REIHE INFORMATIK
14/2002

A Reactive Location Service for Mobile Ad Hoc Networks

Michael Käsemann[1] Holger Füßler[2] Hannes Hartenstein[1] Martin Mauve[2]

[1]: NEC Network Laboratories
NL-E Heidelberg
Adenauerplatz 6
D-69115 Heidelberg

[2]: Universität Mannheim
Praktische Informatik IV
L15, 16
D-68131 Mannheim

A Reactive Location Service for Mobile Ad Hoc Networks

Michael Käsemann * Holger Füßler † Hannes Hartenstein * Martin Mauve †

November 2002

Abstract

We present and analyze a *reactive location service* (RLS) for mobile ad hoc networks. RLS provides a mobile node in a wireless ad-hoc network with the means to inquire the current geographical position of another node on-demand and can be used as a building block for location-based routing. We provide a comparison of RLS to an ideal omniscient location service as well as to the complex Grid Location Service (GLS). In addition, we compare the performance of greedy location-based routing in combination with RLS to the performance of a non-location-based ad hoc routing approach, namely Dynamic Source Routing (DSR). DSR was chosen for the comparison since RLS can be considered an adaptation of DSR's route discovery mechanisms to the location-based domain. We also introduce and study possible optimizations for RLS, in particular caching, random re-broadcast jitter, and re-broadcast suppression. The quantitative results of our NS-2 simulation study show a very good performance of RLS combined with greedy routing, outperforming GLS and DSR for scenarios with high mobility and high node density.

1 Introduction

Location-based ad hoc routing protocols [11] represent a promising approach to enable multi-hop communications in highly mobile wireless ad-hoc networks of location-aware nodes [8]. One key application area for location-based routing lies in the field of inter-vehicle communications where location-awareness of the nodes will become standard within the next few years [6]. In location-based routing, forwarding decisions are based on the location of the forwarding node in relation to the locations of the source and destination nodes. A greedy forwarding strategy, for example, will give a packet to the neighbor that shows the largest progress towards the destination. In contrast to purely topological ad hoc routing approaches, i.e. approaches that do not make use of location information, no route set-up or route maintenance is needed with a location-based scheme since packets are forwarded 'on the fly'. However, the challenge of location-based ad hoc routing lies in learning the current geographic location of a desired communication partner. In this paper we put forward a *reactive location service* (RLS) that allows a mobile node in an ad hoc network to inquire the geographic position of another node in an on-demand fashion. We present NS-2 simulation results that show that RLS is a simple, yet efficient and effective location service for mobile ad hoc networks.

In previous work, research interest was either not focusing on the location service itself and therefore an omniscient location service was used for its simplicity, as in [8], or location services with a strong 'proactive' component, as in [5, 1], were proposed. We say that a location service has a strong proactive component when location information is distributed and managed even when there is no node in the network requesting this information at that point in time. In contrast, a reactive location service only retrieves location information 'on-demand'.[I] The objective of the work described in this paper was to devise and study a purely reactive approach. The reactive location service we propose and analyze represents an adaptation of the Dynamic Source Routing (DSR) [7] route discovery mechanism to the domain of location-based routing approaches. Since DSR's route discovery and, therefore, RLS' location request are based on flooding principles, the well-known 'broadcast-storm' problem [2] has to be addressed. We make use of strategies outlined in [4] to deal with this problem. In order to show

*Hannes Hartenstein and Michael Käsemann are with the NEC Network Laboratories Heidelberg
({Hannes.Hartenstein, Michael.Kaesemann }@ccrle.nec.de)
†Holger Füßler and Martin Mauve are with the "Lehrstuhl für Praktische Informatik IV" of the University of Mannheim
({fuessler, mauve }@informatik.uni-mannheim.de)

[I] As outlined in [11], classification of location services into 'proactive' and 'reactive' is too coarse-grained since most proposals combine the two aspects. However, for the work described in this paper the classification is sufficient.

RLS's efficiency and effectiveness, we compare it with an omniscient location service (OLS) as well as the Grid Location Service (GLS) [1, 9, 10] that represents an approach with a strong proactive component. In addition, we compare the combination of greedy location-based forwarding and RLS with DSR. An in-depth analysis based on NS-2 simulations that evaluate the impact of node mobility and density is presented.

The remainder of this work is organized as follows: In Section 2, the RLS algorithms are presented. Simulation set-up and results are explained in Section 3. Finally, Section 4 summarized our findings and the paper is concluded with an outlook on future work in Section 5.

2 Reactive Location Service

2.1 Baseline algorithm

When a node wants to communicate with a another node using position-based routing, it needs to add the geographical position of the destination node to the headers of all packets sent to this destination. The routing scheme then forwards these packets to the given position, which – in an ideal case – would be the present location of the destination node at the time of the packets arrival. In a real world situation the geographical position can never be 100% accurate and it is the task of a location service to provide the current location as precisely as possible. To do this the location service may use any algorithm that is able to provide an (id,location)-pair to an inquiring node.

In RLS the algorithm works as follows: A node querying the geographical position of a certain node issus a *location query* packet. The query packet contains the source node's location and id as well as the id of the destination node. It is flooded throughout the network until it reaches said destination node or its time-to-live (TTL) expires. If the destination is not reached, RLS assumes unreachability, which represents one of the following cases:

Network Partitioning If the network is partitioned, with source and destination in different partitions, the destination can never be reached by the query and it will eventually expire.

Inactive Node The destination node does not exist or has (temporarily) been deactivated. Thus it can never be reached by the query.

Large Distance Source and destination may be farther apart (in hops) than the maximum time-to-live would account for.

The above cases are indistinguishable for RLS since error detection is based on a timeout mechanism at the sending node. Note that network congestion can have the same impact on RLS as network partitioning.

To avoid infinite packet looping and duplication during flooding, nodes must be kept from forwarding queries they have already processed. Therefore the source node marks all location query packets it sends with a sequence number that increases with each attempt of the source node to aquire the destination node's location[II]. Each forwarding node then uses a sequence number cache to check if it is permitted to forward this packet by comparing the sequence number stored in the query packet to the one in its cache. If no cache entry exists or the stored sequence number is smaller, the node has not sent out this query before and updates its cache before rebroadcasting it. A cached sequence number bigger than the one contained in the packet indicates a duplicate or looped packet and the query is discarded.

When a destination node receives a query packet that carries its id, it creates a *location reply* packet that is marked with the querying node's id and location as destination information and carries the responding node's id and location as payload. This reply is sent back to the source by means of the underlying routing protocol (e.g. greedy unicast routing, flooding, etc.). Reception of a reply packet at the querying source completes the location discovery cycle. The destination's location is inserted into all packets buffered for this destination, and the packets are sent out.

