PROTOCOLS FOR MOBILE AD HOC NETWORKING

APPROVED BY SUPERVISING COMMITTEE:

____

Dr. Samir R. Das, Supervising Professor

____

Dr. Robert Hiromoto

____

Dr. Kleanthis Psarris

____

Dr. Weining Zhang

____

Dr. Parimal Patel, Outside Member

Accepted: ______________________________________

Dean of Graduate Studies

Dedication

This dissertation is dedicated to:

- **Robert Gabriel Castañeda**, my son;

AND

- **Henry G. Castañeda**, my father, and
- **Romana Castañeda**, my mother.

Without their love, support, and encouragement I could not have completed the required research for this dissertation. Thank you.

PROTOCOLS FOR MOBILE AD HOC NETWORKING

by

ROBERT CASTAÑEDA, B.S., M.S.

DISSERTATION
Presented to the Graduate Faculty of
The University of Texas at San Antonio
in Partial Fulfillment
of the Requirements
for the Degree of

DOCTOR OF PHILOSOPHY IN COMPUTER SCIENCE

THE UNIVERSITY OF TEXAS AT SAN ANTONIO
College of Sciences and Engineering
Division of Computer Science
December 2000

Acknowledgements

I would like to express sincere gratitude for the time, encouragement, and guidance provided by my advisor, Dr. Samir R. Das whose help cannot be truly expressed in words. I also would like to thank Dr. Robert Hiromoto, Dr. Weining Zhang, Dr. Kleanthis Psarris, and Dr. Parimal Patel for serving on my final Ph.D committee and for all of their support and suggestions.

This research was supported by the AFOSR grant F49260-96-1-0472, Texas Advanced Technology Program grant 010115-248b, National Science Foundation grant ACI-9733836, NSF MII grant CDA-9633299, and the University of Texas at San Antonio Division of Computer Science. The research for this dissertation was conducted at the University of Texas at San Antonio within the laboratories of the Division of Computer Science.

I would also like to thank all of my friends with whom I made during the past five years on my path toward earning my doctoral degree. Thanks for all the great times and for all the encouragement and suggestions. It made this journey much more pleasant and meaningful. Thank you all.

August 2000

PROTOCOLS FOR MOBILE AD HOC NETWORKING

Robert Castañeda, B.S., M.S., Ph.D.
The University of Texas at San Antonio, 2000

Supervising Professor: Samir R. Das

A mobile ad hoc network is an autonomous system of mobile hosts connected by wireless links. Such networks are useful in military and other tactical applications, e.g., emergency rescue or exploration missions, where cellular infrastructure is unavailable or unusable. Because of constantly changing topology, ad hoc networks must adopt efficient dynamic routing techniques. Routing overhead is one important concern as the wireless link bandwidth is typically low. Recently, a new class of protocols called “on-demand” protocols has received attention because of their potential for low routing overhead. On-demand protocols discover routes on an “as-needed” basis, rather than in a more proactive fashion, as in the traditional link state or distance vector protocols. This dissertation presents a comparative simulation study of on-demand protocols with the more traditional proactive protocols on a common platform across a range of traffic and mobility scenarios. The simulation results show that on-demand protocols indeed demonstrate low routing overheads, but over-reliance on query flooding can actually cause on-demand protocols to lose most of this overhead advantage at high loads.

As a countermeasure, we develop and evaluate two techniques that reduces the routing overhead for on-demand protocols. The first technique, called *query localization*, uses certain locality heuristics to prevent network-wide query flood. It effectively limits the query to a small region where the route is very likely to be found. The second technique, called *multipath routing*, focuses on reducing the frequency of query flooding by exploring multiple, disjoint routes per flood operation. This provides the routing protocol with alternate routes when the primary route breaks. Simulation results show that either of these techniques can reduce routing overheads significantly.

Unlike stationary hosts, mobile hosts can operate only as long as the their batteries last. Thus, battery power conservation is an important component of a routing protocol. We describe a “power-aware” mechanism that can work with any on-demand routing protocol to offload the routing responsibilities from the nodes running low on battery power towards the nodes with more power. The simulation results demonstrate that this technique distributes the available battery power more evenly and improves the operation lifetime of the ad hoc network.

Contents

List of Figures

Chapter 1

Introduction

A mobile, *ad hoc* network [12] is an autonomous system of mobile hosts connected by wireless links (Figure 1.1). There is no static infrastructure such as base stations. If two hosts are not within radio range of each other, 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. 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 on-the-fly conferencing applications, networking intelligent devices or sensors, etc.

Interest in such dynamic wireless networks is not new. It dates back to the seventies, when the U.S. Defense Research Agency, DARPA worked on PRNET (Packet radio Network) [39] and SURAN (Survivable Adaptive Networks) [60] 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 U.S.). In an interest to run internetworking protocols on ad hoc networks, a new working group for Mobile, Ad hoc Networking (MANET) has been formed within the Internet Engineering Task Force (IETF) [48], whose charter includes developing a framework for running IP (Internet Protocol)-based protocols in ad-hoc networks. Interest has

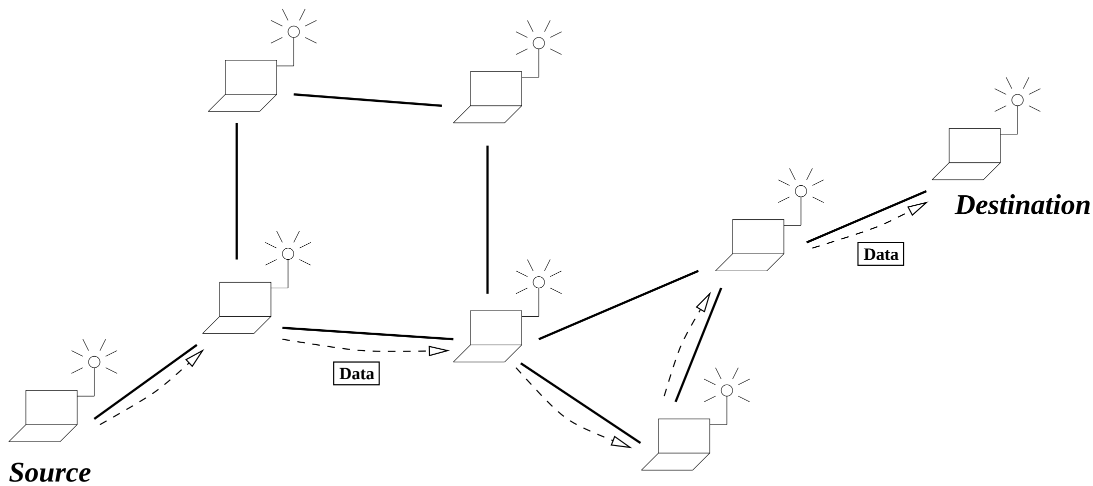


Figure 1.1: A mobile ad hoc network.

also been partially fueled by the recent IEEE standard 802.11 [19] that includes the MAC (Medium Access Control) protocol and physical layer specifications for wireless LANs without any fixed infrastructure.

Efficient packet routing is the key research issue in ad hoc networks. Several characteristics of the ad hoc networks make the routing problem very different than wired networks. First, dynamically changing network connectivity requires that routes be updated frequently. Second, low bandwidth of wireless links necessitates that routing overhead be kept low, so that only a small fraction of network bandwidth is spent on transmitting routing packets (as opposed to data packets). Third, quick convergence of the routing protocol may be crucial even if the routes obtained may be sub-optimal. This is because spending a long time to obtain a high quality route (e.g., shortest path) may not be very productive, as routes may change by the time route discovery is complete. Thus, there exists interesting trade-offs between the quality of the route, and time and overhead spent in discovering that route. Fourth, the longevity of a node's battery becomes an issue since they will be using battery energy to sustain life within an ad hoc network. Routing protocols designed with energy conservation in mind may become an important criteria that may need to be addressed.

Several *qualitative* properties for designing a routing protocol are desired for a mobile ad hoc network. First, the operation of the protocol should be distributed and not centralized to any one location. A centralized protocol leaves open the door for congestion problems. Also, the whole network will fail if the central controller node fails. Second, the protocol should find loop-free routes. Third, the protocol must have the ability to adapt to the topology and traffic patterns of the network. The protocol should be able to do this with low routing overhead. Fourth, availability of undirectional or bidirectional links will influence the performance of a routing protocol. Designing routing protocols with the assumption of bidirectional links is usually the norm and makes the functionality of the algorithms easier. In real life, networks may not have this luxury and may in fact have unidirectional links that may incur substandard performance on protocols that work well with bidirectional links.

There are several *quantitative* performance metrics that can be used to assess the performance of routing protocols within a mobile ad hoc network. First, throughput and end-to-end delay are typical performance measures that show a routing protocol's effectiveness in doing its job (i.e. delivering data packets). Second, for certain protocols that acquire routes on-demand the amount of time it takes to acquire a route or *route discovery latency* is also an important performance measure. This measurement more easily conforms to those protocols that are of a demand-base property and thus should be acquired. Third, bandwidth utilization should be observed to see how effective the protocol is if both routing packets and data packets share the same channel. One such measure would be to obtain the number of bytes (or packets) of routing packets transmitted per number of bytes (or packets) of data packets delivered. Another such measurement may be the amount of data bits transmitted per data bit delivered to show the efficiency of data delivery throughout the network.

The objective of this dissertation is understanding and improving the performance of routing protocols for mobile ad hoc networks. In an attempt to understand the performance issues better, we have performed a comparative performance evaluation of some existing routing protocols of a different nature under varying traffic and mobility conditions. Our work is one of the first major comparative evaluation work reported that has been reported in literature [17, 18]. We will then focus on a class of routing protocols, called "on-demand" protocols, that show a significant promise compared to others because of their low routing overhead, and design techniques to reduce their routing overheads further. Two different techniques, *query localization* and *multipath routing* are proposed and evaluated to reduce the routing overhead. Finally, we turned our attention to battery power management for on-demand protocols and develop techniques that limit power usage without any significant negative impact on routing performance. Power management of the routing protocols are important as ad hoc networks are expected to be primarily battery operated without many opportunities for frequent recharging of batteries. At the end, we will present our conclusions and pointers to future research.

In the rest of this chapter we review current state-of-the-art in ad hoc network research. Specifically, we review two areas: unicast routing protocols and power management. Link layer issues such as medium access control or MAC protocols or other routing protocols, such as multicast are somewhat orthogonal to our interests and are not reviewed here.

1.1 Unicast Routing Protocols

Routing protocols in packet-switched networks have traditionally used either a *link-state* or *distance-vector* type routing algorithm [42]. Both algorithms allow a host to find the next hop neighbor to reach the destination via the "shortest path." The shortest path is usually in terms of the number of hops; however, other suitable cost measures such as link utilization or queueing delay can also be used. Such shortest path protocols have been successfully used in many dynamic packet-switched networks. Prominent examples include usage of the link-state protocol in OSPF (Open Shortest Path First) [49] and use of the distance-vector protocol in RIP (Routing Information Protocol) [33] for interior routing in the Internet. Even though any such protocol would, in principle, work for ad hoc networks, a number of protocols has been specifically developed for use with ad hoc networks. The primary motivation is that the shortest path protocols, either link-state or distance-vector, take too long to converge and have a high message complexity [12]. Because of the limited bandwidth of wireless links, message complexity must be kept low. Also, a potentially rapidly changing topology makes it important to find routes quickly, even if the route may be suboptimal [12].

Several new ad hoc routing protocols have been developed with this basic philosophy. They, however, vary widely in characteristics. For example, some of these protocols are variations of distance vector routing. Some protocols explicitly maintain redundant routing paths so that alternatives are available when a route changes. Some recently proposed protocols use a *reactive* approach for route discovery and maintenance, instead of the more traditional, *proactive* approach [32]. In a reactive approach protocols are "source initiated;" routes are discovered and maintained on an as needed basis, thus circumventing large overheads of always maintaining routes between all possible source and destination pairs. The protocols are briefly reviewed in the following section.

1.1.1 Link State Protocols

Each node maintains its own view of the network topology, including link costs of all its outgoing links. To keep views up-to-date, each node broadcasts the link costs of all its neighbors[1] to all other nodes in the network using flooding. This is done whenever there is a change in link costs. As a node receives this information, it updates its view of the network topology and applies a shortest path algorithm (Dijkstra's shortest path algorithm [21] in our simulations) to choose the next hop to a destination. Asynchronous link cost updates may give rise to short-lived routing loops; however, they disappear by the time update messages have propagated throughout the network [42]. In our simulation we used the SPF implementation of link state protocol as described in [61].

1.1.2 Distance Vector Protocols

In the distance vector approach, for each destination i, every node j maintains a set of distances or costs, $d_{ik}(j)$, where k ranges over the neighbors of i. Node k is treated as the next hop node for a data packet destined for i, if $d_{ik}(j) = \min \forall k\{d_{ik}(j)\}$. To keep these distances up-to-date, whenever there is any change of this minimum distance because of link cost changes, the new minimum distance is reported to the neighboring nodes. If, as a result, a minimum distance to any neighbor changes, this process is repeated. This technique is the classical distributed Bellman-Ford algorithm [4].

Routing loops, both short-lived and long-lived, are possible in the distributed Bellman-Ford algorithm. There is also a possibility of the *counting-to-infinity* problem, where it takes a large number of update messages to detect that a node is unreachable [42]. Several protocols have been proposed to avoid the long-lived loops and counting-to-infinity problems. They typically work by increasing the amount of information exchanged between nodes or providing some sort of inter-nodal coordination. For example, in the Border Gateway Protocol (BGP) the entire path between the source and the destination is sent instead of just the

[1] In a mobile, ad hoc network any node can potentially be a neighbor.

distance [59]. In DUAL (Distributed Update Algorithm) [28] inter-nodal coordination is achieved via a technique known as *diffusing computation.*

Here, we focus our attention on two distance vector protocols which show the most promise for ad hoc networks. The first, extended Bellman-Ford, achieved good performance for stationary networks in earlier simulation studies [61]. The second, DSDV [56], was specifically proposed for mobile, ad hoc networks.

