



Query Localization Techniques for On-Demand Routing Protocols in Ad Hoc Networks

ROBERT CASTAÑEDA

Division of Computer Science, The University of Texas at San Antonio, San Antonio, TX 78249-0667, USA

SAMIR R. DAS and MAHESH K. MARINA

Department of Electrical & Computer Engineering and Computer Science, University of Cincinnati, Cincinnati, OH 45221-0030, USA

Abstract. Mobile ad hoc networks are characterized by multi-hop wireless links, absence of any cellular infrastructure, and frequent host mobility. Design of efficient routing protocols in such networks is a challenging issue. A class of routing protocols called *on-demand* protocols has recently found attention because of their low routing overhead. We propose a technique that can reduce the routing overhead even further. The on-demand protocols depend on query floods to discover routes whenever a new route is needed. Our technique utilizes prior routing histories to localize the query flood to a limited region of the network. Simulation results demonstrate excellent reduction of routing overheads with this mechanism. This also contributes to a reduced level of network congestion and better end-to-end delay performance of data packets.

Keywords: ad hoc networks, routing protocols, on-demand routing, flooding

1. Introduction

A mobile, *ad hoc* network is an autonomous system of mobile hosts connected by wireless links. There is no static infrastructure such as a base station. If two hosts are not within radio range, all message communication between them must pass through one or more intermediate hosts that double as routers. The hosts are free to move around randomly, thus changing the network topology dynamically. Thus, routing protocols must be adaptive and able to maintain routes in spite of the changing network connectivity. Such networks are very useful in military and other tactical applications such as emergency rescue or exploration missions, where cellular infrastructure is unavailable or unreliable. Commercial applications are also likely where there is a need for ubiquitous communication services without the presence or use of a fixed infrastructure. Examples include home-area wireless networking [17], on-the-fly conferencing applications, networking intelligent devices or sensors, communication between mobile robots, etc.

Design of efficient routing protocols is the central challenge in such dynamic wireless networks. Much work has been done in this area starting from the seventies, when the US Defense Research Agency, DARPA supported the PRNET (Packet radio Network) [13] and SURAN (Survivable Adaptive Networks) [21] projects. They supported automatic route set up and maintenance in a packet radio network with moderate mobility. Interest in such networks has recently grown due to the common availability of wireless communication devices that can connect laptops and palmtops and operate in license free radio frequency bands (such as the Industrial-Scientific-Military or ISM band in the US). In an interest to run internetworking protocols on ad hoc networks, a work-

ing group for Mobile, Ad hoc Networking (MANET) [16] has been formed within the Internet Engineering Task Force (IETF), whose charter includes developing a routing framework for running IP-based protocols in ad hoc networks. Several new routing protocols have been proposed in connection with the MANET working group efforts [16]. Of particular interest is the new class of *on-demand, source-initiated* protocols, that set up and maintain routes from a source to a destination on an “as needed” basis. This approach is in sharp contrast with the traditional *shortest path*-based protocols (e.g., link-state and distance vector [14]) that have been used successfully in dynamic, wireline networks, including the Internet.

1.1. On-demand protocols and flooding

The motivation behind the on-demand protocols¹ is that the “routing overhead” (typically measured in terms of the number of routing packets transmitted, as opposed to data packets) is typically lower than the shortest path protocols as only the actively used routes are maintained. However, as some recent performance evaluation work has shown [7], the routing overhead still approaches that of the shortest path protocols, if a moderate to large number of routes needs to be actively maintained (when, for example, there is a moderate to large number of active peer-to-peer conversations). This is because the on-demand protocols discover routes via a *flooding* technique, where the source (or any node seeking the route) floods the entire network with a query packet in search of a

¹ In this paper, we use the term “on-demand routing” synonymously with protocols that search routes via flooding. Note that on-demand routing is a general paradigm which is used in other contexts as well (e.g., QoS routing).

route to the destination. After receiving the query, each non-destination node propagates it to its neighbors via a wireless broadcast. Each query carries a unique identifier which helps prevent multiple propagation of the same query by the same node. This technique guarantees that the query reaches all nodes in the same connected component. Thus, if the destination is reachable from the source, the query will eventually reach the destination.

The sequence of hops traversed by the first query message received by the destination defines the route to be used for sending data packets. On receiving the first query, the destination replies to the source by sending a route reply message. If the wireless links are bidirectional, the reply goes back to the source just by retracing the route in the opposite direction.² The actual mechanics of doing this varies from protocol to protocol. For example, the DSR (*Dynamic Source Routing*) protocol [12], builds the route incrementally as a sequence of nodes visited by the query, and stores it in the header of the query packet. The reply packet carries this route and simply traverses it backwards. The AODV (*Ad Hoc On-Demand Distance Vector*) routing protocol [20], on the other hand, maintains the route in a distributed fashion using routing tables in the nodes on the route.

Flooding is straightforward to implement. However, as mentioned before, network-wide flooding incurs a considerable overhead and diminishes the performance advantage of on-demand protocols. Although several optimizations of the basic flooding mechanism have been proposed previously (e.g., use of a *time-to-live* or TTL field to limit the query within a specific number of hops from the source [3,4]), the flooding scheme can still deliver the query to a very large number of nodes in the network, leading to a high routing overhead. The problem can be severe when the mobility is high (very frequent route discoveries) and/or the network is large (many routing messages generated in regions far away from the source and the destination). Our goal in this work is to investigate new approaches to reduce the routing overhead by localizing the query flood to a limited region in the network. Similar ideas were explored before. The most prominent among them is the location-aided routing or LAR technique [15] which uses the Global Positioning System (GPS) to limit the query flood to a restricted region. However, our approach makes intelligent use of routing histories and does not need location information. On the other hand, it delivers comparable performance advantages.

The rest of the paper is organized as follows. For ease of later discussion, we briefly review the two prominent on-demand routing protocols, DSR and AODV, in the next section. In section 3, we describe our query localization protocols. In sections 4 and 5, we evaluate the performance of the query localization protocols via simulation. In section 6, we review some related query control approaches. We conclude in section 7.

² Unidirectional links can be handled by initiating a separate route discovery back to the source from the destination [12]. For brevity, we stick to bidirectional links in our discussions.

2. On-demand routing protocols

2.1. Dynamic Source Routing (DSR)

DSR [3,12] uses source routing – a technique where the source of a data packet determines the complete sequence of nodes through which to forward the packet; the source explicitly lists this route in the packet’s header. DSR builds routes on demand, using flooded *query* packets that carry the sequence of hops they passed through. Once a query reaches the destination, the destination replies with a *reply* packet that simply copies the route from the query packet and traverses it backwards. Each node has a *route cache*, where complete routes to desired destinations are stored as learned from the reply packets. These routes are used by data packets. Route failure is detected by the failure of an attempted message transmission. Such a failure initiates an error packet sent backward to the source. The error packet erases all routes in the route caches of all intermediate nodes on its path, if the route contains the failed link.

A recent implementation of DSR by the authors of the protocol included a query containment mechanism [4]. Here, only the one-hop neighbors are queried using a single broadcast from the source. A network-wide query flood is initiated only when none of the one-hop neighbors has a route to the destination.

DSR has an unique advantage by virtue of source routing. As the route is part of the packet itself, routing loops, either short- or long-lived, cannot be formed as they can be immediately detected and eliminated. This property opens up the protocol to a variety of useful optimizations. For example, a flooded query can be quenched early by having any non-destination host reply to the query if that host has a route to the intended destination. Ordinarily, this might cause routing loops, which would require elaborate mechanisms to detect or prevent. Also, a node can learn a route to a destination while passing on route reply packets. Finally, routes can be improved by having nodes promiscuously listen to conversations between other nodes in proximity.

2.2. Ad hoc On-demand Distance Vector protocol (AODV)

AODV [19,20] is an on-demand variation of distance vector protocols. AODV uses sequence numbers maintained at destinations to determine freshness of routing information and to prevent loops. In AODV, a network-wide query flood is used to create a route, with the destination responding to the first such query, much as in DSR. However, AODV maintains routes in a distributed fashion, as routing table entries, on all intermediate nodes on the route. Routing table entries are tuples in the form of $\langle \textit{destination}, \textit{next hop}, \textit{distance} \rangle$. Nodes propagating query packets “remember” the earlier hop taken by each such query packet. This hop is used to forward the reply packet back to the source. The reply packet, in turn, sets up the routing table entries in the nodes on its path. AODV advocates use of “early quenching” of request packets, i.e., any node having a route to the destination can reply to a request.

An important feature of AODV is the maintenance of timer-based states in each node, regarding the utilization of individual routing table entries. A routing table entry is “expired” if not used recently. A set of predecessor nodes is maintained for each routing table entry, indicating the set of neighboring nodes that use that entry to route data packets. These nodes are notified with route error packets when the next hop link breaks. Each predecessor node, in turn, forwards the route error packet to its own set of predecessors, thus effectively erasing all routes using the broken link.