If unreachability occurs or a reply is lost during its way back to the source node, a timeout for the data packet will occur at the source and another location request cycle will be initiated: The sequence number is increased by one and a new location query packet is sent. The justification for this automated retry scheme is the fact that all unreachability criteria mentioned above are subject to change in mobile scenarios. Furthermore, the loss of a reply packet is also indistinguishable from location service-based unreachability criteria. Thus repeating a location query increases the chances of a successful location discovery. On the other hand, there are upper limits on how long a packet may be delayed before it becomes obsolete and on how much additional load

[II] Therefore a sequence number is associated with a (source,destination)-pair.

the location service should put on the network by itself. Therefore, the retry mechanism has an upper bound at which retries cease and the data packet is discarded.

2.2 RLS Extensions & Improvements

2.2.1 Flooding Schemes

Flooding defines a simple distribution method for packets: Each node re-broadcasts all packets it receives. If we assume that the network is not partitioned, we can state that, if the TTL of the packet is at least as high as the diameter of the network and no link layer failures occur, all packets will reach every node in the network.

This basic flooding approach suffers from a number of problems, e.g., non-reliability of link layer broadcasts or the overload of the link layer by the 'broadcast storm problem' [2]. In order to help with the latter, a number of proposals were devised in the literature (see [4]). Other problems can be addressed by different modifications to the basic flooding algorithm.

For RLS we considered the following three options:

Linear Flooding This variant floods a small neighborhood region first, by limiting the packet TTL value, d_{max}, to a small number of hops. If a timeout occurs, d_{max} is increased by a constant d_{step} and the query is restarted. When the destination is not found before d_{max} reaches the allowed limit, the destination is labeled unreachable.

The biggest drawback that we encountered with linear flooding was that nodes were easily able to remain "hidden"[III] to a source node. The reason for this is that a node that moves away from the source can remain in front of the expanding query wave by moving fast enough to pass at least d_{step} hops in the timeout period needed by the source before it restarts the query with an increased hop limit.

Exponential Flooding To attenuate the phenomenon of "hidden" nodes, we tested exponential flooding. This method works like linear flooding, but instead of increasing d_{max} by an additive constant it is multiplied by a factor. This limits the chance of nodes staying "hidden", because nodes would have to move very fast to outrun the "query wave" after the first few retries.

Binary Flooding In many real-world scenarios communication is often local, for example at conferences or during courses on campus. It therefore makes sense to use a flooding scheme that distinguishes two types of communication: *near* and *far* communication. We called this approach, which was inspired by the route-requests of DSR, binary flooding. The source node first floods a close range neighborhood (e.g. one or two hops) to see if near traffic is intended. If no reply is received, the traffic is classified as far and d_{max} is set to the allowed limit – instead of increasing it gradually.

For the evaluation of RLS we chose binary flooding as the main flooding scheme, and only did measurements with exponential flooding for comparison. An in-detail study of exponential flooding is left for future work.

2.2.2 Caching

Every node forwards queries, replies, and data packets that all carry location information of their respective source nodes and maybe even of a destination node. By evaluating these passing-by packets and by storing this information in a location cache, a node can acquire valuable location information of other nodes for free (i.e. without any extra costs in form of packet overhead). If a node wants to initiate a communication, RLS then checks its location cache first and may be able to answer the location query of the data packet right away. This saves a location discovery cycle and reduces packet delay as well as network traffic.

In addition, it is also possible to evaluate any other type of passing packets for information that may be used to fill the local cache.

2.2.3 Cached Replies

A query does not necessarily have to be answered by the target node itself. A cached-reply strategy allows nodes to answer queries not destined for them, if they have the required location information available in their respective location caches. Since every node gathers location information for its own communications or even from packets that are passing by (as described in Section 2.2.2), replies might be generated by nodes much closer to the source than the intended destination. This reduces latency, but

[III] A hidden node describes a node that is part of the network, but whose position can not be acquired by a node that queries for it.

might provide the source with less accurate positions. Another problem of this approach is that it is likely to produce more than one reply and the source node has to be able to deal with this fact.

Cached replies may also be used to reduce network load, but would have to be used in conjunction with a flooding scheme like exponential flooding, as described in Section 2.2.1, to do so. If the whole network is flooded no relief in network load can be achieved due to the fact that the destination node is likely to be reached anyway and a cached reply does not prevent the location query from being forwarded by other nodes.

2.2.4 Radial Flooding

In all standard flooding schemes each node within radio range that receives a location query packet rebroadcasts it as soon as it can acquire access to the wireless medium. This may lead to collisions that can be reduced by introducing a random backoff at each node. We extended this approach by adding a "radial" component, that is supposed to increase the expansion speed of the query flooding while still providing the congestion alleviation of a random backoff. We achieve this by having each node that receives a query compute its distance to the last hop node and calculate a backoff time with respect to this distance by

$$t_{backoff} = t_{max} \cdot \left(1 - \left(\frac{d_{last}}{d_{rrange}}\right)^2\right) \qquad (1)$$

where t_{max} is the maximum delay a packet may be 'backed off' , d_{last} is the distance to the last hop node and d_{rrange} equals the nominal radio range. Note that $d_{last} \leq d_{rrange}$.

This assures that the farther away a node is the sooner it will rebroadcast the query. When we assume a two-dimensional uniform distribution of nodes in a circle with a radius equal to the radio range, the distribution function of distances to the center of this circle is quadratic. This means that there are potentially more nodes with high distances than nodes with low distances.

To keep these nodes from trying to transmit at the same time, the resulting timer distribution should distribute uniformly over a certain time interval. The timer function shown in equation (1) has this property (as shown in Appendix A). This means that the intervals from which the timer values are taken increase with the distance from the last hop node, since the number of nodes in a uniformly distributed scenario also increases with the distance from the source node.

2.2.5 Rebroadcast Suppression

When each node rebroadcasts queries, there will be 'redundan' packets, i.e., packets that reach little or none additional nodes[IV] but congest the network. This phenomenon is called the *broadcast storm problem* and was discussed in [2] where a way to alleviate this problem through packet suppression has also been presented. Different suppression mechanisms were shown that keep nodes, which are unlikely to reach additional nodes, from rebroadcasting packets.

Based on the evaluations of these suppression mechanisms done in [4] we chose to implement a combined distance-/counter-based scheme for RLS to study the effects suppression has on the performance of a flooding-based location service. The results are described in Section 3.