Extended Bellman-Ford Extended Bellman-Ford [11] augments the classical Bellman-Ford by maintaining on node j, (in addition to the set of distances $d_{ik}(j)$) a set of nodes $N_{ik}(j)$, which immediately precedes the destination i in the path from j to i via neighbor k. Then it is possible for the source node i to construct the whole path to the destination, by repeatedly using the preceding node $N_{ik}(j)$ as a new destination. It can be shown that the protocol is free from both long-lived loops and counting-to-infinity problems, if each node avoids sending route change updates to a neighbor for any destination d, if that neighbor is in the path to d.

We used the EXBF implementation described in [61] for our evaluation. Here, periodically or whenever a failure or reconnect occurs, link costs are recalculated, and if there is a change in the minimum distance, the new minimum distance is reported to the neighboring nodes. Several protocols proposed in the literature are also based on a similar idea of maintaining the second-to-last hop (predecessor) for the shortest path to each destination to achieve loop freedom. See [50] and the references therein.

DSDV The *destination sequenced distance vector* or DSDV protocol [56] has been specifically targeted for mobile networks. DSDV augments the classical, distributed Bellman-Ford by tagging each distance entry $d_{ik}(j)$ by a sequence number that originated in the destination node i. Each node maintains this sequence number, incrementing it each time the node sends an update to the neighbors. The sequence number is disseminated in the network via update messages. The destination sequence number is used to determine the "freshness" of a route. Always the latest sequence number is used for updating routes. For equal sequence numbers, the one with the smallest distance metric is used. It has been shown that DSDV avoids the long-lived loops and counting-to-infinity problems [56].

1.1.3 Multipath Protocols – TORA

The unique feature of the *temporally ordered routing algorithm* or TORA [54] is the maintaining of multiple routes to the destination so that many topological changes do not require any reaction at all, as having just a single route is sufficient. The protocol reacts only when *all* routes to the destination are lost. In that case routes are re-established via a temporally ordered sequence of diffusing computations, which are essentially *link reversals*, (to be described momentarily). In the event of network partitions, the protocol is able to detect the partition and erase all invalid routes.

TORA is based, in part, on the classical work by Gafni and Bertsekas [27], who consider a similar problem of maintaining a *destination oriented* directed acyclic graph (DAG) in the face of topological changes. A DAG is considered destination oriented, if for every node, there is a path to a given destination. If link failures make such a graph "destination disoriented," a series of link reversals ensue so that the graph again becomes destination oriented in finite time. The graph is initially constructed (route discovery or construction phase) in a "source-initiated" fashion, using a *query* flood followed by *update* routing packets. From that point it is maintained (route maintenance phase) using link reversals alone, whenever a topological change causes a node to lose its last downstream link. If the destination becomes unreachable because of a network partition, the protocol erases (route erasure phase) all invalid routes.

TORA uses a notion of node "height" to maintain the destination oriented DAG. Each node maintains a height and exchanges this value with each neighbor. The significance of the height is that a link is always directed from a "higher" node to a "lower" node. Note that this notion of height and link directions are destination specific. Independent copies of the protocol runs for each possible destination node in the network.

In the initial route construction phase, the height of a node carries the notion of distance (in hops) of the node from the destination. However, this distance information is eventually lost during route maintenance phase. Since multiple routes are maintained in TORA, an obvious question is the choice of route. Two alternatives are suggested – choosing a neighbor randomly so that the loads are more or less evenly distributed or choosing the lowest neighbor [54]. We have chosen the latter in our evaluations reported in a later chapter.

1.1.4 On-Demand Protocols

Link-state and distance-vector protocols are primarily *proactive* protocols in the sense that routes are maintained to all potential destinations (possibly all nodes in the network) at all times, regardless of whether or not all such routes are actually used. Route maintenance can be a large overhead because of a significant amount of route update traffic, especially for large networks. *Reactive* or on-demand protocols, on the other hand, create and maintain routes only on an “as needed” basis. Thus, when a route is needed, some sort of global search procedure is employed via *flooding*. Flooding is implemented as follows. When a “source” node S needs to find a route to a “destination” node D, node S broadcasts a *route query* message to all its neighbors.[2] A node, say X, on receiving a route query message, compares the desired destination with its own identifier. If there is a match, it means that the query is for a route to itself (i.e., node X). Otherwise, node X broadcasts the query to its neighbors. To avoid redundant transmissions of route queries, a node X broadcasts a particular route query only once. Repeated reception of a route query is detected by the use of sequence numbers. This flooding mechanism guarantees that if the destination is reachable from the source, the query message 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 symmetric, the reply goes back to the source just by retracing the route in the opposite direction. The actual mechanics of doing this varies from protocol to protocol. For example, the DSR (Dynamic Source Routing) protocol [38], 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 Ad Hoc On Demand Distance Vector (AODV) protocol [55], on the other hand, maintains the route in specific data structures on the nodes on the route.

DSR and AODV are the prime examples of reactive or on-demand protocols, and are described in further detail below as many of our later evaluations will deal with these protocols. Note that TORA [54], described earlier is also partially reactive in the sense that route creation is initiated on-demand. However, route

[2] A single transmission reaches all neighbors due to inherent broadcast property of the wireless medium. Two nodes are said to be neighbors if they can directly communicate with each other.

maintenance is done on a proactive basis such that multiple routing options are available in case of link failures.

DSR Dynamic source routing or DSR [38] uses 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. [3] Each node has a *route cache*, where complete routes to desired destinations are stored as gleaned 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 [6]. 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. A node can learn a route to a destination while passing on route reply packets. Also, routes can be improved by having nodes promiscuously listen to conversations between other nodes in proximity. We haven't implemented this last optimization, however.

AODV Ad hoc, on-demand distance vector protocol or AODV [55] is an on-demand variation of distance vector protocols. AODV uses destination sequence numbers like DSDV to determine freshness of routing information. In AODV, flooded requests are used to create a route, with the destination responding to the first

[3] We assume that the wireless links are symmetric, which may not be the case in practice. DSR can tolerate asymmetric links by, for example, using an independent route discovery from destination back to the source. We did not consider asymmetric links in our evaluations in this work.

such request, 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 $<$*destination, next hop, distance*$>$. Nodes forwarding queries remember the earlier hop taken by the query packet. This hop is used to forward the reply packet back to the source. The reply packet sets up the routing table entries 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. AODV also uses a technique called *route expiry*, where a routing table entry expires after a pre-determined period, after which a fresh route discovery must be initiated.

AODV maintains the addresses of the neighbors through which packets destined for a given destination were received. A neighbor is considered *active* (for a destination), if it originates or relays at least one packet for that destination, within the past *active timeout* period. A routing table entry is *active* if it is used by an active neighbor. The path from a source to a destination via the active routing table entries is called an *active path.* On a link failure, all routing table entries are erased for which the failed link is on the active path. This is accomplished by an error packet going backwards to the active neighbors, which forward them to their active neighbors and so on. This technique effectively erases the route backwards from the failed link.

Neither DSR nor AODV guarantees the shortest path. If the destination alone can respond to route requests (i.e., early quenching of route requests is not used) and the source node (and not an intermediate node) is always the initiator of the route request, the initial route may be the shortest. But depending on the changes in topology this route may not always remain the shortest.

1.1.5 Other Protocols

Several other protocols have appeared in literature for mobile, ad hoc networks. Zone routing protocol (ZRP) [30, 31, 32] is a zone or cluster based routing protocol that is a hybrid between proactive and reactive routing. It is targeted for very large networks and divides the network into zones or clusters of nodes. The nodes within a zone are close to one another.

Cluster Based Routing Protocol (CBRP) [36] is another cluster based protocol that is based on source routing similar to that used in [38]. Nodes are grouped together into clusters that can be overlapping or disjoint. Using dynamic route discovery, sources will cache their paths to destinations for future use. The protocol also takes into consideration the existence of unidirectional links and incorporates them into intra-cluster and inter-cluster routing. Quite a few more cluster-based approaches have appeared in the past. See, for example, [60, 29, 46].

Optimized Link State Routing Protocol (OLSRP) [35] is based on the link state technique. The protocol is proactive, therefore nodes exchange topological information among each other periodically. Using this information, a routing table is calculated so that destinations may be reached by a source. The protocol is developed to work with the IMEP protocol [14] and is suitable for large dense networks.

1.2 Power Management

Unlike stationary hosts, a mobile host can operate only as long as the longevity of its battery. In critical environments such as rescue operations the conserving of battery power is vital in keeping an ad hoc network operational. Note that even if a host does not communicate on its own, it must still route data and routing packets for others. Much work has gone into reducing the power consumption in non-communication devices such as disks [22] [34] and CPUs [47] [68] in mobile systems.

However, relatively less work has gone into reducing power consumption on the communication components of a mobile host within a wireless environment. When addressing the communication components, the main objective is to turn off the component when it is idle to conserve power. However, reducing the power of a mobile host by turning off its communication component also brings forth the unwarranted loss of communication or delay, as it is not usually clear when the component should turn back on.

In [67], the authors simulated trace-driven simulations using the network simulator *ns* [24], for protocols TCP and UDP on reading email and web access by hand-held devices. They then suspended the network interface (wireless Ethernet Card) periodically and showed by increasing the *sleep* duration, an increase in power savings was attainable.

In [45], the authors propose a *transport* level protocol to suspend the communication device during idle periods. Using simulation-based communication patterns and trace-driven communication patterns they showed a reduction of power usage during these idle periods. The power savings varied from 48% to 83%

on the *communication* savings which translated up to 9% *total* savings for high end laptops and up to 40% *total* savings for hand held PCs.

In [62], they propose an elaborate *MAC* level protocol to turn off the power of a mobile host during unnecessary power usage. Their protocol (PAMAS) is based on the MACA protocol [40] coupled with a separate *signaling* channel. Their results showed power savings from 10% to 70% depending on the network topology. Unlike the previous work [67] [45], the PAMAS protocol was simulated under a *multi* hop wireless network environment.

1.3 Concluding Remarks

An ad hoc network is characterized by dynamically changing network topology and the existence of multi-hop wireless links between communicating hosts. In this chapter we reviewed some network layer issues in such networks, particularly, unicast routing protocols (both proactive and reactive) and power management. Our later work for the dissertation evolves around these areas. Efficient packet routing is, of course, the most critical issue because of the dynamic nature of ad hoc networks. In the following chapter we will present a comparative performance evaluation of a suite of protocols for mobile ad hoc networks. The suite contains both proactive and reactive (on-demand) protocols. In Chapter 3 we will focus on a technique to reduce routing overheads for on-demand protocols. The technique works by controlling the extent of route discovery flooding. Chapter 4 will also focus on routing overhead reduction for on-demand protocols, but will emphasize reduction of frequency of such floods via multipath routing. Chapter 5 will concentrate on power-aware on-demand routing that attempts to extend the operational life of a battery operated ad hoc network. Conclusions will be presented in Chapter 6.

Chapter 2

Performance Evaluation of Unicast Routing Protocols

Even though many unicast routing protocols have been proposed for ad hoc networks, their comparative performance is not well understood. Current literature reports only a limited amount of performance study and when performance is reported, typically comparisons have been made only to a selected few protocols (typically only to link-state and distance-vector protocols). Specifically, before our work reported here the unicast routing protocols had not been evaluated against one another in the same framework. Concurrently with our work that we presented in [17] or later, several independent comparative evaluations have been reported. They will be reviewed in a later section.

We will present a comparative simulation study of a suite of unicast routing protocols with different characteristics. The protocol suite includes all protocols reviewed in the previous chapter, *viz.*, proactive protocols such as SPF (the link state protocol), EXBF (extended Bellman-Ford) and DSDV; multipath protocol TORA; and on-demand protocols such as DSR and AODV. We start out by describing the simulation framework and then we will proceed to the performance results and their interpretation.

2.1 Simulation Model

A discrete event, packet-level, routing simulator called *MaRS* (Maryland Routing Simulator) [3] was used for comparative performance evaluation. MaRS is a flexible platform developed specifically for evaluation and comparison of network routing algorithms. MaRS was used previously for comparative evaluation of link-state and distance-vector routing protocols for the NSFNET T1 backbone network with the possibility of link failures [61]. We augmented MaRS to provide node mobility. The nodes can move around in a rectangular region according to a given mobility model (to be described momentarily). Each node has a fixed radio range and has a link to every other node in the system. If the other node is not within range the link cost is infinity. Otherwise, the link cost is modeled by the *hop-normalized delay* function, same as the revised ARPAnet cost metric [43, 61].

Each node is modeled by a store-and-forward queueing station, and is characterized by parameters such as buffer space and processing speed. Each link is characterized by a bandwidth and propagation delay. A link is modeled as an FCFS queue with service time as the transmission time. Currently, our study is limited to network layer details. Thus, no link layer details, such as MAC protocol, multiple-access interference or link errors are modeled, nor are any physical, radio channel level details.

The routing protocol is modeled as an independent routing module, one at each node, which maintains routing information (such as next-hops, distances, routing table etc. depending on the protocol used) and responds to routing packets and link status changes. Routing packets are distinct from data packets in the simulator and are used for route maintenance. The nodes forward data packets via the next hop link as per the routing information provided by the routing module. If the next hop link is broken or there is no next hop information available, data packets are dropped until some usable next hop information is available. In source routed protocols, however, the data packets themselves contain the route gleaned from the route cache maintained by the routing module.

Workload is defined in terms of *conversations.* A conversation is a unicast conversation between a source and a sink. The source and sink are modules associated with nodes. Several workload models are provided in MaRS. In this dissertation, however, we use the simplest model, similar to a datagram, where the source generates data packets destined for the sink at a steady rate. This traffic is characterized by a packet length and a random (exponentially distributed in our simulation) inter-packet generation interval. There is no flow or congestion control.

2.1.1 Detecting link status changes

An important feature of mobile networks is detection of link failures or appearances. This can be done in a few ways, such as periodic link status sensing/probing by so-called *hello* messages. Link layer protocols that use acknowledgments can also be used to detect link failures. Since no link layer details are modeled, a link layer event is generated automatically whenever a link fails or reappears, i.e., a node goes out or comes in range. The routing protocol responds to this event. No hello messages have been modeled in our implementations.