Neither DSR nor AODV guarantees shortest path. This is particularly true if early quenching is used. However, earlier performance evaluation shows that the lengths of the routes discovered are usually very competitive with the shortest path protocols [4,7].

3. Query localization protocols

The proposed query localization protocols are based on the notion of *spatial locality*: “a mobile node cannot move too far too soon”. Thus, prior routing histories can be cached to estimate a small region in the network with high probability of finding the destination node. Only this region needs to be flooded. In the approach we studied, prior route histories are used to limit the query to a region in the *neighborhood* of the prior routes. Hop-wise distance is used to define the neighborhood. Following the terminology used in the literature [15], we refer to this region as the *request zone*. Similar ideas have been used in the selective paging schemes in cellular networks [1], where a mobile is paged only in the neighboring cells of the last cell it reported visiting.

In the discussion that follows, assume that source routing is used, i.e., the query packet includes the path P (as a sequence of nodes) it has traversed so far (as in DSR). This assumption makes the discussion straightforward. Alternative approaches, where source routing is not used, will be explored later. Two heuristics to exploit locality are considered.

- *Exploiting path locality (Protocol 1)*. This approach relies on the assumption that after a route to the destination node breaks, the new route cannot be very different than the most recently used route. The protocol maintains a set of nodes P_{old} , which includes all nodes on the last valid route between the specific source–destination pair. During route discovery, the query flood is propagated by only such nodes for which the accumulated path P in the query packet has *at most* k nodes *not* in P_{old} . To accomplish this, the set P_{old} is sent in the header of the query packet, in addition to P . A counter is also sent as a part of the query, which is initialized to zero. The counter is incremented each time a node not in the set P_{old} is encountered by the query. The query is no longer propagated if the counter exceeds the threshold value k . (As will be discussed later, the protocol can also be implemented without including P_{old} in the query.)

Figure 1 illustrates the above protocol with an example network. Nodes S and D are the source and destina-

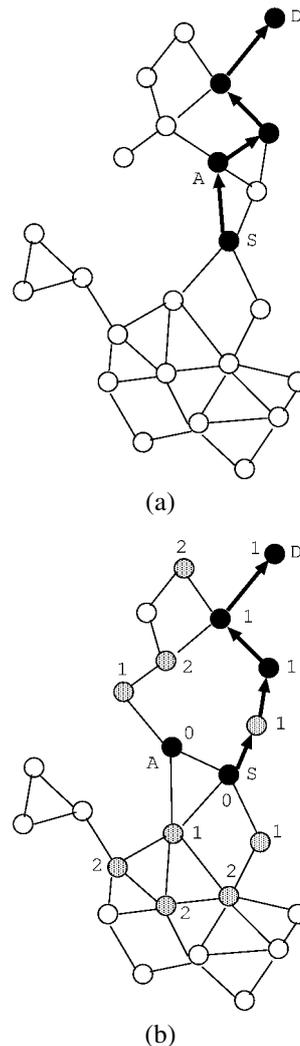


Figure 1. Illustration of the path locality principle with $k = 1$: (a) before route change, (b) after route change.

tions, respectively. Node A has moved causing a route change. The “dark” nodes were on the old route. All “filled” (dark or light) nodes are flooded with a query using the path locality heuristics with $k = 1$. The value of the counter is shown at each node visited by the query after the route change. The query is not propagated if the counter reaches 2. Note that with the naive flooding protocol, all nodes in the graph will be flooded. This is true even if flooding is controlled by a TTL field where the query is only propagated up to 4 hops (the number of hops between the source and destination is 4 in this example). Discounting the nodes on the route, query localization saves about half of the nodes in the network from being flooded in this example.

- *Exploiting node locality (Protocol 2)*. The assumption here is that the destination node can be found within a small number of hops from *some* node on the most recently used route. As before, the set of nodes P_{old} and a counter (initialized to zero) are sent as a part of the query. The counter is reset to zero whenever the query reaches a

node in the set P_{old} , otherwise it is incremented by one. As before, the query is dropped when the counter exceeds the threshold value k .

The request zones defined by the above two heuristics are different, in general. For a given value of k , the request zone defined by the node locality heuristics will subsume that defined by the path locality heuristics.

3.1. Dynamic neighborhood evaluation

The parameter k is used to limit the distribution of route queries. For minimizing the routing overhead, an appropriate value of k must be used. The optimal value of k is dependent on the mobility pattern, which may be time-varying. Too small a value of k may result in the failure of route discovery (and, thus, dropping a large number of data packets), while too large a value will increase routing overhead by enlarging the request zone. Incremental, dynamic techniques can be used to determine the right value of k . One such technique is described below.

Initially, $k = 1$. If the destination cannot be located within a reasonable timeout period, k is incremented by 1, and another query is initiated. This incremental approach leads to the right value of k (say, K). Next time around, after another route failure, instead of starting from $k = 1$, the protocol starts from $k = K - 1$, for example. A related technique can use all nodes on the last few routes in the set P_{old} , instead of only the most recent route. Another variation can be to use all routes used in the last τ time units.

3.2. Distributed maintenance of locality information

In the above protocols, we have included the set P_{old} in the query messages. It is also possible to maintain P_{old} in a distributed fashion, eliminating the need to include it in every query. Here, each node in the network maintains a flag, F_{ij} , indicating whether it has been on a route between a given pair of nodes (i, j) in the recent past (say, in the past τ time units). It is always possible to maintain this flag F_{ij} because the nodes *en route* see the route reply packet passing through them, and the other nodes do not.³

Thus, a node considers itself to be in the set P_{old} , for the source–destination pair (i, j) , only if this flag is set. The query packet now carries only the counter (initialized to zero). The counter is incremented by one, whenever a node with $F_{ij} = \text{False}$ is visited. Otherwise, the counter is not modified. The query is dropped when the counter exceeds the value k . This is similar to the path locality-based protocol considered earlier. A simple variation implements the node locality-based heuristics. Here, the counter is incremented as before, when a node with $F_{ij} = \text{False}$ is visited, but is reset to zero, when a node with $F_{ij} = \text{True}$ is visited.

³ A little ingenuity is needed for asymmetric networks, where the reply may come via a different path. In such cases, the first data packet going through a recently established route will set the flags.

This distributed implementation of P_{old} has the advantage of reducing the size of the query messages, thus saving network bandwidth. Another advantage of this mechanism is that it can also be used with protocols that do not use source routing, such as AODV [20]. One disadvantage of the distributed maintenance is that it may be hard to support alternative definitions of the set P_{old} . For example, the set P_{old} may be defined as the union of all routes found in the last n route discoveries, instead of being based solely on a timer. This will be easy to support in the original protocol, but not in the distributed variation.

3.3. Performance tradeoffs

A smaller request zone results in a lower routing overhead, by limiting the route queries to a smaller region. However, if the request zone is too small (i.e., for small values of k), a route contained within the request zone may not exist, requiring another route discovery – this would increase the latency of route discovery. To increase the probability of finding a route using the initial request zone, the size of the request zone may need to be chosen larger. However, doing so may increase the routing overhead. Thus, there is a tradeoff between the *latency* of route discovery and the *routing overhead*. A right balance of this tradeoff will reduce routing overhead significantly without increasing the route discovery latency to any appreciable extent. In our performance evaluation work we demonstrate that this is indeed possible to achieve.

It is also possible on rare occasions that the route discovered is not the shortest route. This is possible because there is no guarantee that the shortest route will be contained entirely within the request zone. This will increase the end-to-end delay of the data packets which will now go via a suboptimal route. However, our performance evaluation shows that the overall savings obtained by the reduction in routing overheads are much more significant and can actually contribute to lower delays.

4. Performance evaluation via packet-level simulation

We simulated the original version of the query localization mechanism as an add-on to the DSR protocol for a mobile, ad hoc network. An event-driven, packet-level routing simulator, called MaRS [2] is used for the simulation. MaRS has been used before for studying dynamic routing protocols in wired networks with dynamic traffic conditions and faulty links [22]. We have also used MaRS earlier for a comparative evaluation of ad hoc routing protocols [7]. MaRS simulates all relevant network layer details including queuing delays and packet processing times in each node. We extended MaRS to make the nodes mobile. The links fail and connect as nodes go away or come within the radio range of one another. Link failures and reconnects automatically generate link-layer events in the simulation model, to which the routing protocol responds. No MAC or physical layer is simulated and error-free wireless transmission without any

multiple-access interference is assumed. Even though this is a limitation of the simulation model, our earlier experience [7] with MaRS shows that it is able to make a very good qualitative evaluation at the packet level comparable to more elaborate simulation models [4].