The rebroadcast suppression itself is implemented as follows: The first time a node receives a location query packet it checks its distance to the packet's last hop node. This distance is then evaluated against a threshold value, d_{thres}, to decide whether enough additional area coverage is expected to be reached to justify a rebroadcast. If so the packet is scheduled for retransmission after a so called random assessment delay (RAD) that needs to be long enough to receive packets from all nodes within the radio range. If the expected additional area coverage is low, so is the chance for reaching additional nodes and the packet is discarded. During the RAD the node may hear the packet again from neighbors that have rebroadcasted it. Every time this happens, the node calculates its distance to the source of the rebroadcast and compares this distance to the one calculated the last time. The smaller distance of the two is then checked against the threshold and again a decision, whether or not the packet should be discarded, is made. However, this time the decision to rebroadcast only resumes the wait for the end of the RAD. This method ensures that a node always associates itself with the closest (re-)broadcasting neighbor and calculates the smallest expected additional coverage (EAC) it can achieve. Should the smallest distance ever fall below the threshold the packet gets discarded.

In case the node hears the packet more than c_{max} times it discards it regardless of any distances, because it is unlikely any additional nodes will be reached by its rebroadcast, that have not yet been reached by one of the previous rebroadcasts or will

[IV] An additional node is a node that has never had the packet in question before.

be reached by one of the many neighbors. c_{max} is chosen according to the results in [2].

2.2.6 Passing-Packet Updates

Another optimization that may be implemented in RLS is the passing-packet update service, which checks every packet received by a mobile node for positional information and updates it, should a query of the location cache produce more accurate (i.e. newer) information. The justification for this is that the closer a packet gets to the destination node the more likely it is for forwarding nodes to have more precise location information in their caches due to beacon and other localized traffic and to the time that has already passed since the packet was marked and sent.

3 Evaluation

3.1 Simulation Setup

To evaluate RLS we compared it to two other location services, called GLS and OLS. All location services used GPSR [8] as the underlying greedy position-based routing strategy. We also compared the GPSR/RLS combination to DSR [7], which we chose because it uses a very similar flooding technique for the route discovery and thus highlights the differences between position-based and topological routing in ad-hoc networks. DSR also has the advantage that it is often mentioned in other papers dealing with mobile ad-hoc networks and thus our results become comparable.

GLS is our own implementation of the Grid Location Service, as it was introduced in [1]. Since the original authors used a greedy routing scheme based on two-hop neighborhood and grid-based forwarding, specific to this location service, while we pair it with the standard GPSR scheme, it is very likely that our scheme does not represent the maximum performance of GLS. However, if we had implemented all optimizations we would not have been able to extract as much information from the comparison of the one-for-one[V] scheme used in RLS with the some-for-some[VI] approach of GLS as we could by keeping the differences between the two to a minimum.

[V] Each node is the only participant in the network that may answer queries for his location.

[VI] Some nodes are selected as location servers for some other nodes and a location query may be answered by any such location server that holds the required information.

OLS stands for Omniscient Location Service that is based on the assumption that each node can acquire positional information of any other node in the network without delay or inaccuracy. Although this can never be achieved in a real-world implementation, it is a legit assumption in a simulation environment and provides an impression of GPSR behavior and greedy connectivity. OLS behaves like the location database used in [8].

simulator	ns-2.1b8a
area size	2000 m × 2000 m
number of nodes	100 – 400
mobility model	random waypoint
node speed	10, 30 and 50 m/s

Table 1: Simulation Setup

The basic simulation setup can be found in Table 1.

The traffic patterns selected 20 node pairs during a time window of 25 seconds (starting at second 15 and ending at second 40 of the simulation) and send 200 packets per connection from the source node to the destination node with a rate of 4 packets per second. A complete simulation run was 120 seconds in length. Thus all nodes had at least 50 seconds to send all of their packets and 30 seconds to finish routing any packets that were not yet delivered or dropped. For traffic we used ping packets that were 128 bytes in size at the agent level, i.e., without routing header overhead. To each ping packet that arrived at the destination an echo packet of the same size would be sent back to source[VII]. GPSR as well as location service headers contained all fields necessary for a real-world implementation with appropriate field sizes (e.g. 4 byte for the id or 3 byte per location coordinate). The chosen field sizes match the default sizes of DSR as much as possible to guarantee comparability.

Even though all code was taken from and written for the ns-2 version 2.1b8a, we replaced the MAC 802-11 implementation with a bug-fixed version from the ns-2 distribution 2.1b9 in which we, ourselves, fixed one more bug. The MAC had a data rate set bandwidth of 2MBit/s and a basic rate set bandwidth of 1MBit/s.

For the setup of GPSR, please refer to Table 2.

The GLS parameters were chosen to correspond to those used in [1] as much as possible. We even set the grid size to 250m, even though we do not use two-hop neighbor tables.

[VII] However, no echo packet is sent if a ping packet arrives a second time; which might happen if packet duplication occurs.

implementation	ported from Brad Karp ([8])
perimeter mode	off
beacon piggybacking	on
beacon interval	2 secs

Table 2: GPSR parameters

In RLS caching and radial flooding were always enabled. Cached replies were disabled, because we considered it unnecessary for the given traffic patterns in conjunction with the use of binary flooding. The latter, as described in Section 2.2.1, was used for most simulations, but some results for exponential flooding are also provided and will be explained in Section 3.2. We also did simulation runs with enabled broadcast suppression because even though the broadcast storm problem is not acute for the given traffic patterns we wanted to know how much of an impact suppression would have on delivery and latency.

Other parameters we used in the RLS configuration can be found in Table 3.

Time-to-live:	
Max.Packet TTL	64 hops
Max.Query Distance (d_{max})	32 hops
Timeouts:	
Query Cycle Timeout	5 secs
Location Cache Timeout	5 secs
Sequence Number Cache Timeout	10 secs
Broadcast Suppression:	
Random Assessment Delay (RAD)	0.02 secs
Max.Receive Count (c_{max})	4
Distance Threshold (d_{thres})	45 m

Table 3: RLS Parameter List.

We measured delivery ratio, packet overhead, average single hop latency and average route lengths. Delivery ratio is given as the percentage of ping packets that were successfully delivered to the destination. Since echoes are only sent if a ping packet precedes it and uses the route build by the ping packet, we did not include them in the evaluation process. However, one should keep in mind that echo packets transport information back to the source node and thus improve the performance of location based protocols. The packet overhead denotes the average amount of kilobytes of routing protocol packets that were sent or forwarded throughout a simulation on the network layer level. This means it represents the mere routing protocol overhead and not the kilobytes consumed by the ping and echo packets.

What we call the average single hop latency represents the total delay of a delivered ping packet from the source to the destination, normalized by the number of hops it has taken. Thus it includes the delay of the route set-up phase, which is an order of magnitude higher than normal network layer hop-to-hop delays. The reason for this definition is that it enables us to measure the quality of the route set-up and the importance of keeping intact routes in mobile scenarios, because any route break results in additional set-up phases that increase this value. Since the average single hop latency heavily depends on the amount of delivered packets as well as the number of taken hops, we also measured the average route lengths and will use them as well as the delivery ratio to interpret the single hop latency graphs in Section 3.2.