2.1.2 Simulation Parameters

Physical network We assume a channel bandwidth of 1.5 Mbits/sec. This data rate is similar to what is obtainable from the current generation wireless LAN products using IEEE 802.11 [19] or similar standard. Since no multiple-access contention or interference is modeled, each link essentially enjoys the entire channel bandwidth while transmitting packets. In the simulation model, a packet can be unicast (received only by a specific neighbor) or broadcast (received by all neighbors). Broadcast transmissions are modeled as a sequence of unicast transmissions on all active links of a node, but a routing packet is counted only once in simulation statistics. Data packets are always unicast. [1] Routing packets can be broadcast or unicast depending on the protocol requirement.

All nodes are assumed to have adequate buffer capacity for buffering packets awaiting forwarding. Data packets are processed (includes parsing the header, consulting the routing table or cache and adding the packet to the appropriate outgoing packet queue) in parallel. Data packet processing times are fixed. Routing packets have higher priority over data packets in the node's outgoing packet queue. Routing packets are processed sequentially. Routing packet processing time and routing packet sizes depend on the routing protocol being used. Data packet sizes are defined by the workload model plus a fixed, small header. However, for source routing the header length is variable and can be long depending on the length of the

[1] Passive eavesdropping may improve performance of some routing protocols such as DSR. This could be modeled using broadcast transmission. However, we did not model eavesdropping yet.

route. Packet processing times are estimated via independent simulation runs by timing the processing codes in the simulator itself.

Mobility Nodes move around in a rectangular region of size 1000 m $\times$ 1000 m according to a mobility model. The nodes have a constant radio range of 350 m. Nodes are constantly moving, thus putting stress on the routing protocols. The node movements, however, are discretized for ease of modeling in a discrete event framework. Each node chooses a direction, speed and distance of move 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.

For the experiments described in this chapter, the speed of each move is uniformly distributed between a given range (0.4 – 0.6 m/sec for low mobility experiments and 3.5 – 4.5 m/sec for high mobility experiments), distance is exponentially distributed with a mean of 5 m, and the direction is uniformly distributed within $[-22.5°, +22.5°]$ with respect to the direction of the previous move. 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 protocols. All simulations are run for 10,000 simulated seconds.

Workload A simple 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. Flow or congestion control mechanisms will be influenced by the routing dynamics and thus will change the load on the network. It is not clear how this will influence our performance metrics and how comparison should be made across routing protocols with very different dynamics. Thus the simplest datagram workload model has been chosen. 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 no. of conversations per node in the network.

2.2 Performance Results

We have simulated 30 and 60 node mobile, ad hoc networks with respect to the above mobility and workload models. All protocols are studied with respect to three key performance metrics:

- *Fraction of packets delivered:* measured as a ratio of the number of data packets delivered to the destination and the number of data packets sent by the sender. Data packets may be dropped *en route* for two reasons: the next hop link is broken when the data packet is ready to be transmitted, or no routing table (cache) entry exists for the intended destination.

- *End-to-end delay:* measured in ms (milliseconds). This delay includes processing and queueing delays in each intermediate node.

- *Routing load:* measured in a normalized fashion in terms of number of bytes of routing packets transmitted per byte of data packets transmitted. The latter includes only the data packets finally delivered at the destination and not the ones that are dropped. The transmission on each hop is counted once for both routing and data packets. This gives an idea of network bandwidth consumed by routing packets with respect to "useful" data packets.

The first set of figures present the fraction of data packets delivered for all protocols for low and high mobilities for both network sizes (Figures 2.1). Note the excellent behavior on the part of all link-state and distance-vector protocols, but considerably lower packet delivery fraction for on-demand protocols as well as for TORA. On-demand protocols (DSR and AODV) drop a considerable number of data packets during the route discovery phase, as route acquisition takes time proportional to the distance between the source and destination. The situation is similar with TORA. Packet drops are fewer with proactive protocols as alternate routing table entries can always be assigned in response to link failures. In SPF, an alternate route is assigned from the current node's view of the state of all links in the network. In EXBF and DSDV, an alternate minimum cost route is found via a different neighbor. However, no such alternative is available for DSR and AODV and thus packets are dropped until the route can be repaired. TORA, surprisingly, offers the lowest packet delivery fraction in spite of its multipath capability. In our observation, the key reason for this is that the initial route discoveries take longer. This affects the performance most when there is a reconnection after a network partition. In addition, in TORA wireless links have a sense of direction, which is maintained by the protocol. Because of the asynchrony in the distributed implementation, there can be short-lived inconsistencies about the sense of the direction of a link as perceived by the nodes at the end-points of this link (e.g., during link reversal). If the network is congested and the queueing delays are high, this inconsistency can persist for a while causing both delay and loss of packets. This more than offsets the advantages gained by the multipath nature of the protocol.

We also note here in passing that TORA can be quite sensitive to the loss of routing packets compared to the other protocols. Loss of certain types of routing packets (e.g., UPD, using the terminology in [54]) can put the routing tables towards an inconsistent state, which may take a while to recover. In the most recent specification of TORA [53] it is recommended that TORA be run with an encapsulation protocol called IMEP [14] that guarantees reliable, in-order delivery of routing packets. We did not, however, feel it necessary to implement IMEP in our simulation model as loss of routing packets is unlikely in the absence of any link layer model when the link is up.

Buffering data packets while a route discovery is in progress has a great potential to improve DSR, AODV and TORA performances. However, this alternative has not been evaluated up to this point. We also have not used early quenching of route request packets by a non-destination node in AODV. We have noticed that AODV performs very poorly by picking up stale routes, if early quenching is used. It affects both its packet delivery and delay performance significantly. DSR, on the other hand, does not demonstrate any significant performance differential with or without the use of early quenching.

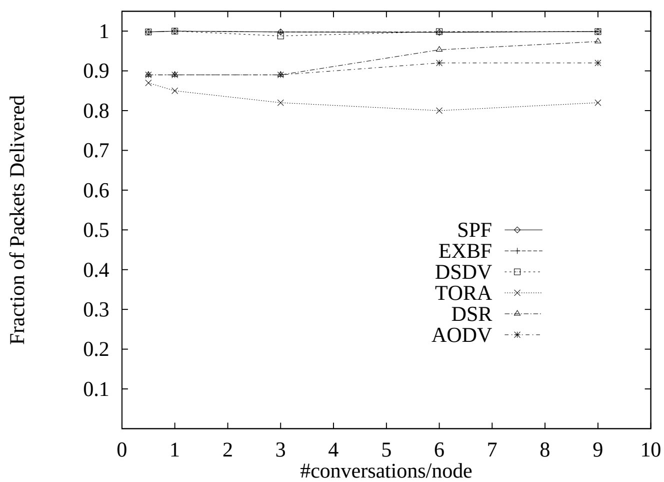
AODV has a slightly worse packet delivery performance than DSR because of higher drop rates. AODV uses route expiry, dropping some packets when a route expires and a new route must be found. This, however, gives better quality routes to AODV in general.

The average end-to-end delays are shown in Figure 2.2, for both network sizes and for low and high mobilities. The shortest path protocols (SPF, EXBF and DSDV) show the minimum delay characteristics. AODV and DSR show worse characteristics as their routes are typically not the shortest. Even if the initial route discovery phase finds the shortest route (it typically will), the route may not remain the shortest over a period of time due to node mobility. Also, note that in AODV routes are maintained as a soft state, i.e., routes expire after a timeout interval and a fresh route discovery is initiated. Accordingly, AODV performs a little better delay-wise and can possibly do even better with some fine-tuning of this timeout period by making it, for example, a function of node mobility. TORA has the worst delay characteristics because of the loss of distance information with progress.

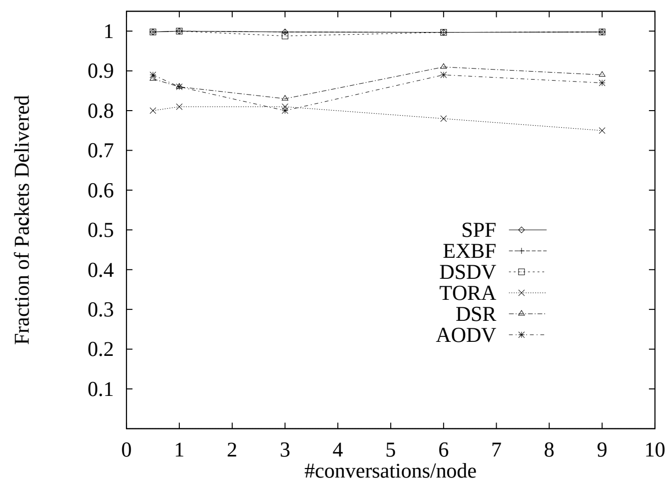
The routing load characteristics shown in Figure 2.3 are interesting. Note that the routing load varies over a very wide range and hence the plots use a logarithmic scale for the vertical axis. SPF expends significantly more routing load than the other protocols. The distance vector protocols, EXBF and DSDV, have very similar routing loads, and much lower than SPF. DSR and AODV perform very well, particularly for smaller number of conversations, with DSR often outperforming AODV. The excellent routing load performance of DSR is due to the optimizations possible by virtue of source routing. TORA's performance is not very competitive with the distance vector and on-demand protocols. We conjecture that it is due to the fact network partitions cause TORA to do substantial work to erase routes even when those routes are not in use.

It appears that the theoretical "worst" case communication complexity (number of messages required to adapt to a link failure/recovery) does not provide much insight into the average case behavior obtained via simulation. For example, SPF has a worst case communication complexity of $O(2e)$ [42], where e is the number of links in the network. On the other hand communication complexity in protocols based on distributed Bellman-Ford, such as EXBF and DSDV, is exponential in the number of nodes N in the network [11], which should be much higher. However, SPF has a much higher routing load in the simulations.

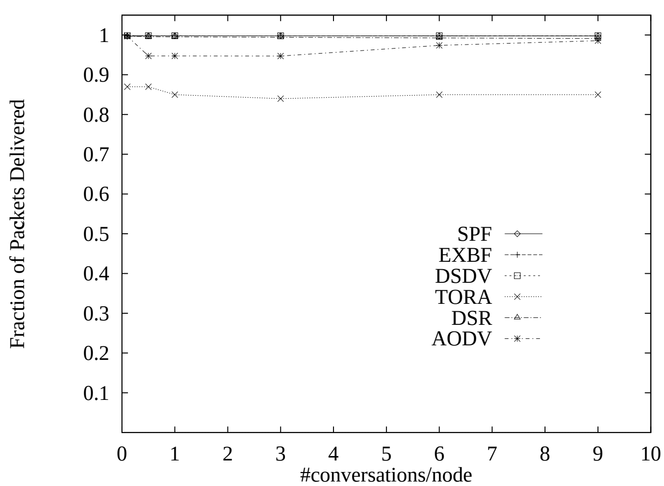
Most protocols benefit to some degree as the number of conversations grow large. This is because a single route repair can potentially benefit many conversations. Thus routing load do not increase as much as the data load with increasing number of conversations. This effect is the most pronounced for proactive, shortest-path protocols where routing load is independent of the data load. Also note that the routing load differentials between the protocols become smaller as the number of conversations grow large. Thus the proactive, distance vector protocols may be favored at large number of conversations as they provide better packet delivery fractions and end-to-end delay characteristics. We also note in passing, that DSR uses



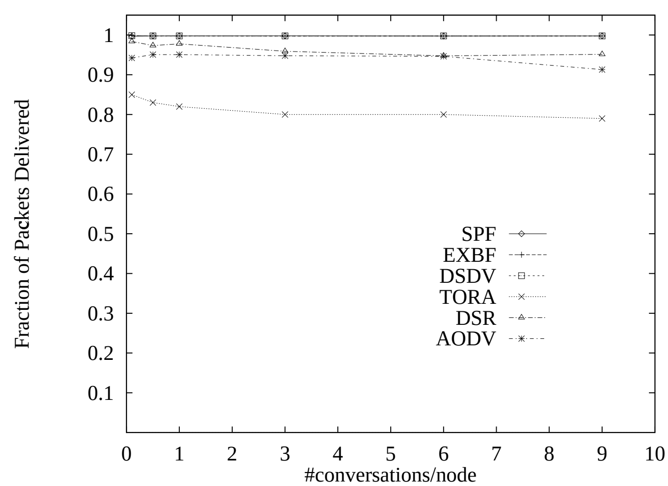
(a) Packet delivery fraction (30 hosts), low mobility case



(b) Packet delivery fraction (30 hosts), high mobility case

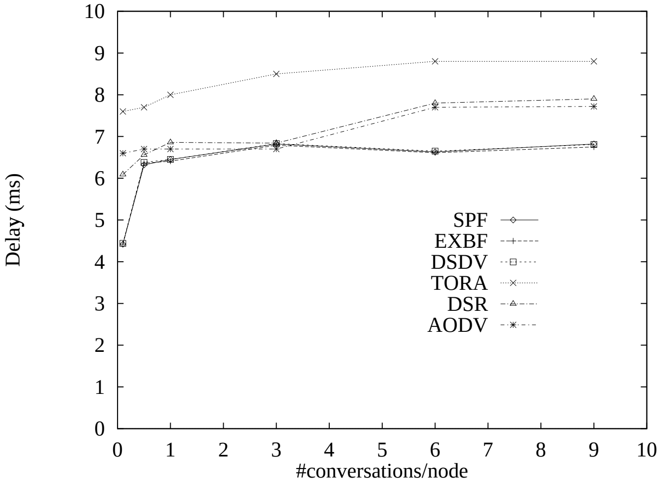


(c) Packet delivery fraction (60 hosts), low mobility case

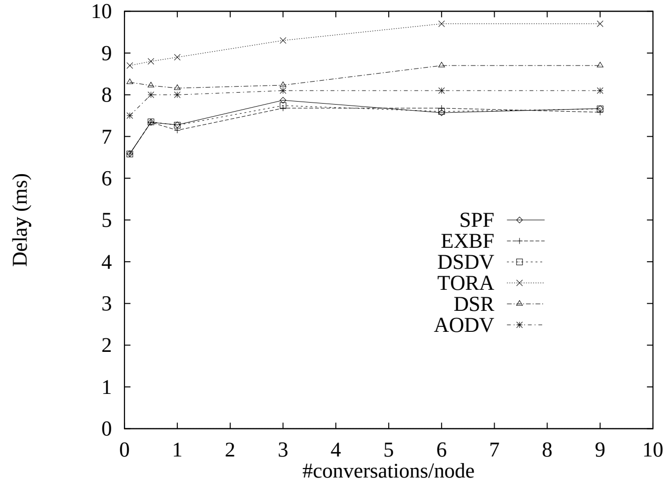


(d) Packet delivery fraction (60 hosts), high mobility case

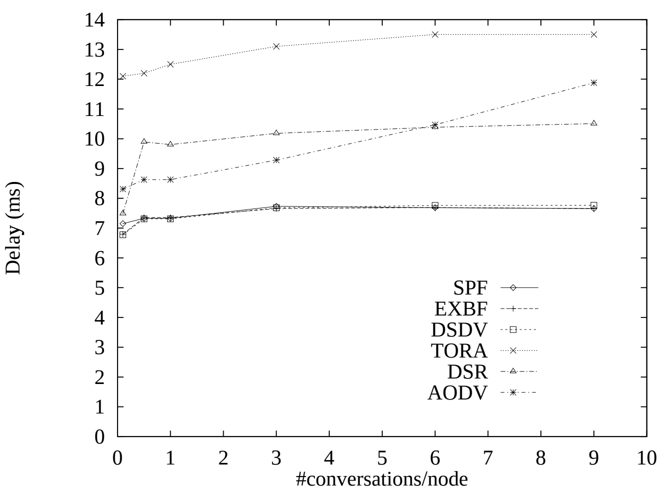
Figure 2.1: Fraction of packets delivered in a **30** and **60** mobile host networks for all routing protocols.