In the model we will evaluate first, 60 mobile hosts move around a rectangular region of size $1000 \text{ m} \times 1000 \text{ m}$ according to the following mobility model. Each node chooses a direction, speed and distance of movement based on a pre-defined distribution and then computes its next position P and the time instant T of reaching that position. Similarly, a new “move” is again computed at simulation time T . A node computes its neighborhood after each such move, thus generating link failure and link repair events that in turn drive the routing protocol. Each node is assumed to have a radio range of 350 m. For the experiments reported here, the speed of each move is uniformly distributed between 3.5–4.5 m/s (low mobility experiments) and 14–18 m/s (high mobility experiments); distance is exponentially distributed with a mean of 5 m; the direction is uniformly distributed within $[+45^\circ, -45^\circ]$ with respect to the direction of the previous movement. Note that in the chosen mobility model, the nodes are always moving (*albeit* in discrete time) without stationary intervals. This presents a stress case for the routing protocol.

A simple datagram workload model is used. All data packets are 512 bytes long, and interarrival times are exponentially distributed with a mean of 300 ms. There is no acknowledgment, or flow or congestion control in the workload model. Workload traffic is always between a pair of source and sink nodes, called a *conversation*. The number of such pairs or conversations is varied over a wide range in the simulation experiments. In the performance plots, it is presented in terms of number of conversations per node in the network. All simulations are run for 300 simulated seconds and each point in a plot represents an average of five runs with different random number streams.

Three important performance metrics are evaluated: (i) *routing overhead* – measured in terms of the total number of routing packets transmitted (broadcast transmissions are counted as a single transmission); (ii) *packet delivery fraction* – measured as the ratio of the number of data packets delivered to the destination and the number of data packets sent by the source; (iii) *end-to-end delay* – measured as the average end-to-end latency of data packets. In the case of a route loss, data packets are not buffered. Thus, longer route discovery latency will typically translate to lower packet delivery fraction. Buffering of data packets, on the other hand, will cause longer delays.

4.1. Simulation results

In the first set of experimental data (figure 2), query localization protocols 1 and 2 are evaluated with various values of the threshold parameter k , chosen statically. Comparison is made against the DSR protocol using network wide flooding (labeled as DSR-NW). All protocols are identical except

how the query flooding is implemented. The plots show the number of routing packets transmitted throughout the simulation run for different values of k , with increasing number of conversations. As expected, the larger the value of k , the greater is the number of routing packets. Also, the second protocol sends more routing packets for the same value of k as it defines a larger request zone. Increasing mobility and the number of conversations also generates a larger number of routing packets.

Figure 3 shows the packet delivery fraction for the first protocol only. (From now on we only present the data for the first protocol for brevity. The data for the second protocol was found to be qualitatively similar.) As expected, lower values of k give poorer packet delivery performance, especially for higher mobility. Note that larger route discovery latency translates to higher packet loss in our simulation, as data packets are not buffered.

Figure 3 also shows the average end-to-end delay. An interesting observation here is the very large average delay in the original DSR for the high mobility case. This is due to large queuing delays in the nodes owing to the presence of a large number of routing packets. As expected, lower values of k give a lower delay.

Figure 4 shows the performance for dynamic choice of k . Here we have a conservative implementation, with k starting from 1 incrementally for each route discovery. For the sake of comparison, we also present the performance of the DSR protocol that uses query containment as described in section 2.1. This is labeled as DSR-QC. Note that the query localization protocol is able to reduce the routing overhead significantly. The savings are often in the neighborhood of 50% compared to DSR-NW and at least 20% compared to DSR-QC. Consequently, the delay is minimized significantly when data load and mobility is high. There is almost negligible impact on packet loss. The average value of k with this dynamic mechanism was found to be between 2.1–2.4, with values tending to be higher for higher mobility. Small average values of k reinforce the “spatial locality” assumption behind the query localization idea.

One problem with the chosen mobility model is that the average number of hops is small (between 3–4 for low mobility and 4–5 for high mobility). Increasing the field size does increase the average number of hops, but it also increases the probability of the network being partitioned. Network partition is a dark cloud looming over successful design of ad hoc networks with unconstrained mobility. Most performance evaluation studies so far choose mobility models such that the probability of network partition is minimal [4]. In order to increase the average hop count without any significant chance of network partition, we ran a complete set of simulations with a different choice of “field” keeping the mobility model identical. Now the field is a square area with a hole inside as shown in figure 5(a). The performance data is presented in figures 5(b), (c) and (d). Only the high mobility data is presented for brevity. Average hop counts between source and destination are now larger, between 6–9. Note that even with

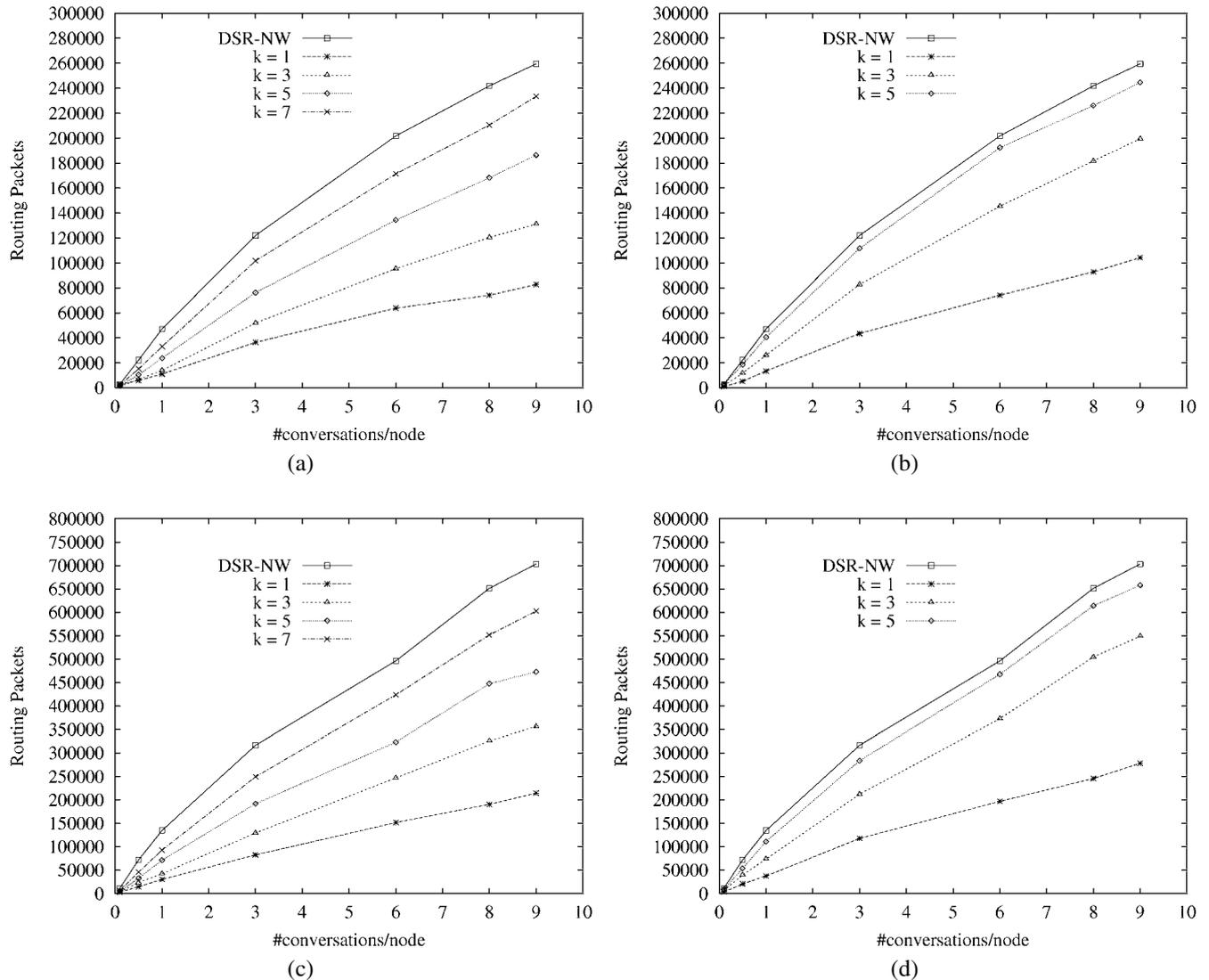


Figure 2. Routing overhead for both protocols for different values of k . (a) Low mobility, protocol 1. (b) Low mobility, protocol 2. (c) High mobility, protocol 1. (d) High mobility, protocol 2.

much larger hop counts the query localization technique performs very well.

