoica. The Impact of DHT Routing Geometry on Resilience and Proximity. In *Proceedings of the SIGCOMM 2003 conference*, pages 381–394. ACM Press, 2003.

[15] P. Gupta and P. R. Kumar. The capacity of wireless networks. *IEEE Transactions on Information Theory*, IT-46(2):388–404, Mar. 2000.

[16] Z. J. Haas and M. R. Pearlman. ZRP: A Hybrid Framework for Routing in Ad Hoc Networks. In C. E. Perkins, editor, *Ad Hoc Networking*, pages 221–253. Addison-Wesley, 2001.

[17] J. He, D. Kaleshi, A. Munro, Y. Wang, A. Doufexi, J. McGeehan, and Z. Fan. Performance investigation of IEEE 802.11 MAC in multihop wireless networks. In *Proceedings of the International Workshop on Modeling Analysis and Simulation of Wireless and Mobile Systems*, pages 242 – 249, Montréal, Quebec, Canada, October 2005.

[18] Y. C. Hu, S. M. Das, and H. Pucha. Exploiting the synergy between peer-to-peer and mobile ad hoc networks. In *Proceedings of HotOS-IX: Ninth Workshop on Hot Topics in Operating Systems*, Lihue, Kauai, Hawaii, May 2003.

[19] P. Jacquet, P. Mühlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot. Optimized Link State Routing Protocol for Ad Hoc Networks. In *Proceedings of the 2001 IEEE International Multi Topic Conference (IEEE INMIC)*, pages 62–68, Lahore, Pakistan, Dec. 2001.

[20] D. B. Johnson. Routing in ad hoc networks of mobile hosts. In *Proceedings of Workshop on Mobile Computing Systems and Applications*, Santa Cruz, CA, U.S., 1994.

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

[22] B. Karp and H. T. Kung. GPSR: Greedy Perimeter Stateless Routing for Wireless Networks. In *Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom 2000)*, pages 243–254, Boston, MA, August 2000.

[23] H. Kim and J. C. Hou. A fast simulation framework for IEEE 802.11-operated wireless LANs. In *Proceedings of the Joint international conference on Measurement and modeling of computer systems*, pages 143 – 154, New York, NY, USA, June 2004.

[24] L. Kleinrock and F. Kamoun. Hierarchical routing for large networks. *Computer Networks*, 1:155–174, 1977.

[25] J. Li, J. Jannotti, D. S. J. D. Couto, D. R. Karger, and R. Morris. A Scalable Location Service for Geographic Ad Hoc Routing. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking (MobiCom '00)*, pages 120–130, Boston, MA, USA, 2000.

[26] A. Medina, I. Matta, and J. Byers. On the Origin of Power Laws in Internet Topologies. *ACM SIGCOMM Computer Communications Review*, 30(2):18–28, April 2000.

[27] G. Pei, M. Gerla, and T.-W. Chen. Fisheye State Routing: A Routing Scheme for Ad Hoc Wireless Networks. In *Proceedings of the IEEE International Conference on Communications*, pages 70–74, New Orleans, LA, June 2000.

[28] G. Pei, M. Gerla, and X. Hong. Lanmar: Landmark routing for large scale wireless ad hoc networks with group mobility. In *Proceedings of IEEE/ACM MobiHOC 2000*, pages 11–18, Boston, MA, U.S., Aug. 2000.

[29] C. E. Perkins. Mobile IP. *IEEE Communications Magazine*, 35(5):84–99, may 1997.

[30] C. E. Perkins and P. Bhagwat. Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers. In *Proceedings of the ACM SIGCOMM 1994 Conference*, pages 234–244, London, United Kingdom, 1994.

[31] C. E. Perkins and E. M. Royer. Ad hoc On-Demand Distance Vector Routing. In *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–100, New Orleans, LA, USA, Feb. 1999.

[32] H. Pucha, S. M. Das, and Y. C. Hu. Ekta: An Efficient DHT Substrate for Distributed Applications in Mobile Ad Hoc Networks. In *Proceedings of the 6th IEEE Workshop on Mobile Computing Systems and Applications (WMCSA 2004)*, English Lake District, UK, Dec. 2004.

[33] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker. GHT: A Geographic Hash Table for Data-Centric Storage. In *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications (WSNA '02)*, pages 78–87, Atlanta, GA, USA, 2002.

[34] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms (Middleware) 2001*, Heidelberg, Germany, Nov. 2001.

[35] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In *Proceedings of the SIGCOMM 2001 conference*, pages 149–160. ACM Press, 2001.

[36] P. F. Tsuchiya. The landmark hierarchy: a new hierarchy for routing in very large networks. In *Proceedings of the ACM SIGCOMM 1988 Conference*, pages 35–42, Stanford, CA, USA, 1988. ACM Press.

[37] A. C. Viana, M. D. de Amorim, and S. Fdida. An Underlay Strategy for Indirect Routing. *Wireless Networks*, 10:747–758, 2004.

[38] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998.

[39] F. Xue and P. R. Kumar. The Number of Neighbors Needed for Connectivity of Wireless Networks. *Wireless Networks*, pages 169–181, 2004.

[40] 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)*, Paris, France, June 2005.