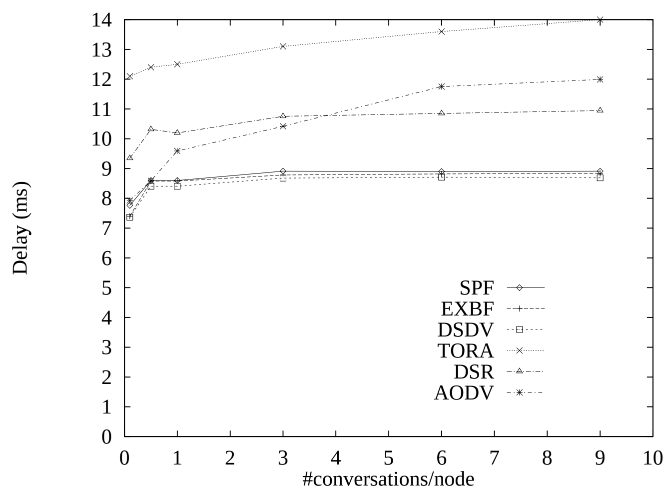
(a) Average delay (30 hosts), low mobility case



(b) Average delay (30 hosts), high mobility case

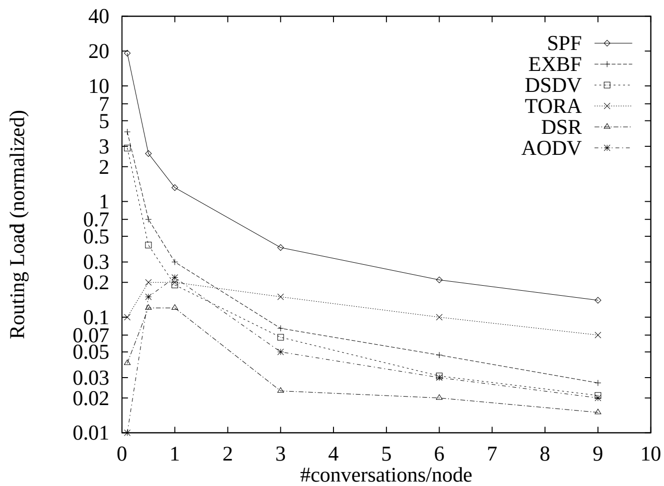


(c) Average delay (60 hosts), low mobility case

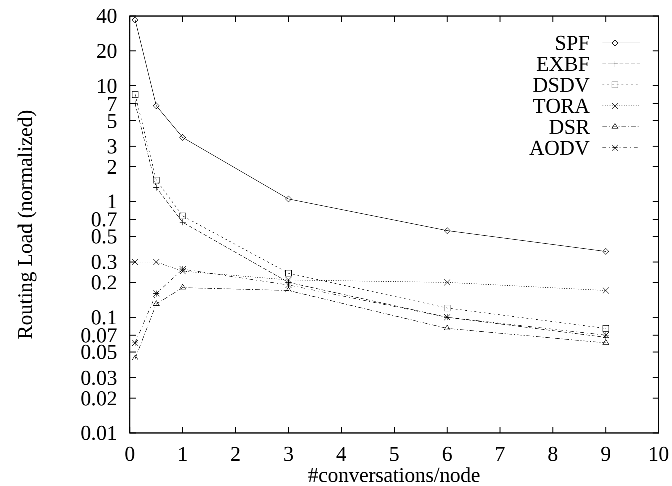


(d) Average delay (60 hosts), high mobility case

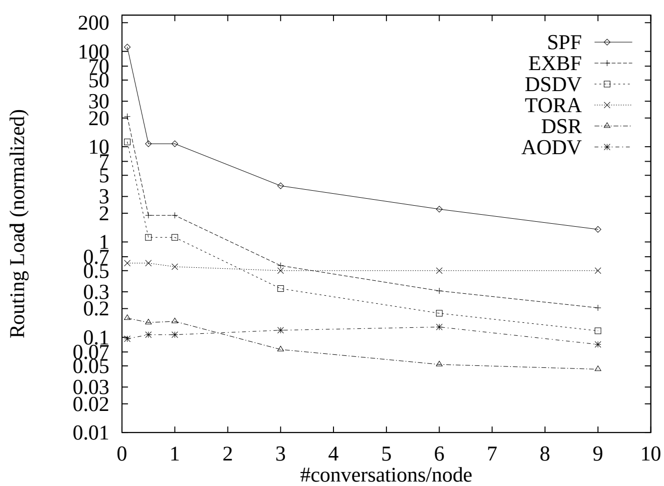
Figure 2.2: Average end-to-end delay in a **30** and **60** mobile host networks for all routing protocols.



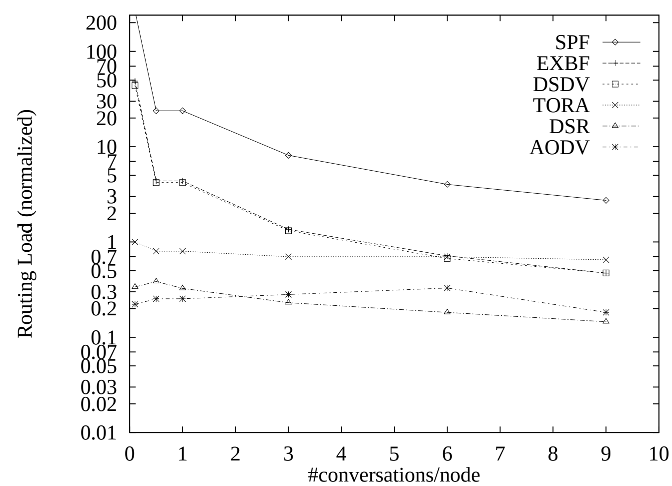
(a) Normalized routing load (30 hosts), low mobility case



(b) Normalized routing load (30 hosts), high mobility case



(c) Normalized routing load (60 hosts), low mobility case



(d) Normalized routing load (60 hosts), high mobility case

Figure 2.3: Normalized routing load in a **30** and **60** mobile host networks for all routing protocols.

somewhat more bandwidth (10–20% in our experiments) because of source routing that increases the size of the header in data packets. Even counting this in as a part of the routing load, DSR is very competitive with AODV. However, this bandwidth usage is expected to increase for larger networks and may make DSR less attractive.

2.3 Related Work

Some simulation studies of the routing protocols evaluated here have been presented earlier in literature. In [52] Park and Corson compared TORA with an ideal link-state routing protocol and demonstrated superior performance of TORA. In their chosen network model, however, there is no true node mobility. The network is connected in a “honeycomb” pattern and links go up and down at some rate with the average network connectivity held constant artificially. In an earlier work [13], Corson and Ephremides presented simulation results of three protocols on which TORA is based in part. Here again stationary networks are chosen with links going up and down at random intervals.

Johnson and Maltz presented the simulation study of DSR in [38]. They simulated a true, mobile network. Their results indicate that DSR is able to find close to optimal routes on an average. However, actual path lengths in hops (and not the end-to-end delay) was measured to determine optimality. Freisleben and Jansen evaluated DSR against DSDV in [26]. They built a more comprehensive simulator complete with a MAC layer model of the IEEE standard 802.11 [19] as well as true mobility. Only packet delivery fraction is evaluated with DSDV performing marginally worse than DSR.

Concurrently with our work, the Monarch research group in CMU did a comprehensive simulation study of four ad hoc routing protocols, *viz.*, DSDV, TORA, DSR and AODV [6]. They used a fairly comprehensive simulator that include physical layer and radio interface models and considered the IEEE 802.11 MAC protocol [19]. From a qualitative point of view, much of their observations regarding the comparative performance of these four protocols are similar to ours. In the following year, a research group from Sweden used the same simulator but with new mobility models [37]. A new mobility metric that measures mobility in terms of relative speeds of the nodes was introduced. Their experiments were conducted on three ad hoc routing protocols, *viz.*, DSDV, AODV, and DSR. Their conclusions differed slightly from the former study in that DSR performed better for low loads whereas AODV performed better high loads. In the Monarch study, DSR was found to have better performance in contrast to the other protocols, including AODV.

There is a large body of simulation study of shortest-path protocols. Most closely related to our work is the performance study in [61] by Shankar et. al., which used both SPF and EXBF protocols, among others, for a performance comparison using the MaRS simulator. They used a static network with dynamically changing link connectivity with links going up and down according to a stochastic distribution. Various traffic and link failure models were studied to evaluate transient and steady-state performance of the routing protocols. Delay and throughput performance of SPF and EXBF were found to be equivalent in the most part.

2.4 Concluding Remarks

Our work is one of the first attempts towards a comprehensive performance evaluation of routing protocols for mobile ad hoc networks. The protocol suite includes several protocols currently being considered in the MANET working group within IETF for standardization. In addition, more established link-state and distance-vector protocols have been included in the suite to provide a reference point. Steady state performance in terms or fraction of packets delivered, delay, and routing load have been considered as the performance metrics.

Even with a packet-level simulation model the essential aspects of the routing protocols are exposed. The key observations are as follows. The proactive, shortest path protocols provide excellent performance in terms of end-to-end delays and packet delivery fraction, however, at the cost of higher routing load. The on-demand protocols suffer from sub-optimal routes as well as worse packet delivery fraction because of more dropped data packets. However, they are significantly more efficient in terms of the routing load. The multipath protocol, TORA, did not perform well in spite of maintaining multiple redundant paths. The overhead of finding and maintaining multiple paths seems to outweigh the benefits. Also, the end-to-end delay performance is poor because of the loss of distance information. The routing load differentials between all routing protocols reduce with large number of peer-to-peer conversations in the network. However, the other performance differentials are not affected conclusively. Rate of mobility and network size do not seem to affect the performance beyond what is normally expected – such as higher routing load, more delay and dropped packets.

It is important to note the limitations of the study. First, a packet-level simulation has its own limitations. No MAC protocol or multiple access interference is modeled. Thus, a high routing packet load does not interfere as much with the data transmissions (except for queueing delays) as it would in reality. Also, there are no transmission errors. The current study best reflects the performance when all active links in the network are on a separate frequency band. Second, only a moderate size network has been studied. Though it is unclear what sizes will be realistic for an ad hoc network running IP based protocols, using a few other sizes, going up to a few hundreds, will provide more maturity to the study. Third, a few different traffic models, for example, dynamically changing peers for conversations and introduction of hot-spots should be studied to evaluate the sensitivity to traffic models. Fourth, fine tuning of certain protocol parameters (e.g., various timeout periods for the on-demand protocols) is possible with changing mobility and traffic characteristics. We have used reasonable values that work well, but have not changed the values for different traffic and mobility. Also, certain protocol specific optimizations (e.g., passive eavesdropping in DSR) as well as more general optimizations (e.g., buffering of data packets on route loss until route is repaired) are possible. They may impact relative performance. Fifth, impact of memory usage by the protocols have been ignored. This may be important as the computing nodes deployed in a mobile, ad hoc environment can be low power and small size devices. In spite of these limitations, we have gained valuable insight into the behavior of routing algorithms in an ad hoc network. Some of our future work will address these limitations.

2.5 Acknowledgement

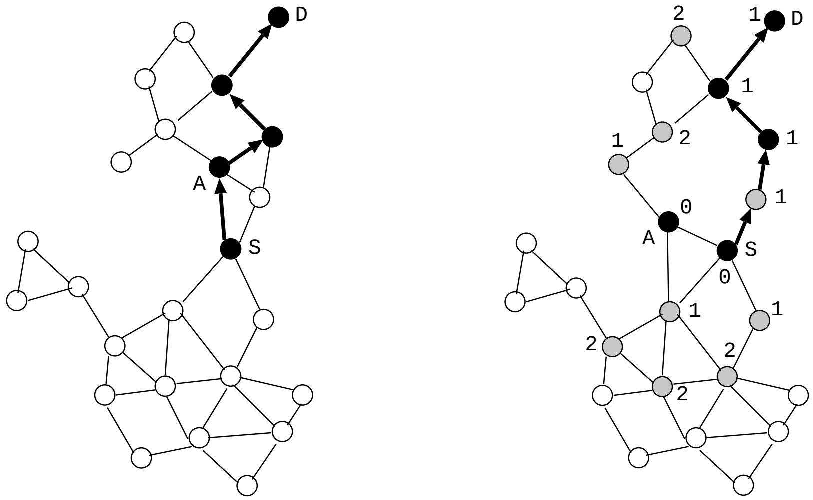
Jiangtao Yan implemented the DSDV and AODV protocols, and helped significantly in the performance evaluation by running many simulations.