3.2 Results

Figure 1 shows the average delivery ratios of all evaluated routing schemes (i.e. all combinations of GPSR with a location service, as well as DSR).

As can be seen in Figure 1, DSR has an advantage of approximately 20% in the 100 nodes scenario with 10 m/s movement speed while its performance rapidly decreases for denser networks and higher movement speeds. In fact the 100 nodes 10 m/s case is the only one where DSR is not outperformed by GPSR/RLS. In scenarios with a lower mobility, routes found by DSR tend to be stable for quite some time, if not even for a whole connection. As movement speed rises routes break often, because the topological neighbors move away. This forces DSR to discover new routes, thereby increasing the load on the network. Since the main reason for low delivery rates is a congested network (as can be seen by correlating Figure 1 to Figure 2), high load beats down DSR's performance in high movement scenarios to delivery ratios of only 14-50%. But DSR also has a problem to scale well with rising node densities, which, again, is due to network congestion. This time the congestion is generated by the route request packets, because they increase in size while being flooded in the network and thus can get very large. Since all nodes participate in the flooding process, this means that many nodes in a close range try to forward these large route requests, when the node density rises, thus producing high stress for the wireless medium very fast, which may rapidly exceed its capacity.

RLS combined with greedy position-based routing does better for both of these problems that limit the scalability and achieves delivery ratios of 90+%

in all cases except for 100 nodes scenarios. Position-based routing schemes – in comparison with the source-routing approach – profit from high movement speeds because changing topology not only breaks routes on a regular basis, but also creates new ones. Since this type of routing scheme also does not care which neighbor forwards a packet as long as it progresses in the right direction, new routes are used more flexible than in DSR, thus saving discovery overhead. Dense networks do not pose a problem either, because position-based routing selects one neighbor to forward a packet, regardless to how many neighbors a node has. Finally RLS, while using a flooding scheme, only floods small constant sized packets that do not stress the wireless medium as much as DSR's route requests do.

However, the reason, why GPSR/RLS does not beat DSR in the 100 nodes scenario with movement speeds of maximal 10 m/s and only achieves 60-75% delivery ratio in 100 nodes scenarios with higher movement speeds, is also the greedy position-based routing scheme of GPSR. To successfully route a packet, GPSR needs greedy connectivity. Greedy connectivity denotes the subset of total connectivity[VIII] that represents all nodes reachable from a given source node by use of a greedy heuristic. Since the greedy heuristic might get stuck in local maxima, it can fail to state connectivity even if, in fact, it exists. In scarce scenarios, like 100 nodes in a 4 km^2 square, greedy connectivity is relatively small and many holes[IX] exist that keep a simple greedy routing mechanism from successful delivery. Figure 1(d) demonstrates this by showing the delivery ratio that a greedy forwarding strategy can achieve if it has perfect location information at its disposal. This also means that a recovery scheme, like the perimeter mode for GPSR, presented in [8], would improve the rates for GPSR/RLS.

When compared to GPSR/GLS we see that GLS behaves similar to RLS (or OLS, which can be seen as a reference), with the exception of a slight decrease for 100 nodes at 10 m/s and a decline in performance for dense high speed scenarios. The first is mostly due to greedy connectivity; only in GLS the effect is magnified because location queries are also sent as unicast packets by means of greedy forwarding. The reason for its weakness in dense high speed scenarios is the fact that in GLS each node needs to send so called position update packets to all nodes that can be queried for its position (its location servers). If nodes move at high speed the frequency of these updates rises and in dense scenarios many nodes try to send them at the same time, thus congesting the network. For a closer look at GLS the reader is referred to [10].

We also included a comparison of RLS with binary flooding to RLS with exponential flooding (Figures 1(a) and 1(e)), because it shows that trying to reduce the overall network load by keeping query floods in as small a subset of the network as possible, can be quite harmful if the communication patterns have no preference for localized traffic. The multiplication in query sends creates an up to 20 times higher load that soon exceeds the capacity of the wireless medium (as can be seen in comparison of Figures 2(a) and 2(e)). It remains as future work to evaluate how RLS with exponential flooding behaves if used in extremely large scenarios[X] with localized traffic patterns. The impact of broadcast suppression also remains to be studied.

Figure 2(d) depicts the amount of traffic generated by GPSR beaconing, which is not related to the speed with which nodes move but only to the number of nodes that periodically generate them.

All other graphs of Figure 2 show the same basic behavior, i.e., an increase in packet overhead for denser and very mobile scenarios, on different scales. Correlated to the delivery ratios in Figure 1 we see that a routing scheme only achieves good ratios as long as its protocol overhead does not congest the network. Achieving scalability is therefore a matter of keeping the overhead low in the targeted application area. All GPSR based routing schemes have an overhead of 90-450 kB per simulation run in the scenarios we chose as application area, which is well below the network capacity. Thus the location service has to be the focus of the design attention. RLS with binary flooding never exceeds 1600 kB per simulation run and outperforms GLS as well as DSR. But RLS with exponential flooding shows that RLS is only as effective as its flooding strategy. GLS, though unoptimized, does not seem to be suited for fast moving scenarios and DSR seems best suited for small and slow scenarios.

The graphs depicted in Figure 3 show the average single hop latency, i.e., the average amount of time a ping packet needs to get from one node to the next on its route to the destination. Only pings that arrive at the destination were used for these graphs and thus the values may misrepresent the average

[VIII] Total connectivity is defined by the output of shortest-path algorithms such as Dijkstra or Floyd-Warshall.

[IX] A hole is an area of at least the radio range in diameter, that is without any nodes and thus two bordering nodes at opposite sides of this hole are unable to communicate.

[X] Extremely large scenarios emphasize the definition of localization, because more disjoint small areas may exist in which nodes have to communicate in a multi-hop fashion.

for (node number, movement speed)-pairs in those cases that the delivery ratio is quite low. For example DSR delivers only 14% of the ping packets in the 400 nodes scenario when nodes move at 50 m/s. This influences the single hop latency, because those pings that are delivered also tend to be very close to the source (according to Figure 4(b) delivered ping packets have only taken about 3 hops for the DSR case we just mentioned).

RLS achieves single hop latencies of 0.01-0.5 seconds, with the best (i.e. smallest) delays in networks that are moderately populated and moving at moderate speeds. These delays are only two to three times higher than those of pure greedy routing, which is due to the fact that the small query packets need at least one shortest roundtrip time to find the destination and get the information back to the source. However, Figure 3(e) shows that delays can only be kept small as long as the network is not congested, in which case single hop latency gets unacceptably high. The same is true for GLS and DSR.