We also ran the same models with the number of conversations/node fixed (1 and 9 conversations/node were used), while the average mobility varied over a predefined range. The average speed of the nodes were varied from 4 m/s up to 28 m/s. Figure 6 shows the results for 1 conversation/node with varying mobility, and figure 7 shows the results for 9 conversations/node. In all plots, the query localization protocol still maintains a great reduction of routing packets over DSR-NW and DSR-QC as the mobility speed is varied. Also, data packet delivery fraction is not impaired with the use of localization. Also, for a higher number of conversations/node (9), the average delay is reduced significantly. This is due to the reduction in network congestion and resulting low queuing delays (see figures 7(e) and (f)). As expected, the performance impact always tends to be greater for higher mobility.

5. Additional performance evaluation via detailed simulation

The MaRS simulation environment gives only a first order idea of the performance benefits of query localization. The simulation environment lacks lower layer models, in particular MAC and physical layers. It also uses a DSR model that does not include certain useful optimizations such as promiscuous listening. To establish the performance impact of query localization further, we used the recently developed wireless simulation models by the Monarch research group in CMU [4]. These models use the *ns-2* network simulator [8]. This simulator with the wireless extensions now includes a realistic model of a shared medium radio device with network interface, radio propagation model and a MAC protocol model following the recent IEEE 802.11 standard [11]. The DSR protocol model also includes all optimizations such as promiscuous listening, query control (as in DSR-QC) and pacing of

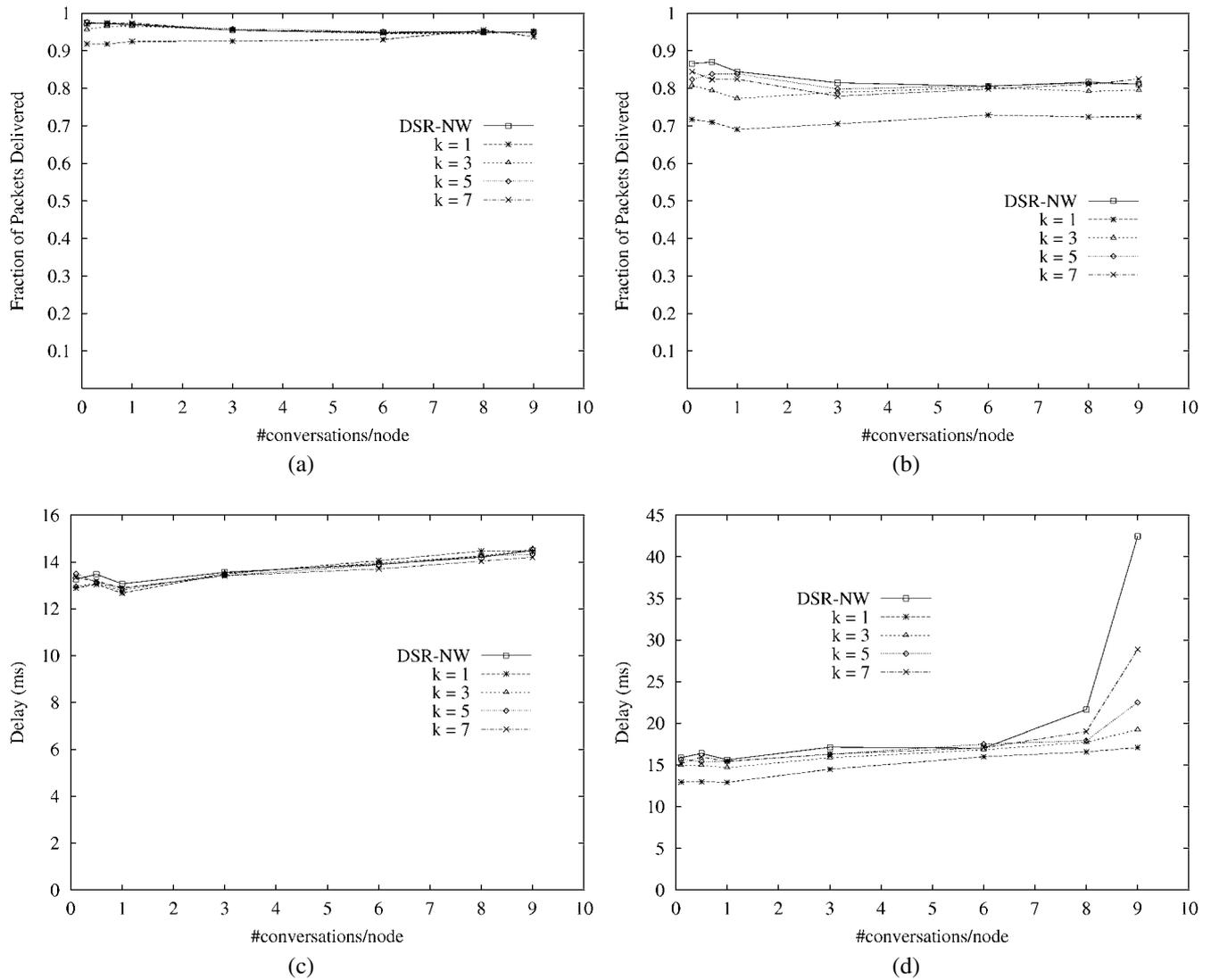


Figure 3. Packet delivery fraction and average delay for protocol 1 with different values of k . (a, c) Low mobility, (b, d) high mobility.

the route discovery process with an exponential backoff after each timeout. All these present a challenging case for query localization. More details of the *ns-2* simulator and the wireless models are available in [4,8]. We implemented query localization as a simple extension of DSR’s route discovery. Suitably optimized timeout values are chosen with the dynamic k mechanism. The path locality heuristic (protocol 1) is used.

We use 50 and 100 node models for the *ns-2* simulations. We use a *random waypoint* mobility model [4], where each node continuously moves towards a randomly chosen location with a random speed uniformly distributed between 0–20 m/s. After reaching this location, the node pauses for a fixed time, *pause time*, before moving again in a similar manner. Two rectangular field sizes have been used: (i) 1500 m × 300 m for 50 nodes and (ii) 2200 m × 600 m for 100 nodes. Each node has a nominal radio range of 250 m, though the real range depends on the channel conditions. The traffic sources are CBR (continuous bit-rate). We varied the number of con-

versations in our evaluation where each conversation corresponds to a randomly chosen source–destination pair. 64 byte packets are used uniformly. Simulations are run for 900 simulated seconds for 50 nodes and 500 simulated seconds for 100 nodes. Each point in the performance plots is an average of five runs with different mobility scenarios.

5.1. Simulation results

First set of simulation results evaluate the performance of query localization with DSR under varying mobility conditions. These results for the 50 and 100 node models are shown in figure 8. Pause time is chosen as the independent variable and is presented along the horizontal axis in each simulation. As we go from left to right in each plot, mobility gradually decreases with the rightmost point denoting no mobility. For the 50 node model, 20 conversations are used where sources in each of the conversations generate 4 packets/s. We use 40 conversations and a rate of 3 packets/s for the 100 node