Chapter 3

Query Localization Techniques

The motivation behind the design of on-demand routing protocols, such as DSR and AODV, 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 it was demonstrated in the simulation studies presented in the previous chapter, the routing overhead still approaches to 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 route to the destination.

Flooding is straightforward to implement. We have described how it is typically implemented in Section 1.1.4 of Chapter 1. However, as mentioned before, network-wide flooding incurs a considerable overhead and diminishes the performance advantage of on-demand protocols. Although several optimizations to 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 [5, 6]), 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 goals 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-



(a) Before route change (b) After route change

Figure 3.1: Illustration of the path locality principle with k=1.

aided routing or LAR technique [44] which uses the Global Positioning System (GPS) [51], 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 advantage.

3.1 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 literature [44], we refer to this region as the *request zone*. Similar ideas have been used in the selective paging schemes in cellular networks [2], where a mobile is paged only in the neighboring cells of the last cell it reported to have visited.

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:* 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 3.1 illustrates the above protocol with an example network. Nodes S and D are the source and destinations, respectively. Node A has moved causing a route change. The "dark" nodes were on the old route. All "filled" (dark or lightly shaded) nodes are flooded with 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:* 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.

3.1.1 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.

3.1.2 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. [1]

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} = False is visited. Otherwise, the counter is not modified. The query is

[1] 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.

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} = False is visited, but is reset to zero, when a node with F_{ij} = True is visited.

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 [57]. One disadvantage of the distributed maintenance is that it may be hard to support alternative definitions of 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.

3.2 Performance Evaluation

We ran the experiments under the same environment as in Chapter 2 except now our mobility speed is uniformly distributed between the ranges of 3.5 – 4.5 m/sec (low mobility) and 14.0 – 18.0 m/sec (high mobility) and the direction is uniformly distributed within $[+45°, -45°]$ with respect to the direction of the previous move. Simulations are run for 300 simulated seconds and the final results presented are the average over 5 runs with different random number streams. Three important performance metrics are evaluated:

- *Routing overhead:* measured in terms of the total number of routing packets transmitted (broadcast transmissions are counted as a single transmission).

- *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.

- *End-to-end delay:* measured as the average end-to-end latency of data packets. This measurement is given in ms (milliseconds).

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.

3.2.1 Simulation Results

In the first set of experimental data (Figure 3.2), query localization protocols for both heuristics (referred henceforth as protocol 1 and 2, respectively) 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.

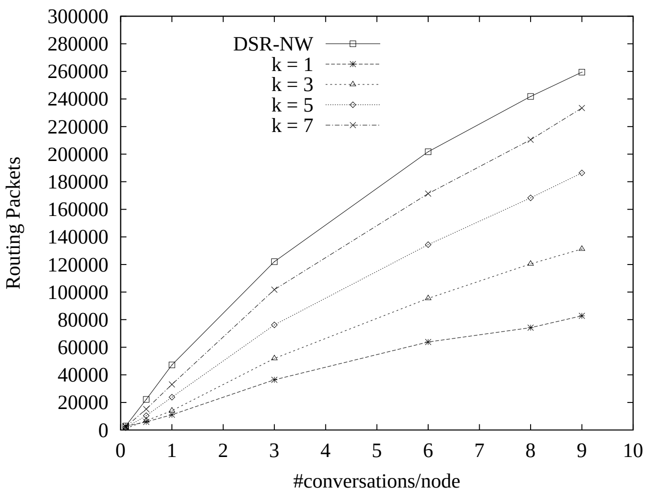
Figure 3.3 shows data packet delivery fraction for both protocols. As expected, lower values of k gives poorer packet delivery performance, especially for higher mobility. Note that larger route discovery latency translates to higher packet loss in our simulation, as data packets are not buffered.

Figure 3.4 shows the average end-to-end delay. An interesting observation here is the very large average delay in the DSR-NW protocol for the high mobility case. This is due to large queueing delays in the nodes owing to the presence of a large number of routing packets. As expected, lower values of k gives a lower delay. Based upon these previous results, protocol 1 was chosen as the protocol with the most potential. Henceforth, all experiments conducted are based upon protocol 1.

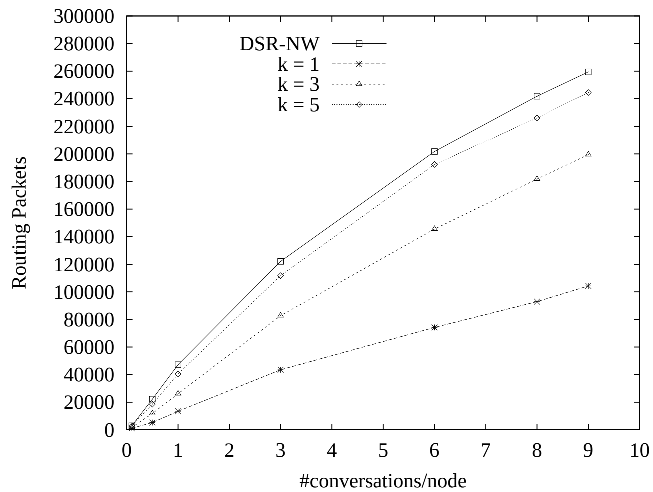
Figure 3.5 shows the performance for dynamic choice of k for both routing overhead and data packet delivery. Here we have a conservative implementation, with k starting from 1 incrementally for each route discovery. For the sake of comparison, we also present the performance of the DSR protocol that uses query containment as described in Subsection 1.1.4 of Chapter 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 (Figure 3.6) with 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 reinforces 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 chooses mobility models such that the probability of network partition is minimal [6]. 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" while keeping the mobility model identical. Now the field is a square area with a hole inside as shown in Figure 3.7. The performance data is presented in Figures 3.8 and 3.9. Average hop counts between source and destination are now larger, between 6–9. Note that even with much larger hop counts the query localization technique performs very well.

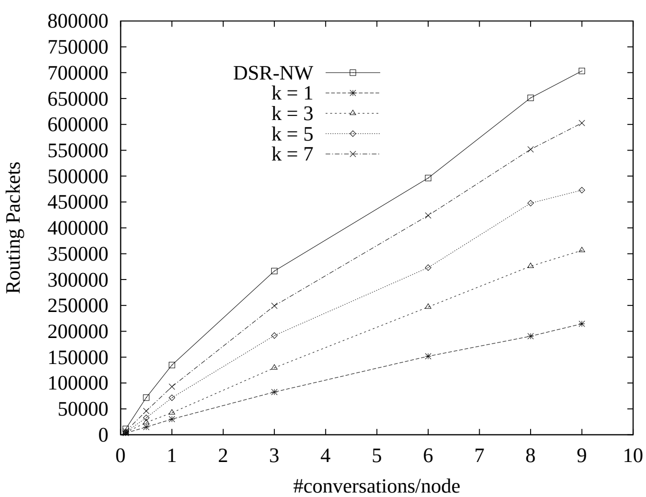
We also ran the same models with the number of conversations/node fixed (1 and 9 conservations/node were used), while the average mobility varied over a predefined range. The average speed of the nodes were varied from 4 m/sec up to 28 m/sec. Figures 3.10 and 3.11 shows the results for 1 conversation/node with varying mobility, and Figures 3.12 and 3.13 shows the results for 9 conversations/node with varying mobility. 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 our localization technique. 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 queuing delays (see Figures 3.13 (a) and (b)). As expected, the performance impact always tends to be greater for higher mobility.



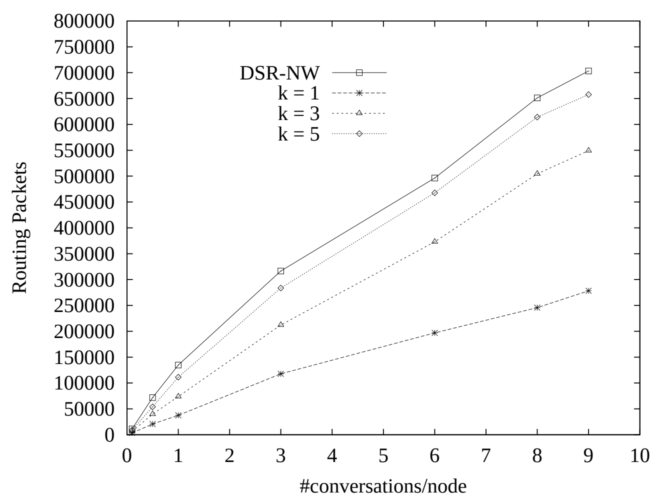
(a) Low mobility, Protocol 1



(b) Low mobility, Protocol 2

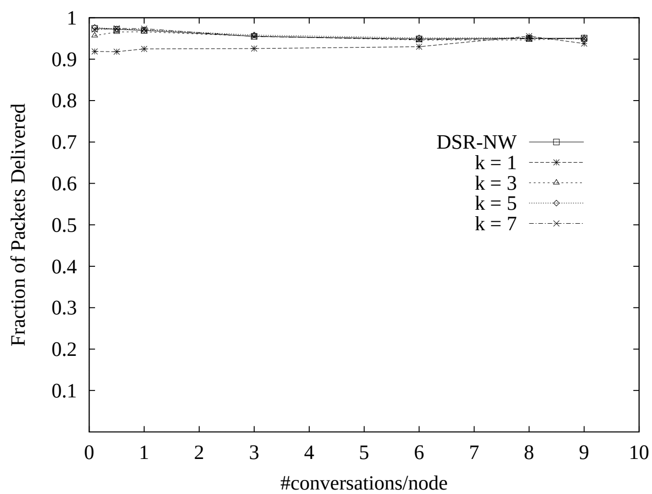


(c) High mobility, Protocol 1

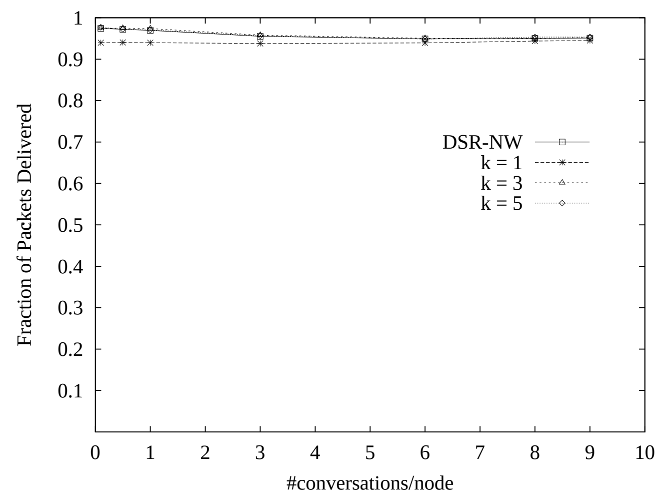


(d) High mobility, Protocol 2

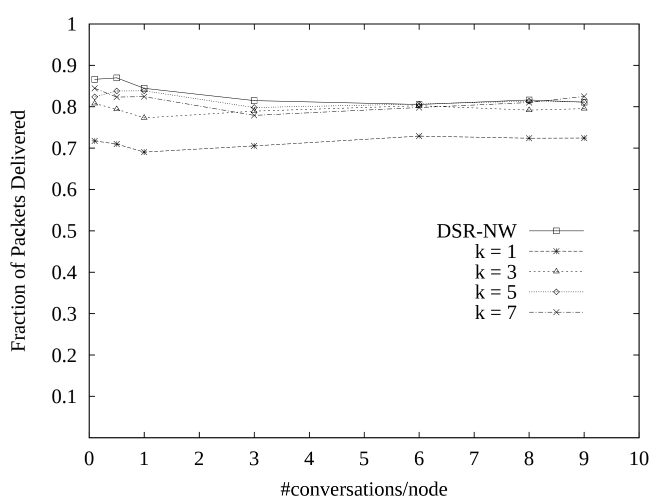
Figure 3.2: Routing overhead for both protocols for different values of *k*.