One thing that is noteworthy and can be observed in all greedy routing based location services, except for OLS, is the fact that single hop latency is lowest for scenarios with 200 nodes instead of scenarios with 100 nodes. For DSR this effect is less noticable. While analyzing this phenomenon we observed that it is based on lack of connectivity. All location services, except for OLS, which delivers location information whether the queried node is reachable or not, need to reach the destination or a location server with a query and use timeouts to retry, if no answer is received. In scarce scenarios nodes may temporarily not be reachable because of network partitioning. This introduces a delay that can not be avoided by any routing scheme or location service. In scarce scenarios this temporary partitioning occurs more frequently and lasts longer, thus producing higher delays for packets. Since DSR makes use of total connectivity it suffers less from this phenomenon than GPSR, which considers the network partitioned more often due to usage of greedy connectivity.

To illustrate this we provided the single hop latency spectra for the 100 nodes scenario in Figure 5 for RLS and DSR. It is easy to see that while the majority of packets has very low latencies some packets have latencies that are many orders of magnitude higher and influence the average. We also see that greedy worst case latency is roughly twice the size of the worst case latency for DSR. We should also note that the default packet retention time of 30 seconds in DSR, which we adapted in GPSR for comparability, is quite unrealistic for real world scenarios where packets this old would probably be dropped. With 200 nodes in a 4 km^2 square, density has improved enough for greedy connectivity to be very close to total connectivity which causes the drop in average single hop latency. The following increase thereof is then due to network load and represents the delay acquired in the MAC 802-11 layer during the network traversal.

To better understand the single hop latency one should also take a look at Figure 4, which depicts the average route length in hops that ping packets have taken to the destination. Three things are worth mentioning about this:

1. RLS has a nearly constant route length of 4-5 hops and thus stays close to the greedy route length determined by the OLS.

2. Route length decreases with the delivery ratio for all schemes, because the amount of reached, far away destinations decreases, since longer routes are harder to maintain.

3. Conversely we see that DSR has route lengths of 5-7 hops in scarce, slow-moving scenarios, which represent connectivity that is not greedy connectivity (e.g. complicated routes) and can not be found by the simple greedy routing we use.

Finally we evaluated RLS with activated broadcast suppression and found that the values were nearly identical to those without suppression and the deviations could not be told apart from the variations of multiple simulation runs. This leads us to the conclusion that broadcast suppression, while not needed in connection patterns that do not lead to the broadcast storm problem (like the ones we used), does not show any negative effect in scenarios with light traffic. However, its use in high load scenarios remains to be tested. For detailed results on broadcast suppression, see Figure 6.

4 Conclusions

In this report, we proposed a flooding-based location service similar to DSR's route request method called *Reactive Location Service*. In addition to that, we presented extensive simulation results comparing Location Based Routing with RLS to DSR which is a topology based routing scheme and to Location Based Routing using a different, more scalable Location Service called GLS. We basically show the simulation results of these methods under different node density / mobility conditions.

As a result we show that Location Based Routing performs better for high density / high mobility conditions than DSR. Hence, we confirm the work in [8], but without the omniscient location knowledge the simulator can provide. In addition we showed that available bandwidth puts serious limitations on "more scalable" location services like GLS (also under the assumption of high node density / mobility).

5 Future Work

Future work includes a detailed study of RLS' behavior in high load scenarios as well as extremely large scenarios. It is especially interesting to see how optimizations like rebroadcast suppression and cached replies could be used and combined to increase the application area of RLS. For scenarios in which the broadcast storm problem occurs it would also be interesting to test the impact of different suppression mechanisms. Furthermore, a comparison of binary flooding to exponential flooding could reveal whether binary flooding is always better or if scenarios exist in which exponential flooding (maybe combined with cached replies and/or rebroadcast suppression) proves to be more suited.

A Distribution of Backoff Timer

Let X be a two-dimensional vector, that denotes a point in a circle with radius d_{rrange}, chosen randomly from the set of all points uniformly distributed in the area enclosed by said circle. Furthermore, let $D = d_{last}/d_{rrange}$ be the normalized, euclidean distance of X to the center of the circle with radius d_{rrange}. Then

$$F_D(d) = P(D \leq d) = \begin{cases} 0 & d < 0 \\ d^2 & d \in [0;1] \\ 1 & d > 1 \end{cases} \qquad (2)$$

is the quadratic distribution function of D.

By applying the conversion function $g(D)$ with

$$g(d) = d^2 \qquad d \in [0;1] \qquad (3)$$

we can transform D into a new random variable U that, according to the prove in [3], is uniformly distributed over $[0;1]$.

Thus, the backoff timer function

$$t_{backoff} = T_{max} \cdot \left(1 - \left(\frac{d_{last}}{d_{rrange}}\right)^2\right) \qquad (4)$$

is uniformly distributed over $[0;T_{max}]$ with the smallest backoff times for the largest values of d_{last}. This means that the nodes farthest away from the center of the circle have the smallest backoff period.

B Evaluation Statistics

The graphs, reflecting the evaluation results, can be found on the appropriate pages listed in the table below.

List of Figures

References

[1] J. Li , J. Jannotti, D. S. J. De Couto , D. R. Karger , and R. Morris . A scalable location service for geographic ad hoc routing. In *Proceedings of the 6th ACM International Conference on Mobile Computing and Networking (MobiCom '00)*, pages 120 – 130. ACM, August 2000.

[2] S.-Y. Ni , T. Yu-Chee , C. Yuh-Shyan , and S. Jang-Ping . The broadcast storm problem in a mobile ad hoc network. In *Proceedings of ACM MOBICOM´99*, pages 151 – 162, Seattle Washington U.S.A., 1999.

[3] A. Papoulis . *Probability, Random Variables, and Stochastic Processes*. WCB/McGraw-Hill, 3rd edition, 1991.

[4] B. Williams and T. Camp . Comparison of broadcasting techniques for mobile ad hoc networks. In *Proceedings of ACM MOBIHOC'02*, pages 194 – 205, EPFL, Lausanne, Switzerland, June 2002.

[5] S. Basagni, I. Chlamtac, V. R. Syrotiuk, and B. A. Woodward. A distance routing effect algorithm for mobility (dream). In *ACM MOBICOM '98*, pages 76 – 84. ACM, 1998.

[6] H. Füßler, M. Mauve, H. Hartenstein, M. Käsemann, and D. Vollmer. Location-based routing for vehicular ad-hoc networks. In *Proc. ACM Mobicom'02, student poster*, Atlanta, GA, September 2002.