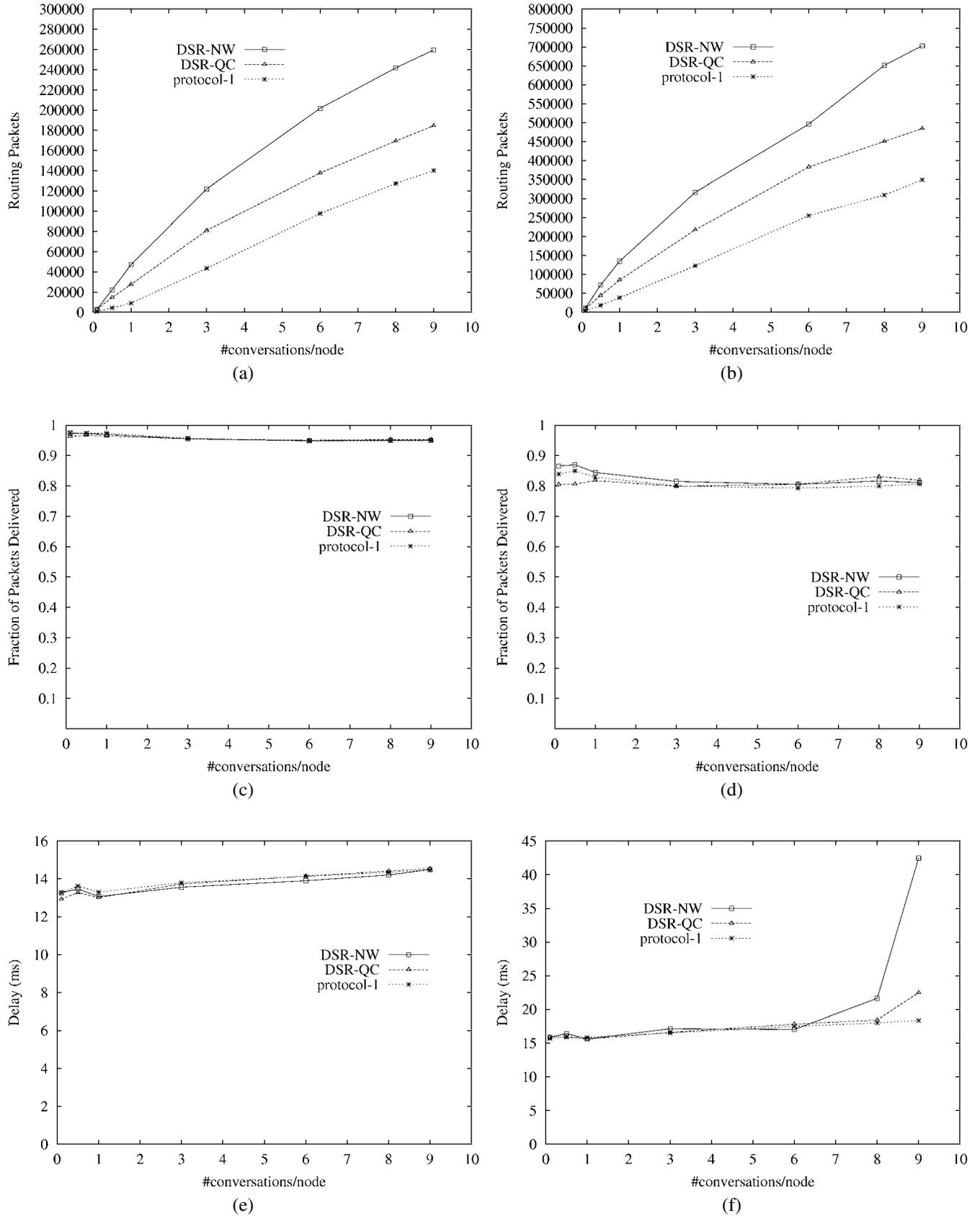


Figure 4. Various performance metrics for dynamic choice of k . (a, c, e) Low mobility, (b, d, f) high mobility. (a, b) Routing overhead, (c, d) packet delivery fraction, (e, f) average delay.

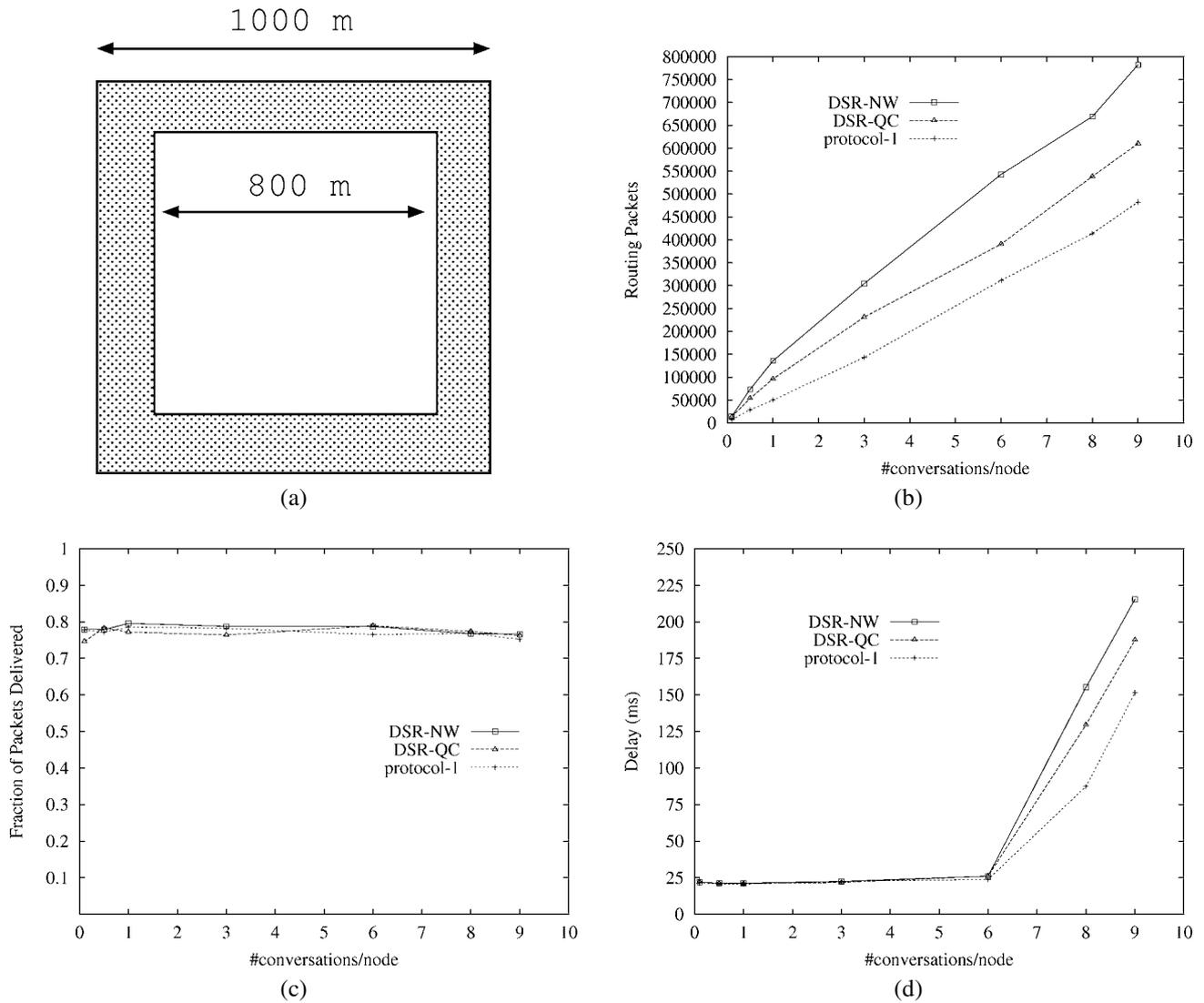


Figure 5. Performance metrics for the experiments with the square-with-a-hole region. k is chosen dynamically. Mobility is high. (a) Square-with-a-hole region. (b) Routing overhead. (c) Packet delivery fraction. (d) Average delay.

model. Note that the 50 node scenarios are identical to the performance data presented in [4] thus making the results comparable, the simulation being run using the same simulation software and models.

As before, query localization has a substantial impact on the routing overhead (figures 8(a) and (b)), saving about 30% to 40% of routing packets for the highest mobility points (zero pause times). The impact is less for lower mobility. The impact is higher for the 100 node case as the average path lengths are larger, and thus without query localization the query flood reaches a substantially larger number of nodes. For the 50 node model, packet delivery fraction and delay (figures 8(c) and (e)) are relatively unaffected with query localization. However, for the 100 node model, query localization gives improvement in both packet delivery fraction and delays. In particular for high mobility scenarios (zero pause time), packet delivery fraction with query localization improves by about 10% (figure 8(d)) and delay is reduced by

about 50% (figure 8(f)). Again this improvements are due to the lower routing load with query localization which in turn results in lower network congestion and reduced multiple-access interference. As expected, the performance gains decrease as the pause time increases (mobility decreases) since the need for route discovery also reduces proportionately.

In our second set of simulation results, we evaluate the effect of traffic load on the performance of query localization. This is done by varying number of conversations. Further, we consider the zero pause time case (high mobility) since this presents a more challenging case. Results for the 100 node model are shown in figure 9. We vary the number of conversations from 10 to 50. A fixed rate of 3 packets/s is used. Query localization results in significant reduction of routing packets in all cases with savings ranging from 40–60% (figure 9(a)). Also, degradation in application performance (packet delivery fraction and delay) with increase in traffic load is much more graceful (figures 9(b) and (c)) with query localization.