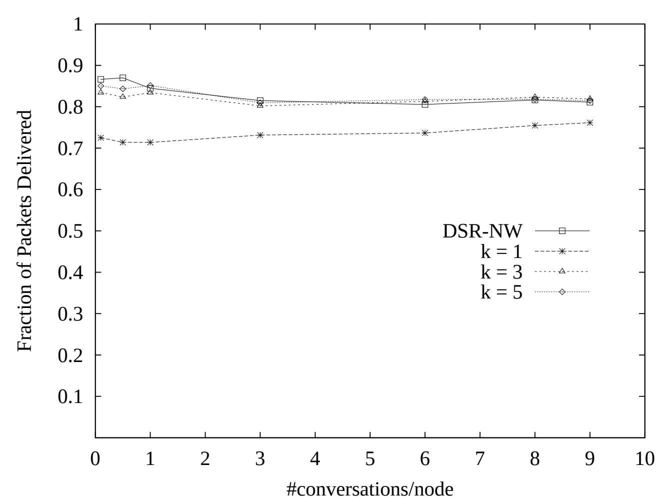
(a) Low mobility, Protocol 1



(b) Low mobility, Protocol 2

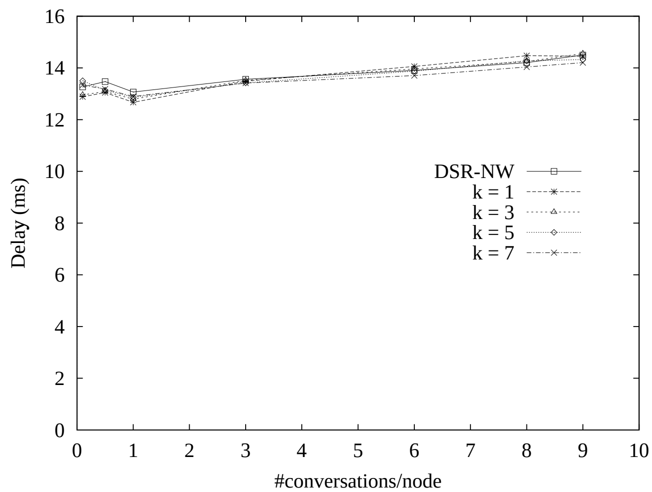


(c) High mobility, Protocol 1

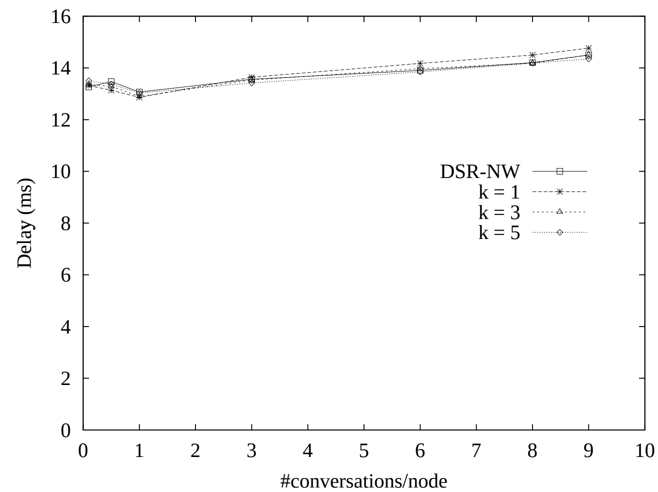


(d) High mobility, Protocol 2

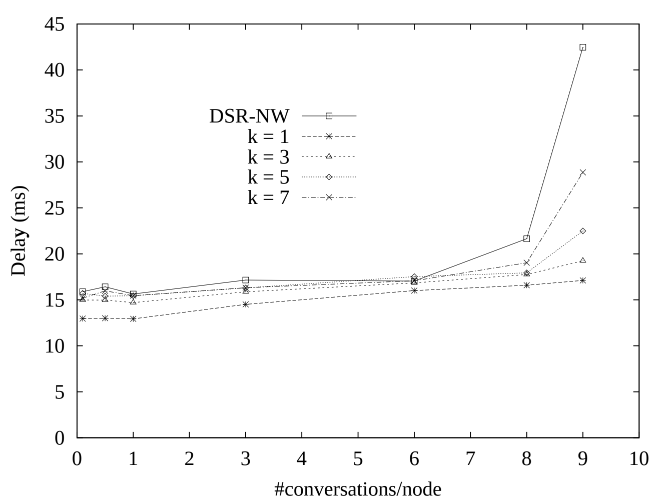
Figure 3.3: Data packet delivery fraction for low and high mobilities for different values of *k*.



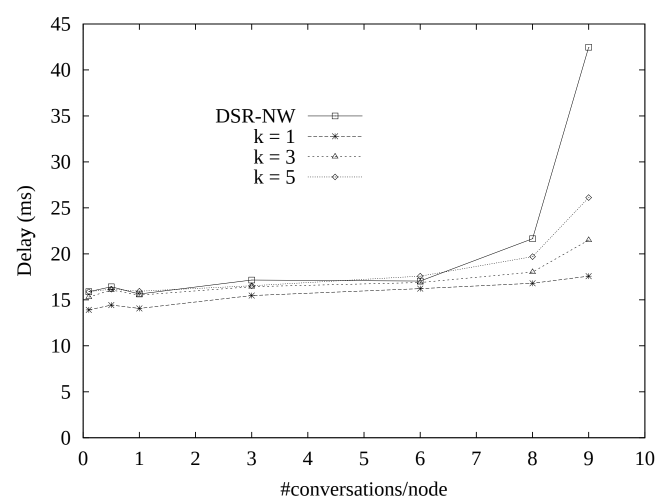
(a) Low mobility, Protocol 1



(b) Low mobility, Protocol 2

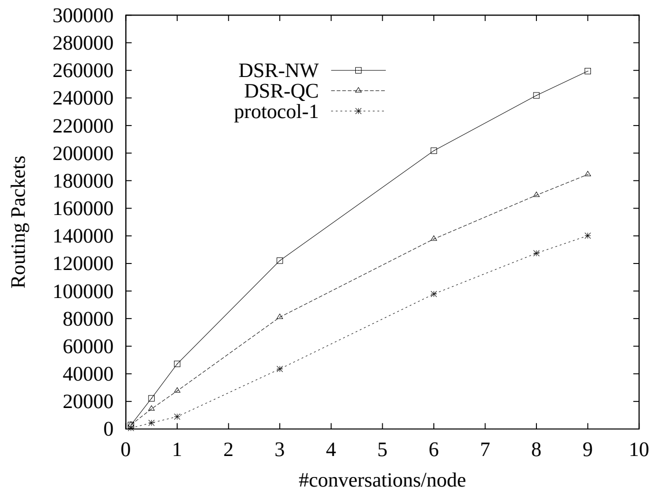


(c) High mobility, Protocol 1

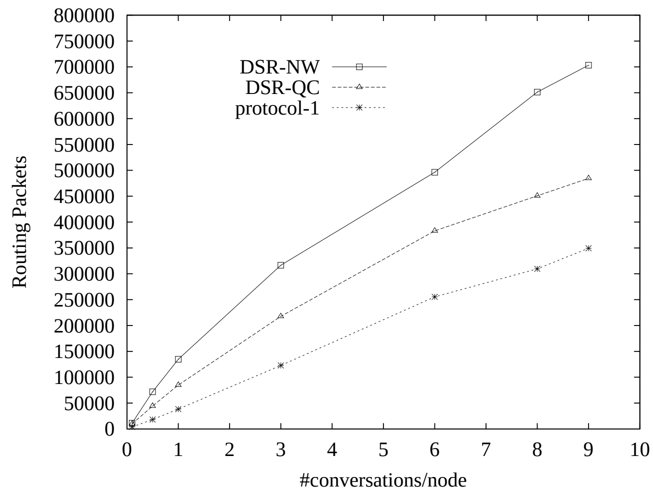


(d) High mobility, Protocol 2

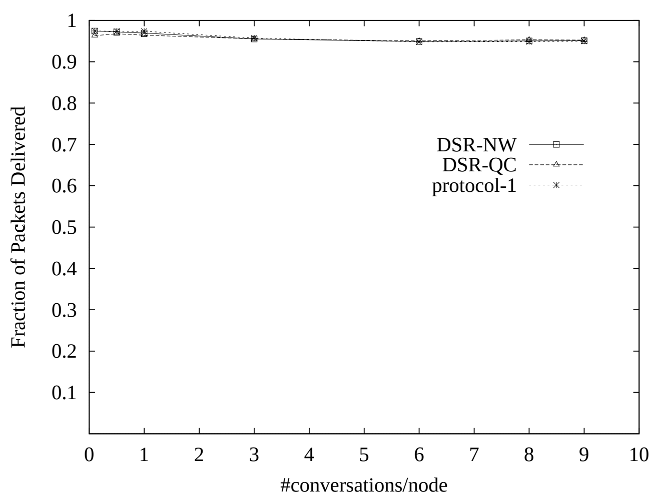
Figure 3.4: Average delay for low and high mobilities for different values of *k*.



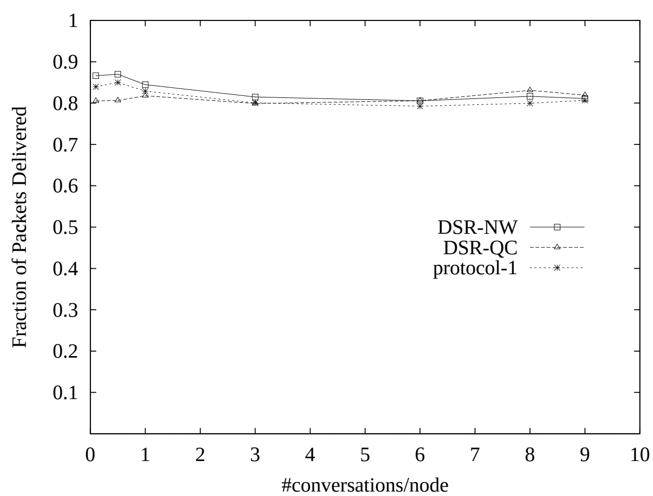
(a) Routing overhead, low mobility case



(b) Routing overhead, high mobility case

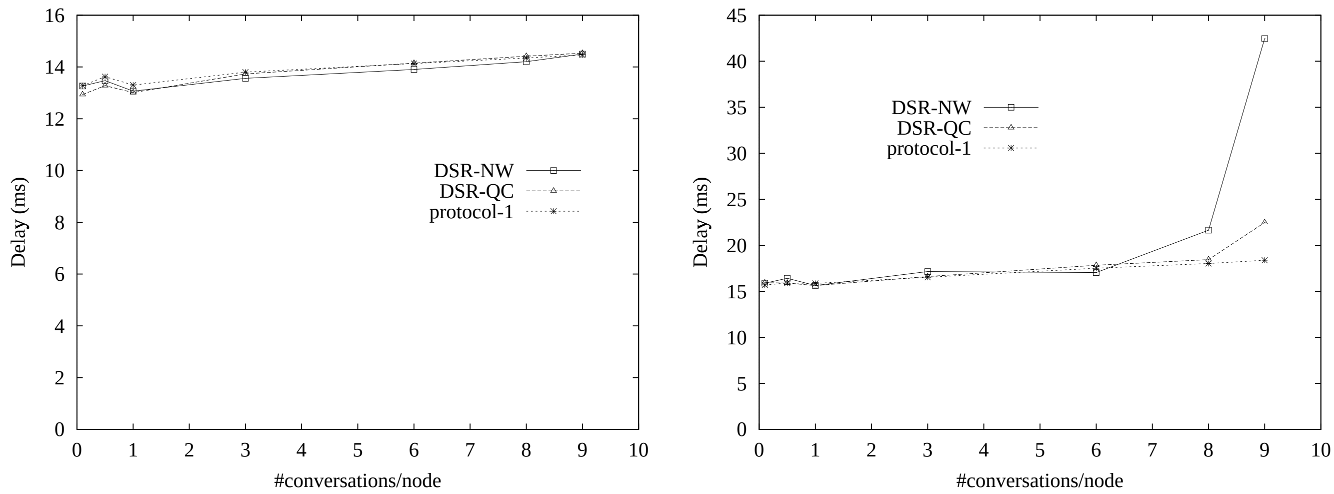


(c) Packet delivery fraction, low mobility case



(d) Packet delivery fraction, high mobility case

Figure 3.5: Routing overhead and data packet delivery metrics for low and high mobilities for dynamic choice of *k*.



(a) Average delay, low mobility case

(b) Average delay, high mobility case

Figure 3.6: Average delay metric for low and high mobilities for dynamic choice of *k*.

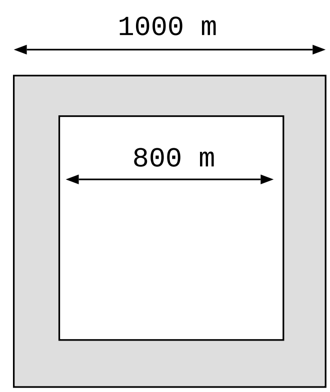
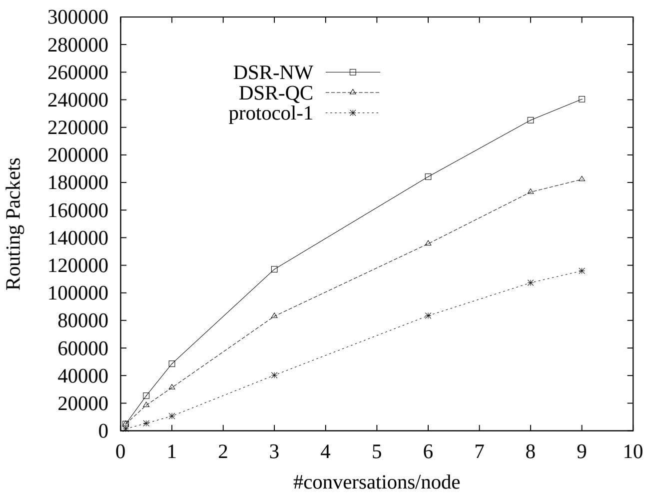


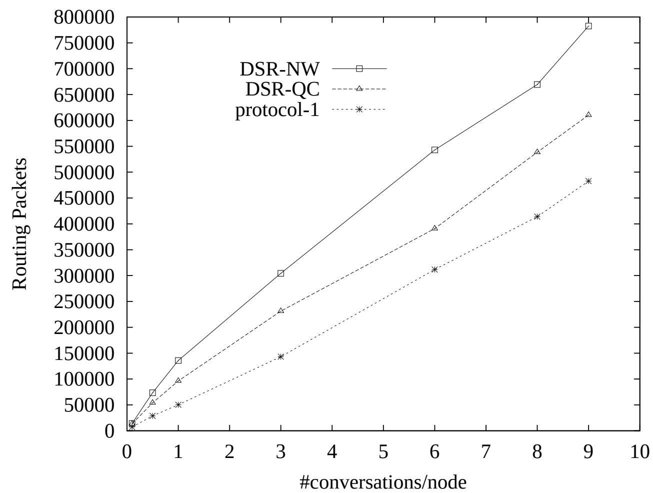
Figure 3.7: Square-with-a-hole region

3.3 Related Work

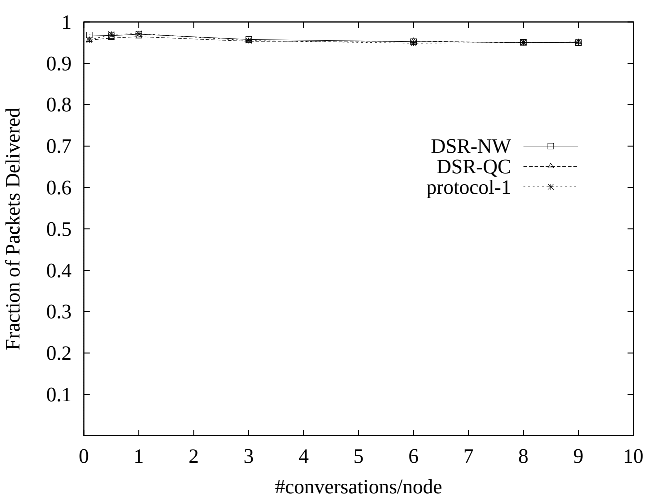
It is not a new observation that network-wide flooding of query messages may be a performance problem in a bandwidth-poor ad hoc, wireless network. A sophisticated technique called Location-Aided Routing (LAR) [44] has been proposed recently which uses the Global Positioning System (GPS) to localize 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