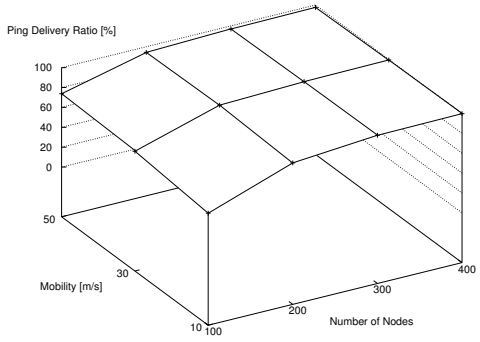
[7] D. B. Johnson and D. A. Maltz. Dynamic source routing in ad hoc wireless networks. In T. Imielinske and H. Korth, editors, *Mobile Computing*, volume 353. Kluwer Academic Publishers, 1996.

[8] B. Karp and H. T. Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In *Proceedings of the 6th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom 2000)*, pages 243–254, Boston, MA, U.S.A., August 2000.

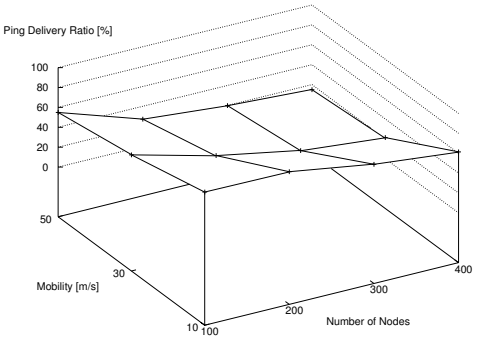
[9] M. Käsemann, H. Hartenstein, H. Füßler, and M. Mauve. Analysis of a location service for position-based routing in mobile ad hoc networks. In *Proc. of the 1st German Workshop on Mobile Ad-hoc Networking (WMAN 2002)*, GI – Lecture Notes in Informatics, pages 121 – 133. GI, March 2002.

[10] M. Käsemann, H. Hartenstein, H. Füßler, and M. Mauve. A simulation study of a location service for position-based routing in mobile ad hoc networks. Technical Report TR-7-2002, Department of Science, University of Mannheim, 2002.

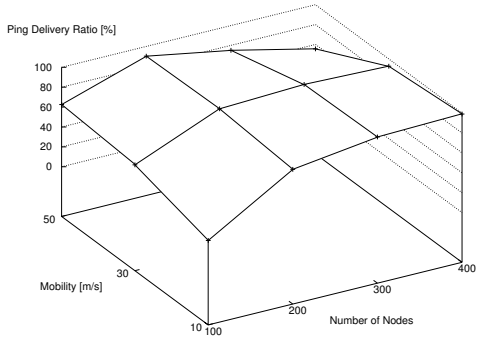
[11] M. Mauve, J. Widmer, and H. Hartenstein. A Survey on Position-Based Routing in Mobile Ad-Hoc Networks. *IEEE Network*, 15(6):30 – 39, November/December 2001.



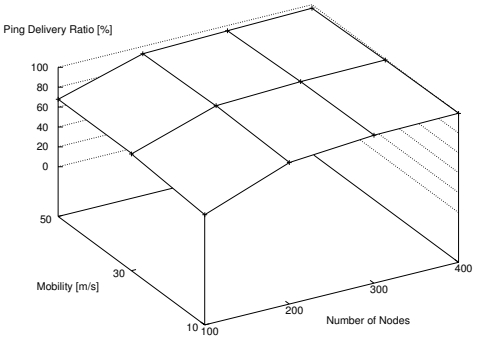
(a) RLS



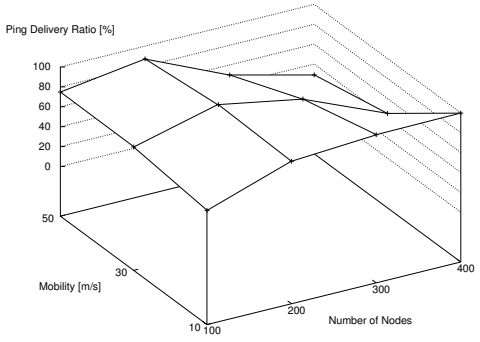
(b) DSR



(c) GLS

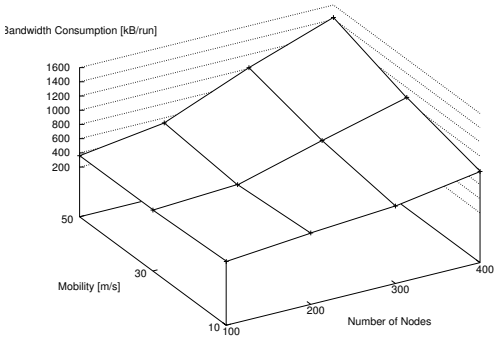


(d) OLS

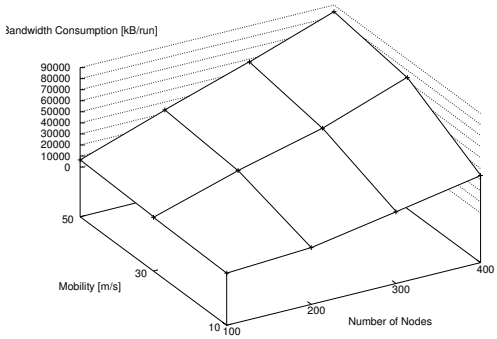


(e) RLS (Exponential Flooding)

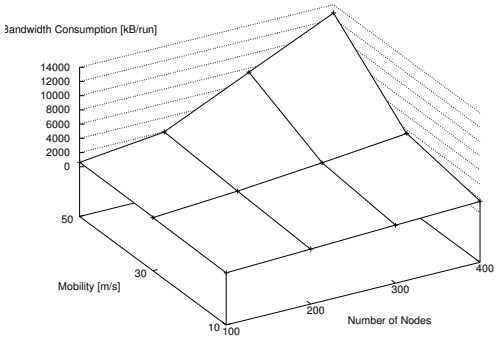
Figure 1: Ping Delivery Ratios for DSR and GPSR with different Location Services.