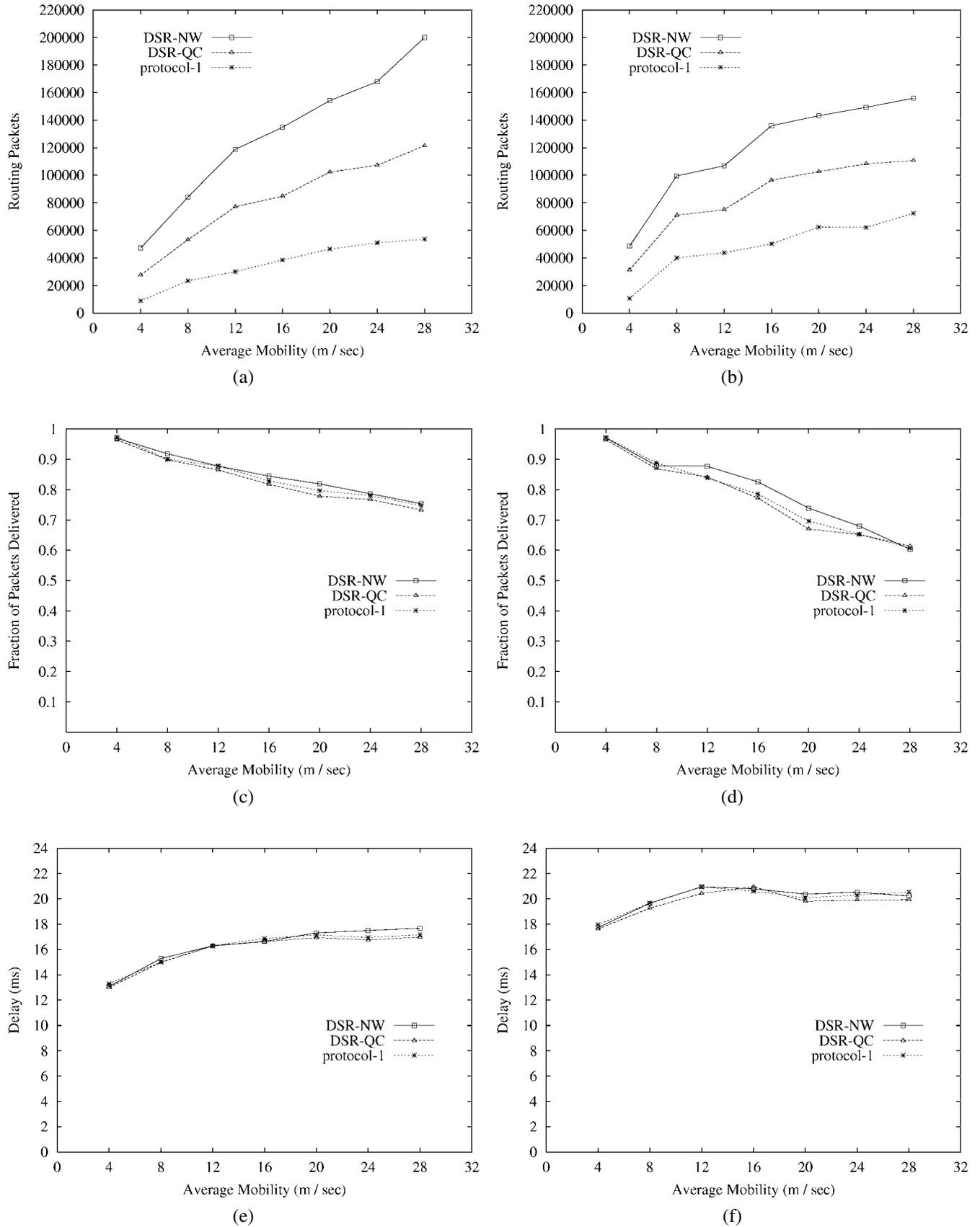


Figure 6. Performance metrics for 1 conversation/node with varying mobilities. (a, c, e) Square region, (b, d, f) square-with-a-hole region. (a, b) Routing overhead, (c, d) packet delivery fraction, (e, f) average delay.

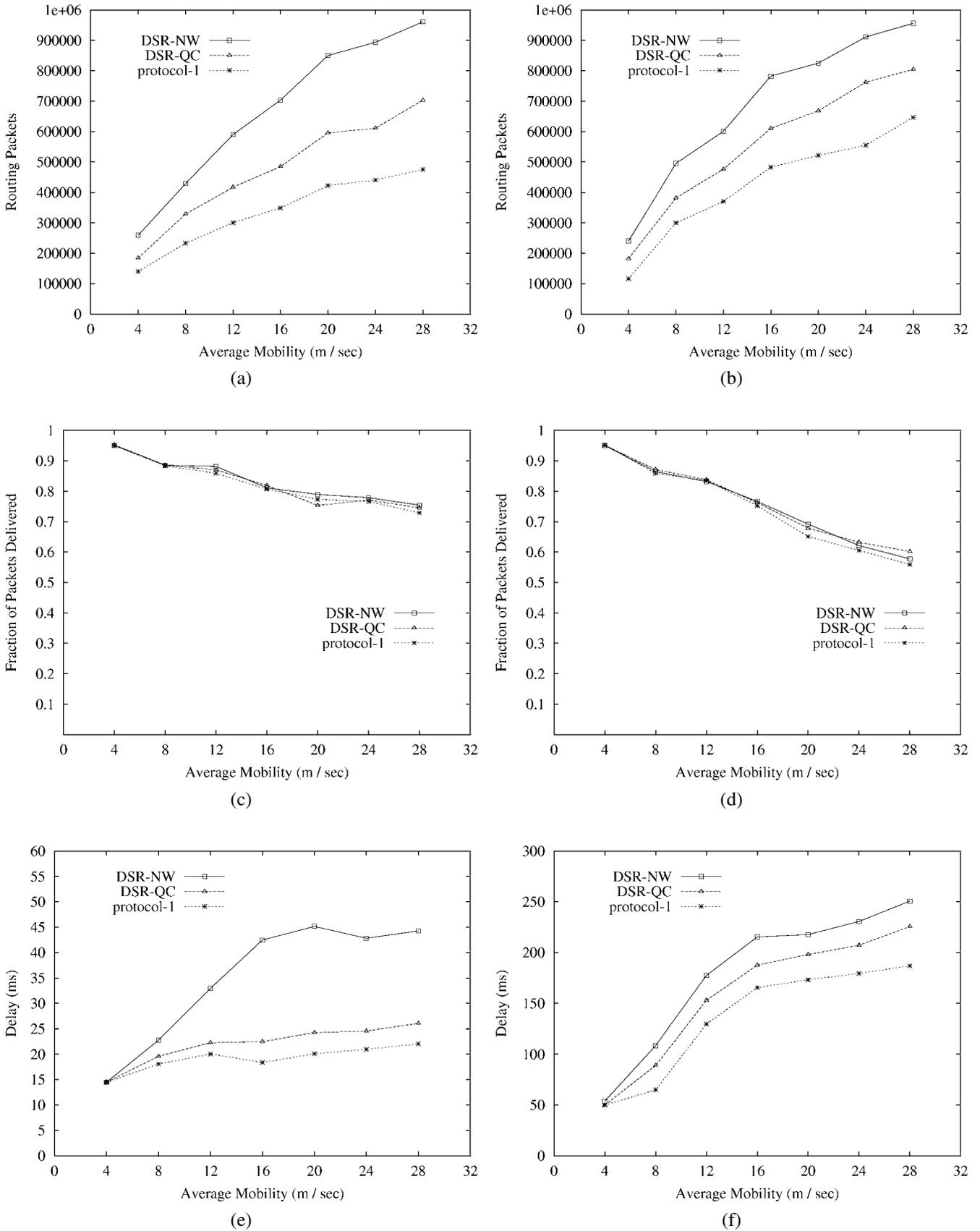


Figure 7. Performance metrics for 9 conversations/node with varying mobilities. (a, c, e) Square region, (b, d, f) square-with-a-hole region. (a, b) Routing overhead, (c, d) packet delivery fraction, (e, f) average delay.