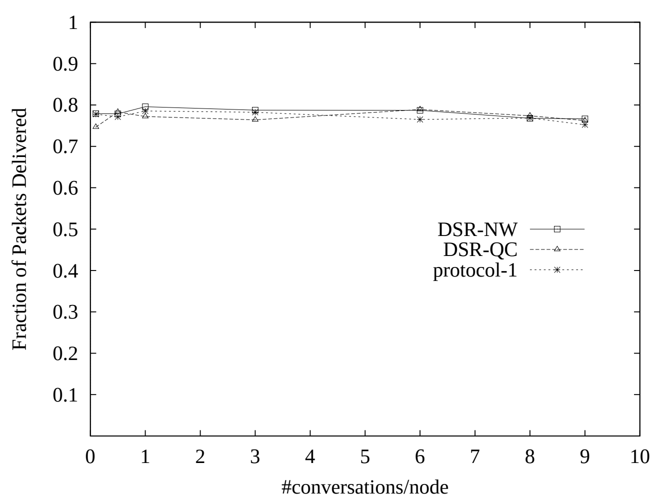
(a) Routing overhead, low mobility case



(b) Routing overhead, high mobility case

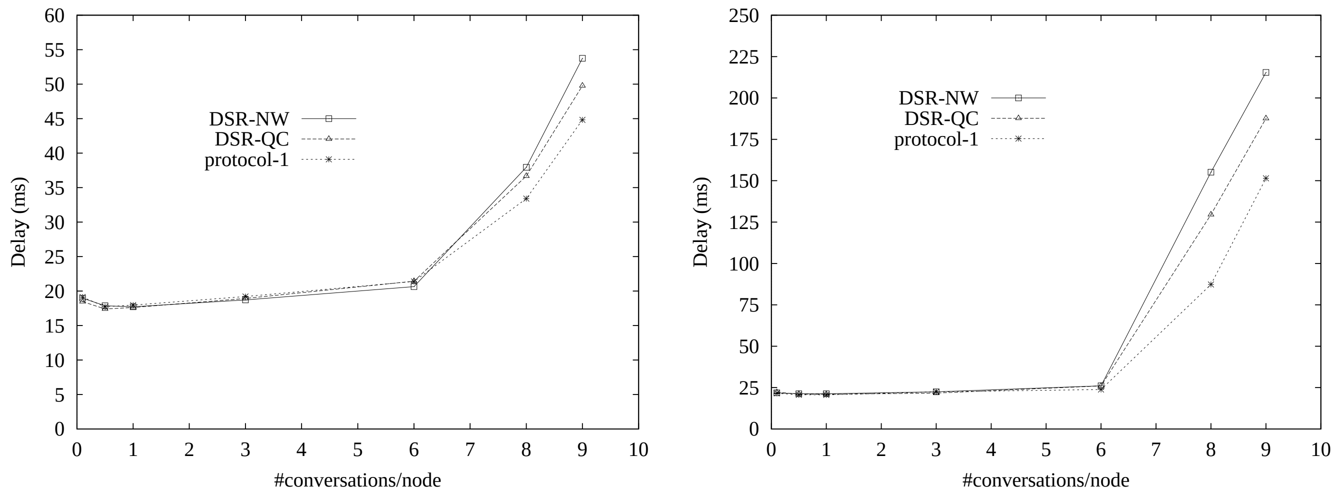


(c) Packet delivery fraction, low mobility case



(d) Packet delivery fraction, high mobility case

Figure 3.8: Routing overhead and data packet delivery metrics for low and high mobilities for dynamic choice of *k* within the square-with-a-hole region.



(a) Average delay, low mobility case

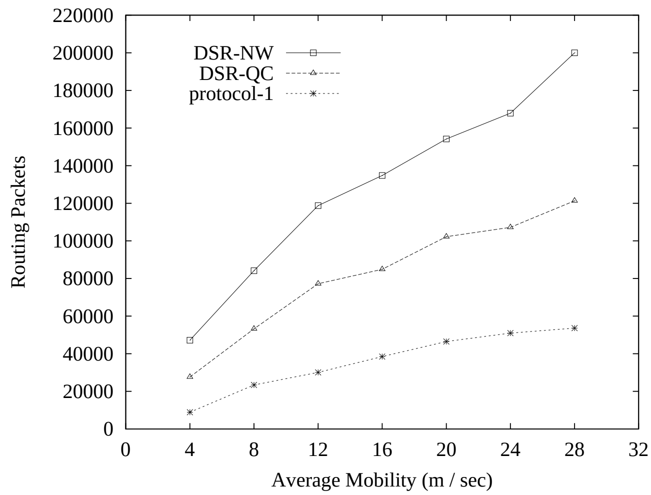
(b) Average delay, high mobility case

Figure 3.9: Average delay metric for low and high mobilities for dynamic choice of k within the square-with-a-hole region.

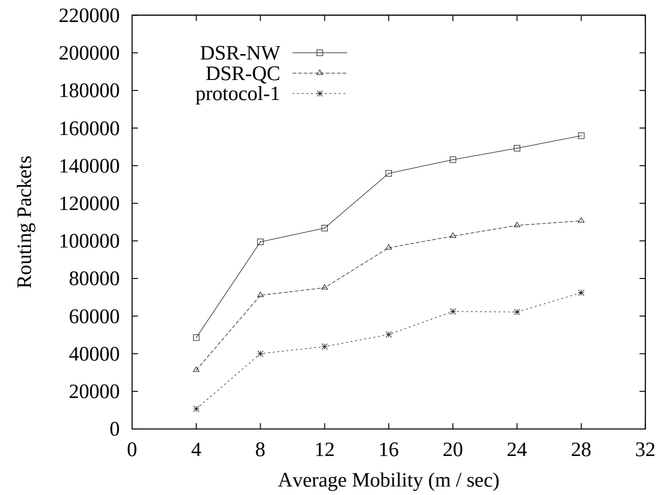
to network-wide flooding. Our approach is similar in principle to LAR and achieves similar improvements with similar mobility models, but it does not require location or speed information. This makes it attractive in many situations where GPS cannot be used because of concerns of (i) cost (GPS may be 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 the Bluetooth [23]) or (iii) usability (GPS cannot be used inside many buildings).

It is also worth noting here that our "location-unaware" locality-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.

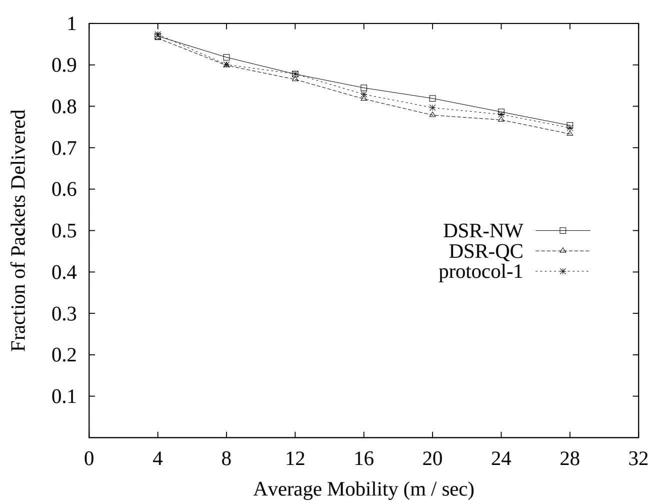
A Zone Routing Protocol (ZRP) [32] has been proposed which responds to the routing overhead vs. route discovery latency tradeoff question in an indirect fashion. It advocates only limited use of on-demand protocols. In ZRP the network is split into *zones* or clusters of suitable size. *Proactive* shortest 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. Sim-



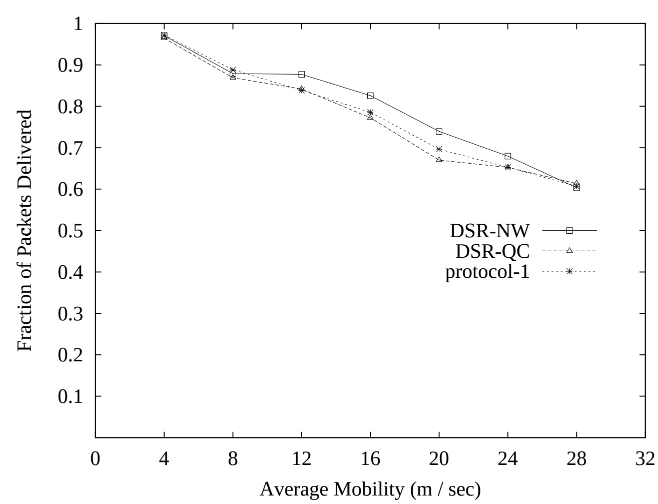
(a) Routing overhead, square region



(b) Routing overhead, square-with-a-hole region

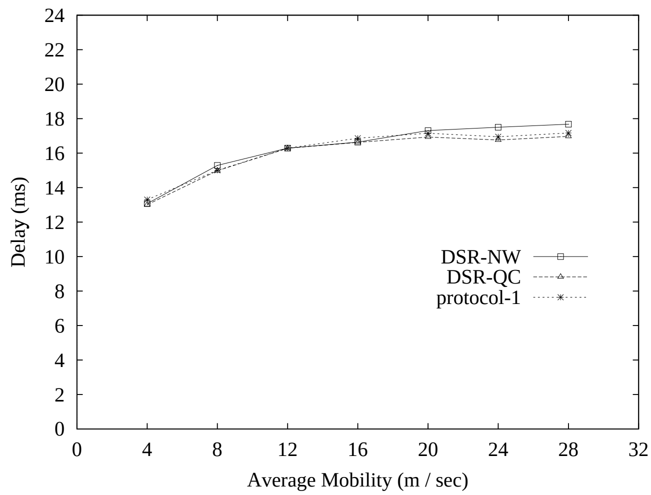


(c) Packet delivery fraction, square region

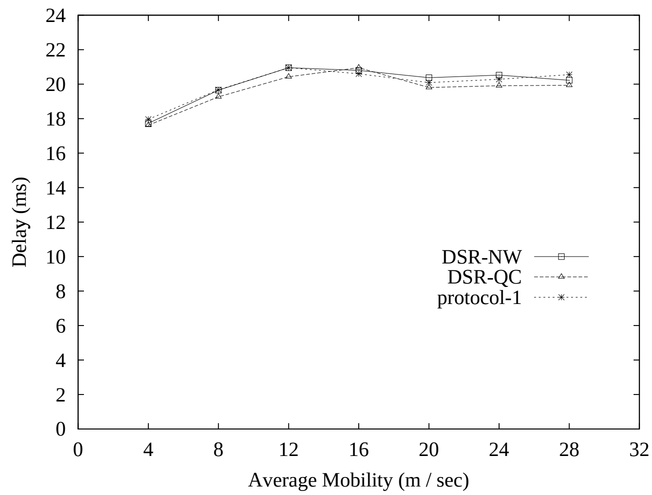


(d) Packet delivery fraction, square-with-a-hole region

Figure 3.10: Routing overhead and data packet delivery metrics for 1 conversation/node with varying mobilities.



(a) Average delay, square region

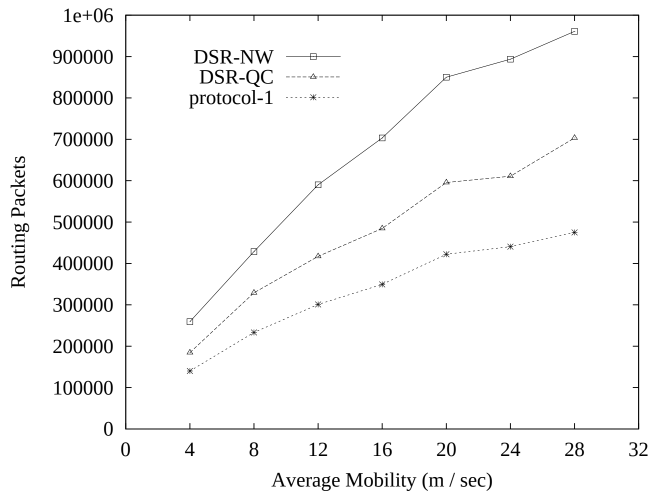


(b) Average delay, square-with-a-hole region

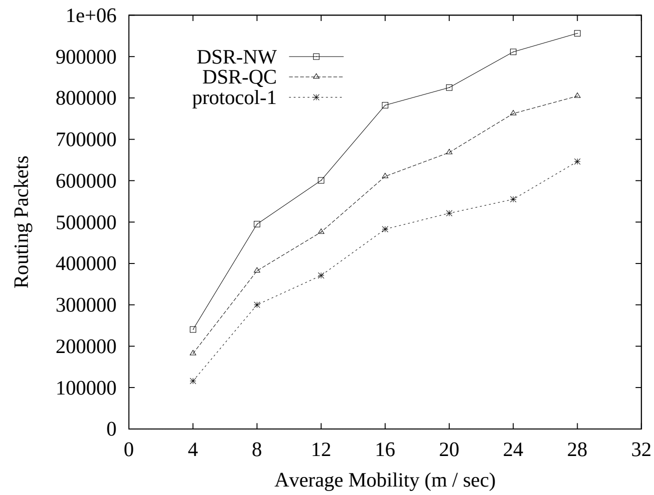
Figure 3.11: Average delay metric for 1 conversation/node with varying mobilities.

ilarly, in the recently proposed CEADR protocol [65], only the so-called “core” nodes participate in route computation, thus limiting the flooding overhead.