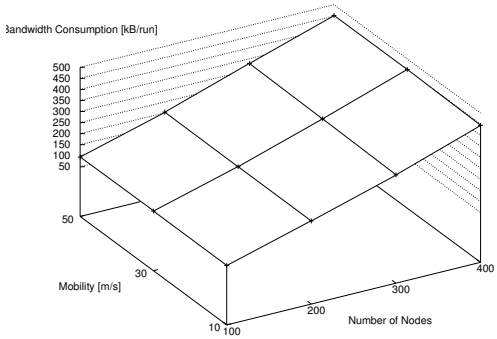
(a) RLS



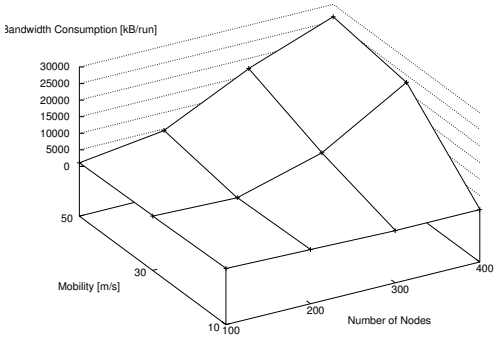
(b) DSR



(c) GLS

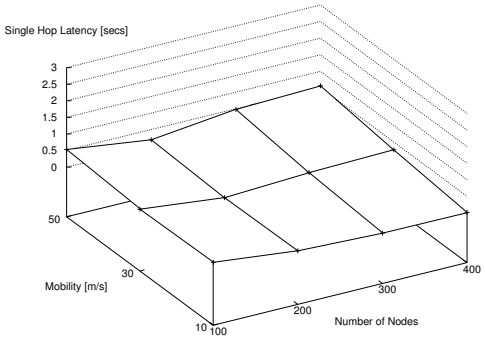


(d) OLS

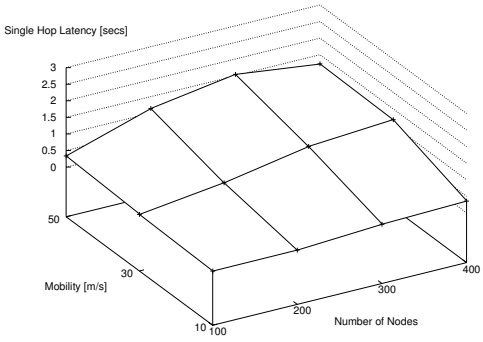


(e) RLS (Exponential Flooding)

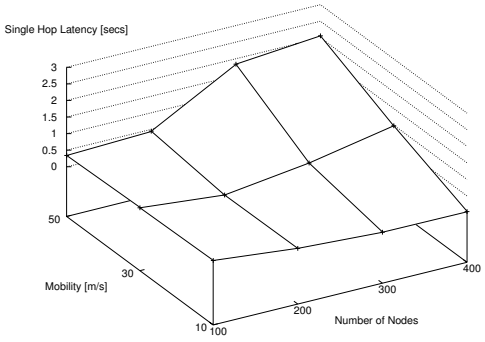
Figure 2: Packet Overhead for DSR and GPSR with different Location Services.



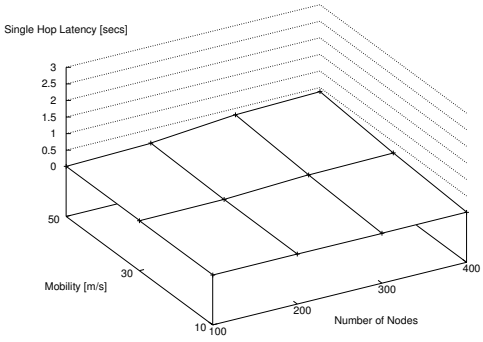
(a) RLS



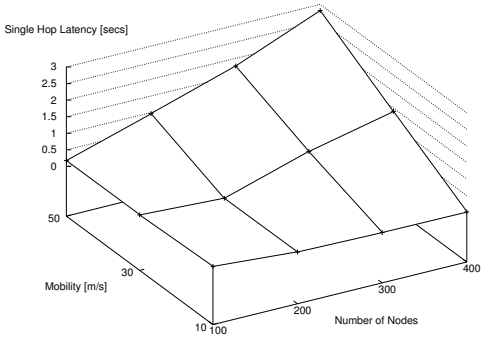
(b) DSR



(c) GLS

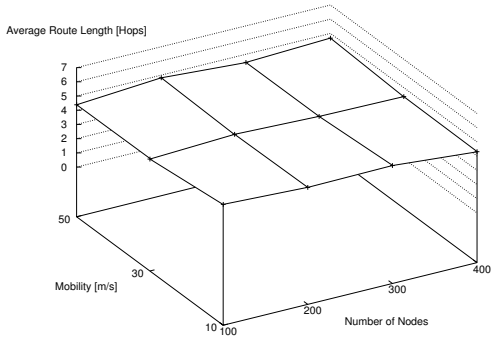


(d) OLS

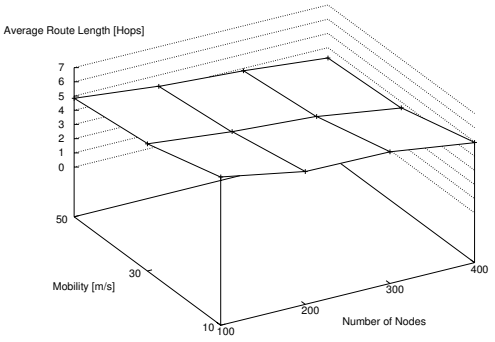


(e) RLS (Exponential Flooding)

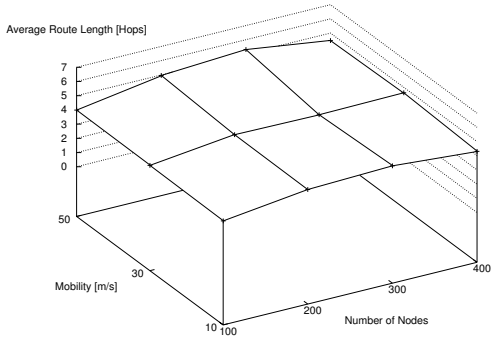
Figure 3: Single Hop Latency for DSR and GPSR with different Location Services.