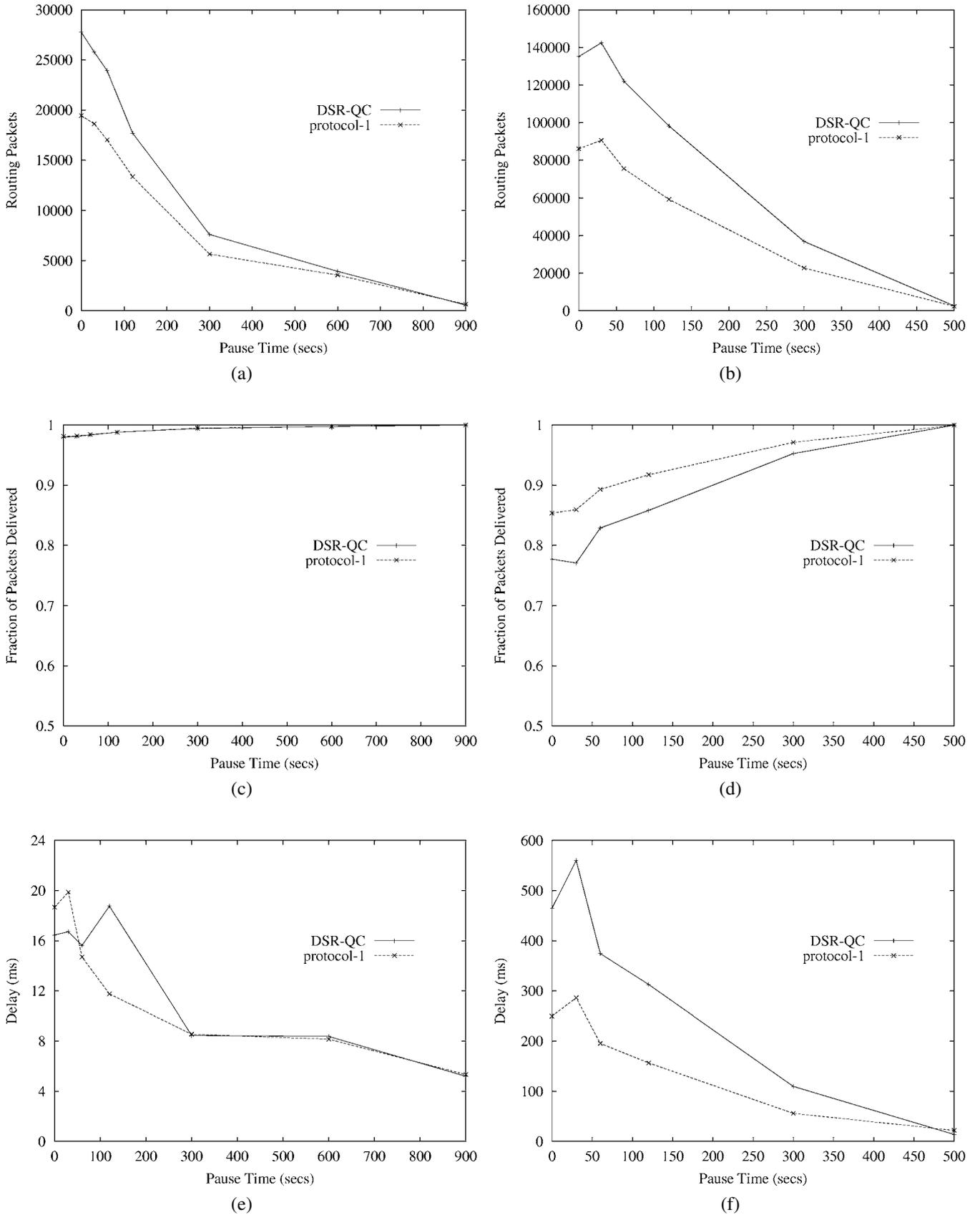


Figure 8. Performance metrics with varying pause times using *ns-2* simulations. (a, c, e) 50 mobile nodes, (b, d, f) 100 mobile nodes. (a, b) Routing overhead, (c, d) packet delivery fraction, (e, f) average delay.

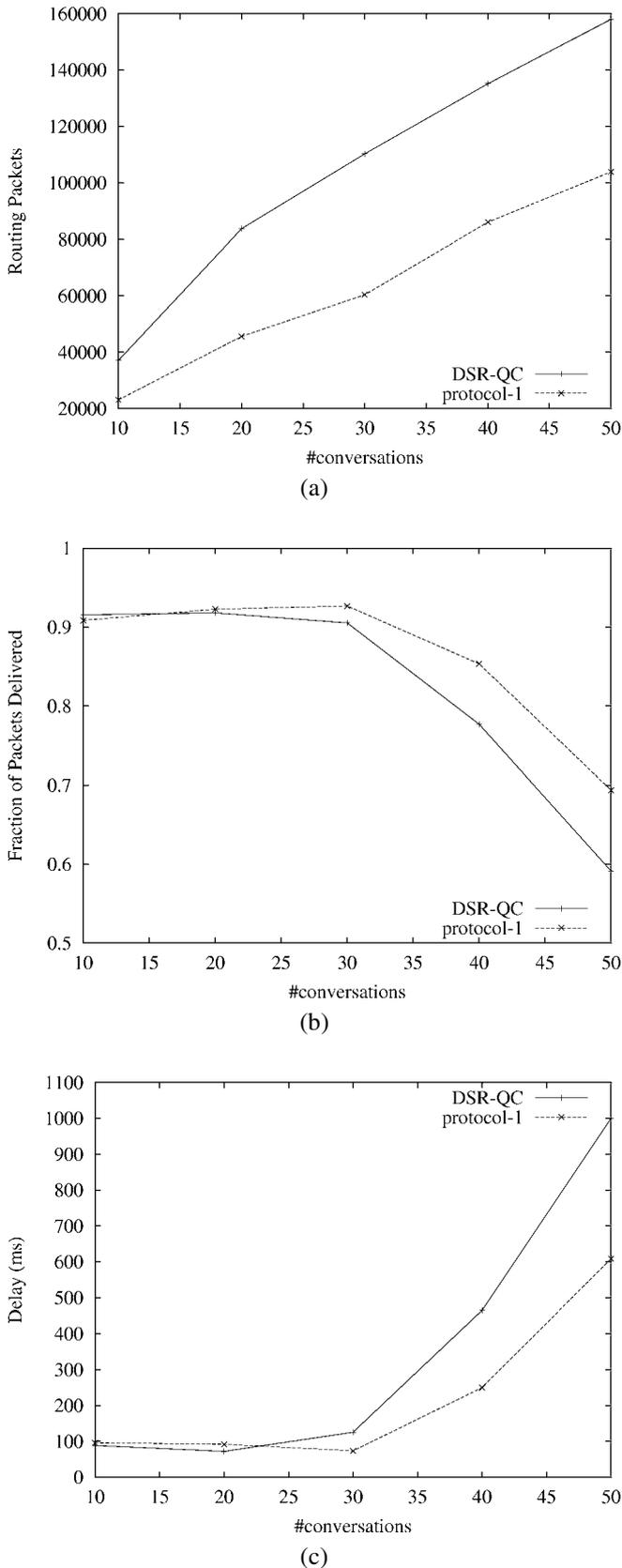


Figure 9. Performance metrics for 100 mobile nodes with varying number of conversations and zero pause time (high mobility) using ns-2 simulations. (a) Routing overhead, (b) packet delivery fraction, (c) average delay.

Relatively low routing load with query localization reduces the network congestion in high traffic cases, thus leading to substantial improvements in packet delivery fraction and delay.

6. Related work

It is not a new observation that network-wide flooding of query messages may be a performance problem in a bandwidth-poor ad hoc, wireless network. Several schemes to reduce redundant broadcasts in a query flood were proposed in [18] using analytical modeling and simulation; but they were not evaluated in the context of a routing protocol. A sophisticated technique called Location-Aided Routing (LAR) [15] has been proposed which uses the Global Positioning System (GPS) to localize queries to a limited geographic region (*request zone*). The request zone is defined based on the past location information of the destination node and its speed. As in our query localization approach, LAR can be used with any on-demand protocol. Simulation studies have shown excellent improvement of routing overhead with respect to network-wide flooding. Our approach is similar in principle to LAR and achieves similar improvements with similar mobility models, but it does not require location or speed information. This makes it attractive in many situations where GPS cannot be used because of concerns of (i) cost (GPS may be more expensive than some low-cost wireless nodes), (ii) accuracy (GPS accuracy may not be sufficient for some in-building networks with small radio range, such as Bluetooth [9]) or (iii) usability (GPS cannot be used inside many buildings).

It is also worth noting here that our “location-unaware”, topologically-based mechanism, may in fact, perform better than the location-aided mechanisms in situations where node movements are relatively fast, but highly correlated. Examples include the movement of an assault troop or a rescue team. In such situations, locality based on physical location is less relevant than locality based on neighborhood with other nodes.

A Zone Routing Protocol (ZRP) [10] has been proposed which responds to the routing overhead versus route discovery latency tradeoff question in an indirect fashion. It advocates only limited use of on-demand protocols. In ZRP the network is split into *zones* or clusters of suitable size. *Proactive* shortest path protocols are used within a zone and *reactive* on-demand protocols are used across zones. Though no explicit query control scheme was developed, the zoning scheme automatically reduces the flooding overhead. Similarly, in the recently proposed CEDAR protocol [23], only the so-called “core” nodes participate in route computation, thus limiting the flooding overhead.

The flood control problem is not unique to ad hoc networks. Other network routing protocols also face this problem. For example, in [5] the construction of multicast trees across different domains in a wide-area wired network is considered. Here, the protocol does consecutive flooding to

search increasingly larger regions until reaching the multicast tree. This technique is called *expanding rings*. This approach can also be used in ad hoc networks, by first setting $TTL = 1$, then to 2, and so on, until the destination is reached [20]. This approach still propagates the query in directions away from the destination. Also, this incremental approach may be too slow if the destination is far away and, as figure 1 demonstrates, can be no better than network-wide flooding. To the best of our knowledge, this technique has not been carefully evaluated against network-wide flooding in ad hoc networks.