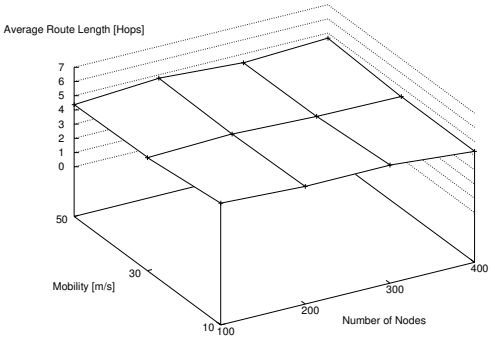
(a) RLS



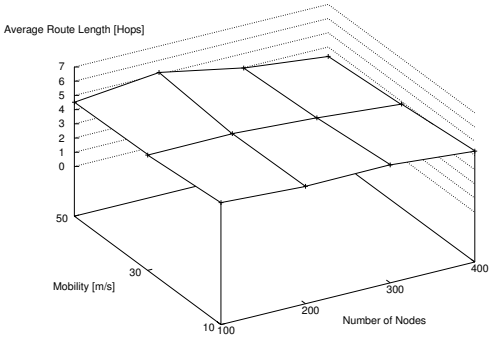
(b) DSR



(c) GLS

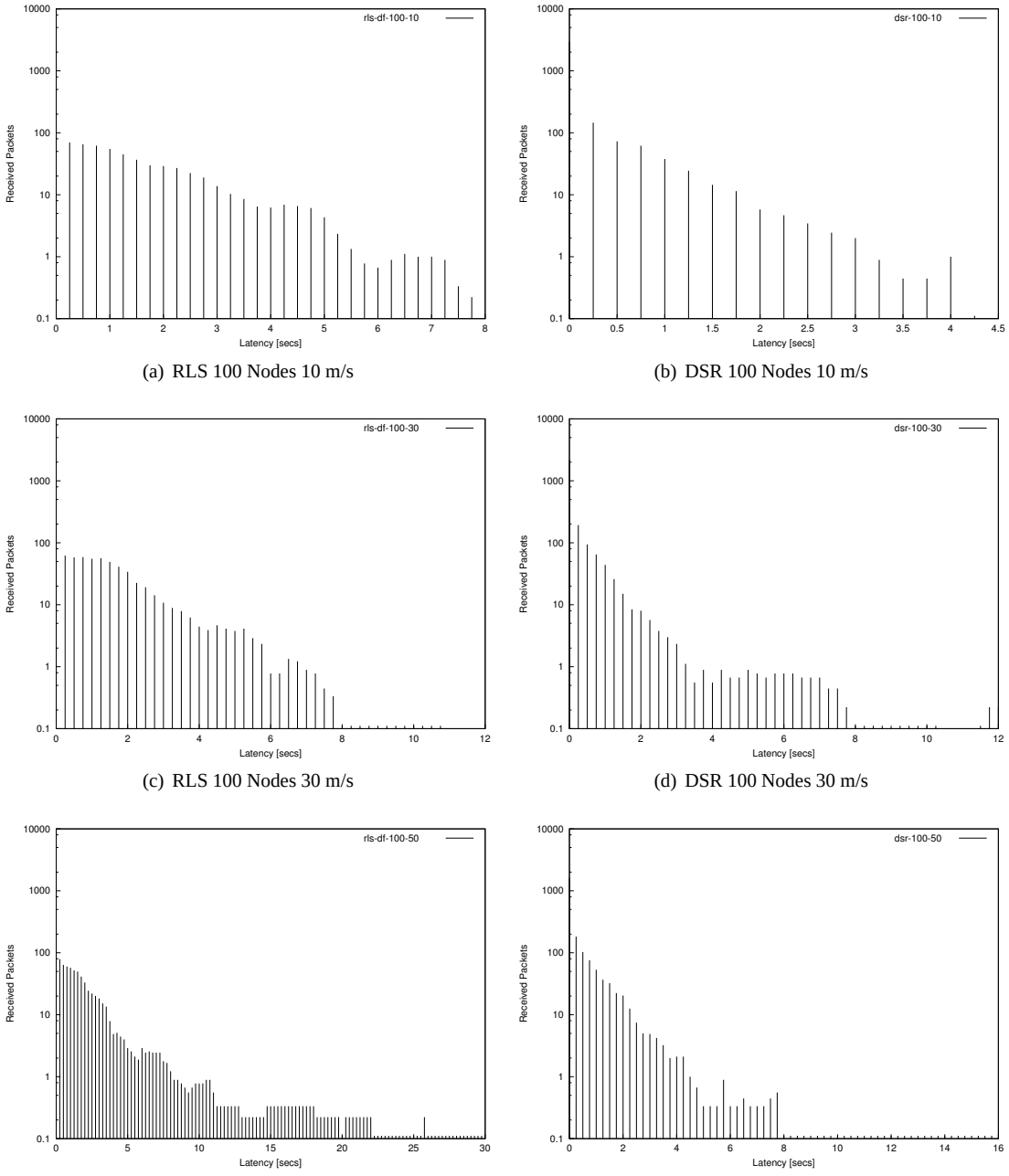


(d) OLS



(e) RLS (Exponential Flooding)

Figure 4: Average Route Length for DSR and GPSR with different Location Services.



(a) RLS 100 Nodes 10 m/s

(b) DSR 100 Nodes 10 m/s

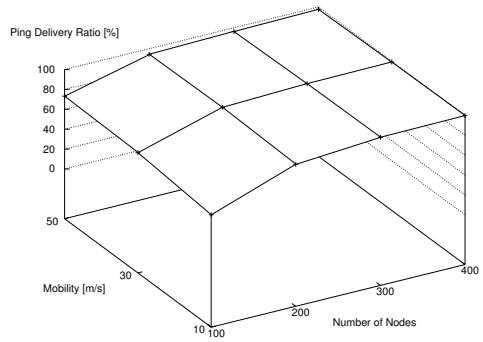
(c) RLS 100 Nodes 30 m/s

(d) DSR 100 Nodes 30 m/s

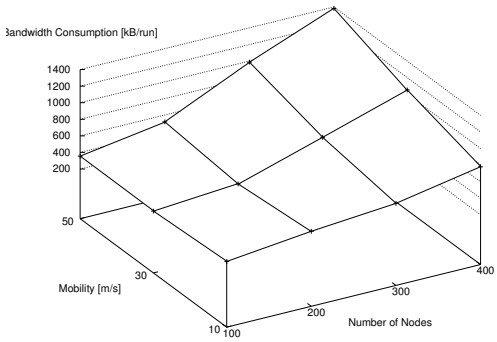
(e) RLS 100 Nodes 50 m/s

(f) DSR 100 Nodes 50 m/s

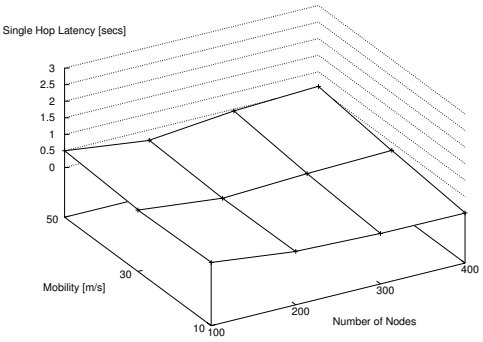
Figure 5: Single Hop Spectra for DSR and GPSR/RLS in a 100 nodes 10 m/s scenario (9 run average).



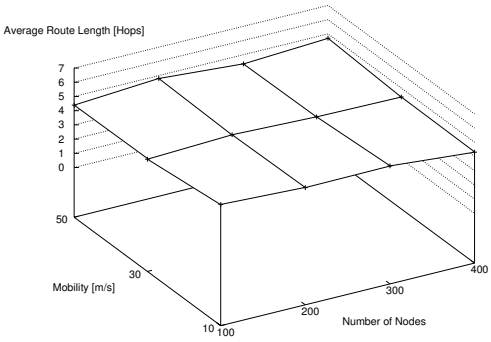
(a) Delivery



(b) Packet Overhead



(c) Single Hop Latency



(d) Average Route Length

Figure 6: Evaluation Graphs for GPSR/RLS with Rebroadcast Suppression.