In [6] the problem of distributed QoS routing in dynamic networks is considered. The goal is to find routes that have sufficient resources to meet the QoS requirement of every admitted connection. To reduce the overhead of a global search using flooding, a *ticket-based probing* technique is introduced. Here the query (called probe in the paper) flood is controlled by introducing the notion of tickets. The source issues a number of tickets that are carried by the initial query. The tickets must get distributed each time the query is propagated by a node. A query must carry at least one ticket. Thus, the number of routing messages at any time in the network is bounded by the number of tickets issued. This technique, however, does not limit the number of transmissions of query messages.

7. Conclusions

On-demand routing protocols are attractive for mobile, ad hoc networks. However, their effectiveness is limited by the need of flooding to discover new routes. We present a query localization technique where the query flooding is limited within a small region in the network. This region is determined based on prior routing histories of the concerned source-destination pair, and no physical location information of the network nodes is needed. Any on-demand routing protocol that depends on flooding can use this technique. We have evaluated query localization using the Dynamic Source Routing (DSR) protocol as a test case. Performance evaluation using a packet-level routing simulator demonstrates excellent reduction of routing overheads. The savings are often around 50% compared to network-wide flooding and almost always more than 20% compared to a simple query containment technique. Performance evaluation using a more detailed simulation model shows qualitatively similar results, and demonstrates a greater impact (up to 60%) of query localization for larger networks. Reduction of routing load also indirectly contributes to lower end-to-end delays for data packets because of reduced network congestion and multiple-access interference. The reduction in delay was found to be close to 50% for high mobility scenarios.

The query control scheme presented here should also find applications in other forms of dynamic networks, where network-wide flooding is used to locate specific nodes or groups of nodes. Our future plan is to identify such applications. We also plan to evaluate the effectiveness of our ap-

proach in various other mobility and traffic scenarios, and also with other on-demand protocols.

Acknowledgment

This work is partially supported by AFOSR grant No. F49260-96-1-0472, NSF MII grant No. CDA-9633299, NSF CAREER award No. ACI-9733836 and NSF grant No. ANI-9973147.

References

- [1] I.F. Akyildiz, J.S.M. Ho and Y.-B. Lin, Movement-based location update and selective paging for PCS network, *IEEE/ACM Transactions on Networking* 4(4) (August 1996) 629–638.
- [2] C. Alaettinoglu, A.U. Shankar, K. Dussa-Zieger and I. Matta, Design and implementation of MaRS: A routing testbed, *Journal of Internet-working: Research and Experience* 5(1) (1994) 17–41.
- [3] J. Broch, D. Johnson and D. Maltz, The dynamic source routing protocol for mobile ad hoc networks, IETF Internet Draft (work in progress) (October 1999) <http://www.ietf.org/internet-drafts/draft-ietf-manet-dsr-03.txt>
- [4] J. Broch, D.A. Maltz, D.B. Johnson, Y.-C. Hu and J. Jetcheva, A performance comparison of multi-hop wireless ad hoc network routing protocols, in: *Proceedings of IEEE/ACM MOBICOM'98* (October 1998) pp. 85–97.
- [5] K. Carlberg and J. Crowcroft, Building shared trees using a one-to-many joining mechanism, *ACM Computer Communication Review* (January 1997) 5–11.
- [6] S. Chen and K. Nahrstedt, Distributed QoS routing with imprecise state information, in: *7th International Conference on Computer Communications and Networks (IC3N)* (October 1998).
- [7] S.R. Das, R. Castañeda, J. Yan and R. Sengupta, Comparative performance evaluation of routing protocols for mobile, ad hoc networks, in: *7th International Conference on Computer Communications and Networks (IC3N)* (October 1998) pp. 153–161.
- [8] K. Fall and K. Varadhan (eds.), *ns notes and documentation* (1999) <http://www-mash.cs.berkeley.edu/ns/>
- [9] J. Haarsten et al., Bluetooth: Vision, goals, and architecture, *ACM SIGMOBILE Mobile Computing and Communications Review* 2(4) (October 1998) 38–45.
- [10] Z.J. Haas and M.R. Pearlman, The performance of query control schemes for the zone routing protocol, in: *Proceedings of ACM SIGCOMM'98 Conference*, Vancouver (September 1998) pp. 167–177.
- [11] IEEE Standards Department, Wireless LAN medium access control (MAC) and physical layer (PHY) specifications, IEEE standard 802.11–1997 (1997).
- [12] D. Johnson and D. Maltz, Dynamic source routing in ad hoc wireless networks, in: *Mobile Computing*, eds. T. Imielinski and H. Korth (Kluwer Academic, 1996) chapter 5.
- [13] J. Jubin and J.D. Tornow, The DARPA packet radio network protocols, *Proceedings of the IEEE* 75(1) (January 1987) 21–32.
- [14] S. Keshav, *An Engineering Approach to Computer Networking: ATM Networks, the Internet, and the Telephone Network* (Addison-Wesley, 1997) chapter 11.
- [15] Y. Ko and N.H. Vaidya, Location-aided routing (LAR) in mobile ad hoc networks, in: *ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM)* (November 1998) pp. 66–75.
- [16] J. Macker and S. Corson, Mobile ad hoc networks (MANET), IETF Working Group Charter (1997) <http://www.ietf.org/html.charters/manet-charter.html>
- [17] K.J. Negus et al., HomeRF and SWAP: Wireless networking for the connected home, *ACM SIGMOBILE Mobile Computing and Communications Review* 2(4) (October 1998) 28–37.

[18] S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen and J.-P. Sheu, The broadcast storm problem in a mobile ad hoc network, in: *Proceedings of the 5th International Conference on Mobile Computing and Networking (ACM MOBICOM'99)*, Seattle (August 1999) pp. 151–162.

[19] 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* (February 1999) pp. 90–100.

[20] C.E. Perkins, E.M. Royer and S.R. Das, Ad hoc on demand distance vector (AODV) routing, IETF Internet Draft (work in progress) (July 2000) <http://www.ietf.org/internet-drafts/draft-ietf-manet-aodv-06.txt>

[21] N. Schacham and J. Westcott, Future directions in packet radio architectures and protocols, *Proceedings of the IEEE 75(1)* (January 1987) 83–99.

[22] A.U. Shankar, C. Alaettinoglu, K. Dussa-Zieger and I. Matta, Transient and steady-state performance of routing protocols: Distance-vector versus link-state, *Journal of Internetworking: Research and Experience 6* (1995) 59–87. Preliminary version in: *Proceedings of ACM SIGMETRICS/PERFORMANCE Conference* (1992) pp. 181–192.

[23] R. Sivakumar, P. Sinha and V. Bharghavan, CEDAR: A core-extraction distributed ad hoc routing algorithm, *IEEE Journal on Selected Areas in Communications, Special Issue on Ad Hoc Networks 17(8)* (August 1999) 1454–1465.

Robert Castañeda got his B.S. degree in 1990, M.S. degree in 1994, and his Ph.D. degree in 2000, all in computer science from the University of Texas at San Antonio. His interests are in wireless networking, simulation, and performance evaluation of parallel and distributed architectures systems. Since then, he has pursued work in industry within the area of wireless data networking.
E-mail: rcastane@cs.utsa.edu



Samir R. Das is currently an Associate Professor in the Department of Electrical and Computer Engineering and Computer Science at University of Cincinnati. Prior to this he was a faculty member in the Division of Computer Science at The University of Texas at San Antonio. He received his B.E. degree in electronics and telecommunication engineering from Jadavpur University, Calcutta, in 1986; M.E. degree in computer science and engineering from the Indian Institute of Science, Bangalore, in

1988; and Ph.D. degree in computer science from the Georgia Institute of Technology, Atlanta, in 1994. He also held research positions in Indian Statistical Institute in Calcutta, India, and Sun Microsystems Lab in Palo Alto. His research interests include mobile/wireless networking, performance evaluation and parallel discrete event simulation. He has published over 30 research articles on these topics. Dr. Das received the US National Science Foundation's Faculty Early CAREER award in 1998. He will be co-chairing the technical program committee of the second Mobile Ad Hoc Networking and Computing (MobiHOC) Symposium in 2001. He co-guest-edited a special issue of the "Computer Communications" journal in 2000. He also served on the program and organizing committees of several prominent workshops and conferences related to networking, distributed computing, simulation and performance evaluation, including PADS, MASCOTS, IC3N, ICDCS, MobiHOC and MobiCom. He is a member of the IEEE, IEEE Computer Society and ACM.
E-mail: sdas@ececs.uc.edu



Mahesh K. Marina (ACM S'98) is a Ph.D. student in computer science at the University of Cincinnati. He received his B.Tech. degree in computer science and engineering from the Regional Engineering College, Warangal, India, in 1998 and his M.S. degree in computer science from the University of Texas at San Antonio in 1999. His research interests are in the areas of mobile/wireless networking, performance analysis and distributed algorithms. The focus of his current work is on protocols for ad hoc

networks.
E-mail: mmarina@ececs.uc.edu