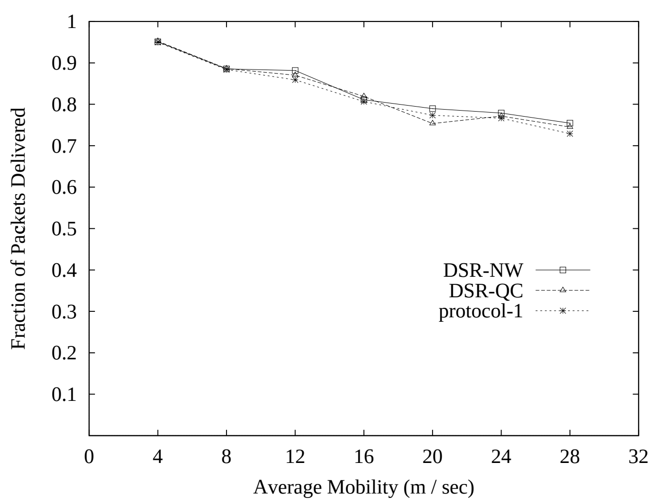
The flood control problem is not unique to ad hoc networks. Other network routing protocols also face this problem. For example, in [8] the construction of multicast trees across different domains in a wide-area wired network is considered. Here, the protocol does consecutive flooding to search increasingly larger regions until reaching the multicast tree. This technique is called the *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. 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 3.1 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, though a recent version of AODV protocol includes this technique [58]. In a recent work the RDMAR protocol [1] automatically estimates the hop-wise distance between the source and destination pair. This estimate is based on the physical distance the nodes may have traveled after the prior route discovery, node velocity and transmission range estimates. The TTL field is set equal to the above estimate.



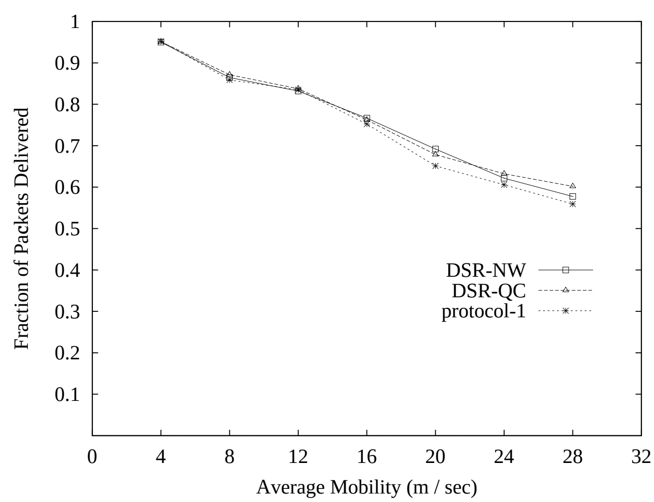
(a) Routing overhead, square region



(b) Routing overhead, square-with-a-hole region

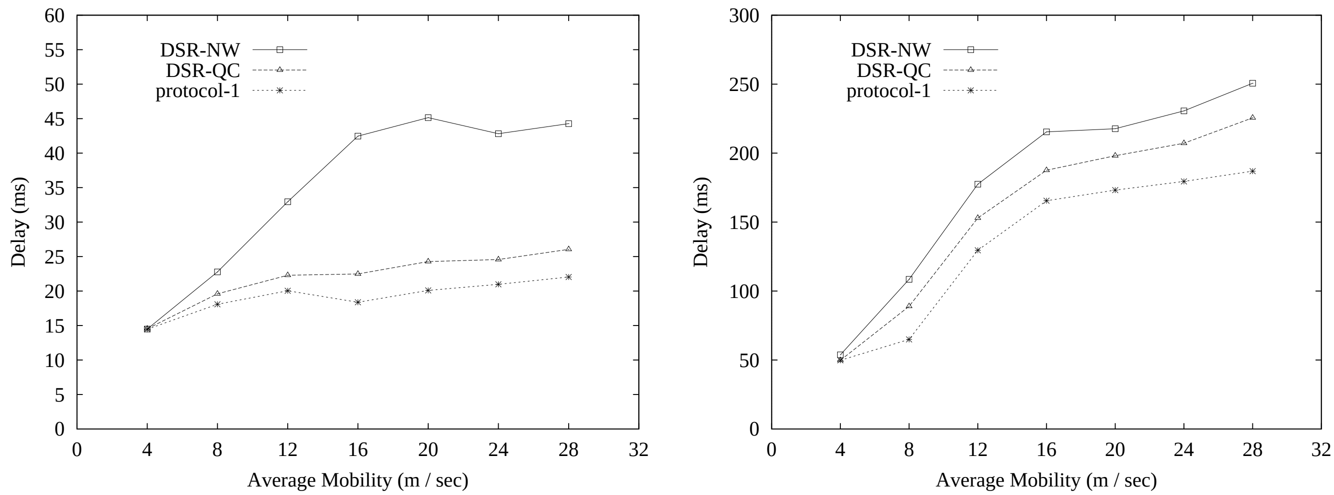


(c) Packet delivery fraction, square region



(d) Packet delivery fraction, square-with-a-hole region

Figure 3.12: Routing overhead and data packet delivery metrics for 9 conversation/node with varying mobilities.



(a) Average delay, square region

(b) Average delay, square-with-a-hole region

Figure 3.13: Average delay metric for 9 conversation/node with varying mobilities.

In [10] the problem of distributed QoS (Quality of Service) 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.

3.4 Concluding Remarks

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. 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 the network-wide flooding and almost always more than 20% compared to a simple query containment technique. This also indirectly contributes to lower end-to-end delays for data packets because of reduced network congestion. The reduction in delay was found to be close to an order of magnitude for high mobility scenarios. Our performance estimates presented here are fairly conservative as our simulation platform does not model multiple-access protocols on the wireless media. We anticipate greater levels of performance benefits in real systems, as reduction of routing overheads will also translate to more efficient bandwidth utilization and less multiple-access interference.

Chapter 4

On-Demand Multipath Routing

As was demonstrated in Chapter 3, the routing overhead can be reduced by localizing the query flood to a limited region by making intelligent use of routing histories. However, this technique is limited to reducing the “extent” of the flood. In this chapter we focus on another technique to reduce the routing overhead by reducing the “frequency” of the flood by discovering multiple possible routes from a single flooded query and using the alternate routes on route failure. Our goal is to provide enough redundancy at a low cost. We propose two multipath techniques for the on-demand routing protocol DSR that use multiple *disjoint* paths and show how intelligent use of these techniques can reduce the frequency of query floods. The goal is to provide enough redundancy at a low cost.

The idea of multipath routing is not new. It always has been a favorable alternate both for circuit switched and packet switched networks, as it provides an easy mechanism to distribute traffic and balance the network load, as well as provide fault tolerance. See for example, [69] for some prior work on multipath routing on packet switched networks. The ad hoc network community has also investigated multipath techniques, *albeit* less vigorously. See [54, 5, 38] for some examples. However, performance benefits of multipaths has never been truly evaluated on mobile ad hoc networks.

4.1 Multipath Dynamic Source Routing

We first consider the situation where the destination replies to a selected set of queries. Note that many copies of the flooded query message arrive at the destination via different routes. Consider that the queries

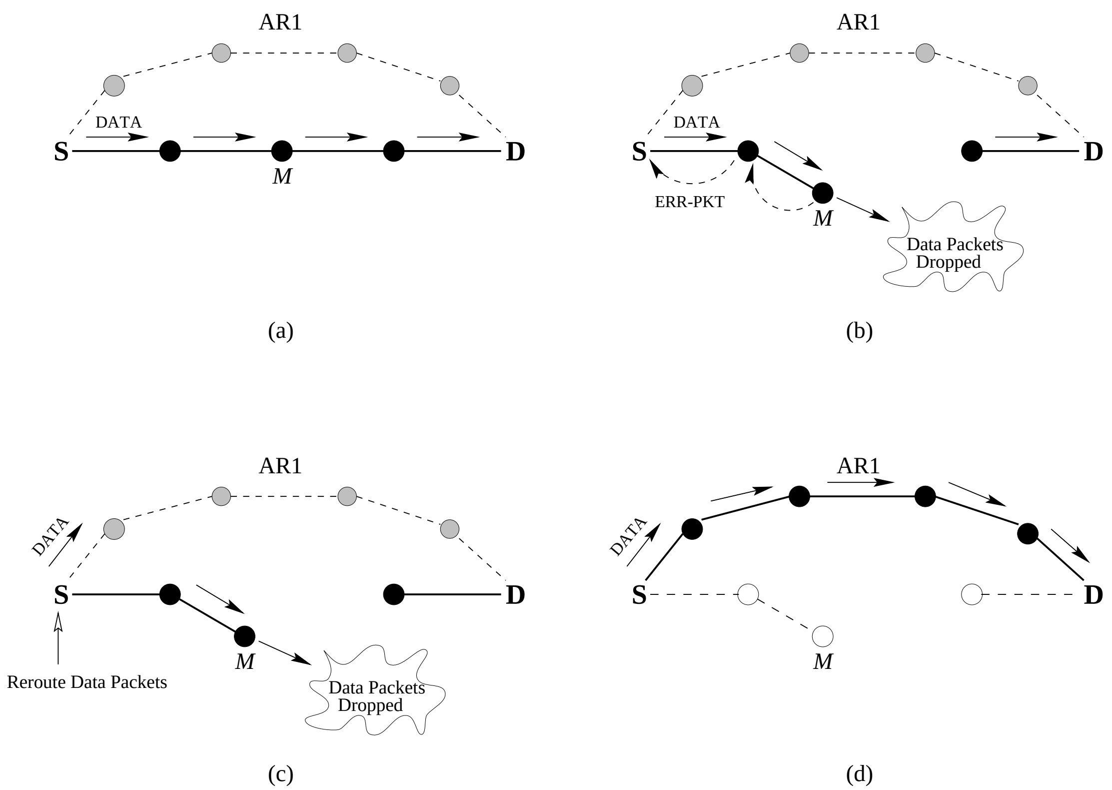


Figure 4.1: Multipath protocol 1.

that are replied to are those that carry a source route that is *link-wise disjoint* from the *primary* source route. The primary source route is the route taken by the first query reaching the destination. This usually defines the shortest route between the source and the destination. The destination "remembers" the primary route, in order to figure out which later requests to respond to. Only disjoint routes are chosen, as then the routes are independent and a link failure in one route does not effect the others. This also implicitly controls the total number of replies, thus preventing a reply flood occurrence. The source keeps all routes received on reply packets in its route cache. When the primary route breaks, the shortest remaining *alternate* route is used. This process continues until all routes break, wherein a fresh route discovery is then initiated. The number of alternate routes used can be a selectable parameter of the protocol. We will later show that only a few routes are actually sufficient. Let us call this technique *protocol 1*, for ease of later reference. Figure 4.1 depicts the operation of protocol 1 with one alternate route (AR1).

The source S uses the primary route (shown with black solid lines and black solid nodes) for transmitting data packets to the destination D until a link failure occurs at some point, (say node M moves). When a link is broken, node M will respond by generating an error packet and sends it back to the source S. This is shown in Figure 4.1 (b). Upon receiving the error packet, the source S will then reroute future data packets by utilizing its alternate route $AR1$, by copying this route into the data packet header as depicted in Figure 4.1 (c). Data packets will continue to use this alternate route $AR1$, Figure 4.1 (d), until another break occurs within this route. Then another error packet will be generated and sent back to the source S whereupon, having no more alternate routes, will begin a new route discovery for a fresh route to destination D.

Protocol 1 equips only the source with alternate routes. An intermediate link failure on the primary route still sends an error packet back to the source, which will then use the alternate route. This causes a temporary loss of route for the data packets that are already in transit upstream from the failed link. To prevent this, a better alternative is explored. All intermediate nodes are now equipped with a disjoint alternate route so that in-transit data packets no longer face any route loss. To accomplish this, the destination now replies to each intermediate node in the primary route with an alternate disjoint route to the destination. Note that any such reply is in response to a query from the source that has traveled through that intermediate node (i.e. this intermediate node is on the primary route that the source is using). The reply is targeted to the intermediate node instead of the source. Note that it may not always be possible for all intermediate nodes to get an alternate disjoint route. This would be particularly true for sparse networks. Thus, this still may result in temporary loss of routes on link failures until an upstream node switches to an alternate route.

Figure 4.2 depicts the operation of this technique (henceforth called *protocol 2*) using one alternate route per node on the primary route. As in the previous example for protocol 1, the source S uses the primary route for transmitting data packets to the destination D until link failure occurs at some point, because say, a node $M1$ moves. This time when the link brakes, node $M1$ will replace the unused portion of the route in the data packet header by the alternate route $AR2$, shown in Figure 4.2 (b). Thus, unlike protocol 1, data packets will then continue along their new route towards the destination unbeknown to the source that any such rerouting has occurred to the data packet (see Figure 4.2 (c)). This will continue until a link on $AR2$ breaks. Say for example node $M2$ moves as shown in Figure 4.2 (d). It will cause an error packet to be transmitted backward up to the source S, where it will then switch all impending data packets to its own alternate route $AR1$ by modifying the source route in the packet header as before, (shown in Figure 4.2

(e-f)). Finally, when a link breakages occurs along $AR1$ the generated error packet will be sent back to the source S, where the actions of protocol 2 will now be forced to do a new route discovery.

Both examples that were just illustrated were with one alternate route for any given node along the primary route. If a node has more than one alternate route, then a node will continue to use any available alternate route until it has exhausted its supply. Then an error packet will be sent up stream to its neighbor where the same proceedings will take place at that node.

Thus, for protocol 2, loss of *all* alternate routes in a node to the destination generates an error packet back to the source. Any intermediate node with an alternate route to the destination may quench the error packet. This node is also responsible for modifying the source route on all following data packets to use its own alternate route. This process continues until the source gets an error packet and has no alternate route to fall back on (implying that all nodes along the primary route have no more valid routes to the destination). Then a new route discovery is initiated. Again, note that in our simple example we maintained only one alternate route per node. In principle, any number of such routes can be maintained for both protocols. However, as our simulation results will show, more than one alternate route provides only a very minimal additional advantage for either protocol.

4.2 Performance Evaluation

We simulated the DSR protocol with single path as well as multipath routing for a mobile, ad hoc network of size 60 mobile hosts. As in previous chapters, the event-driven, packet-level routing simulator MaRS is used for the simulation study. We ran the experiments with the speed of each move uniformly distributed between 14.0 – 18.0 m/sec (average 16.0 m/sec). Distance is exponentially distributed with a mean of 5 m, and the direction is uniformly distributed within $[+90°, -90°]$ with respect to the direction of the previous move. Nodes are again assumed to have a radio range of 350 m. Workloads and region area are the same as described in Subsection 2.1.2 of Chapter 2. All simulations are run for 300 simulated seconds and each point in a plot represents the final average of 5 runs with different random number streams.

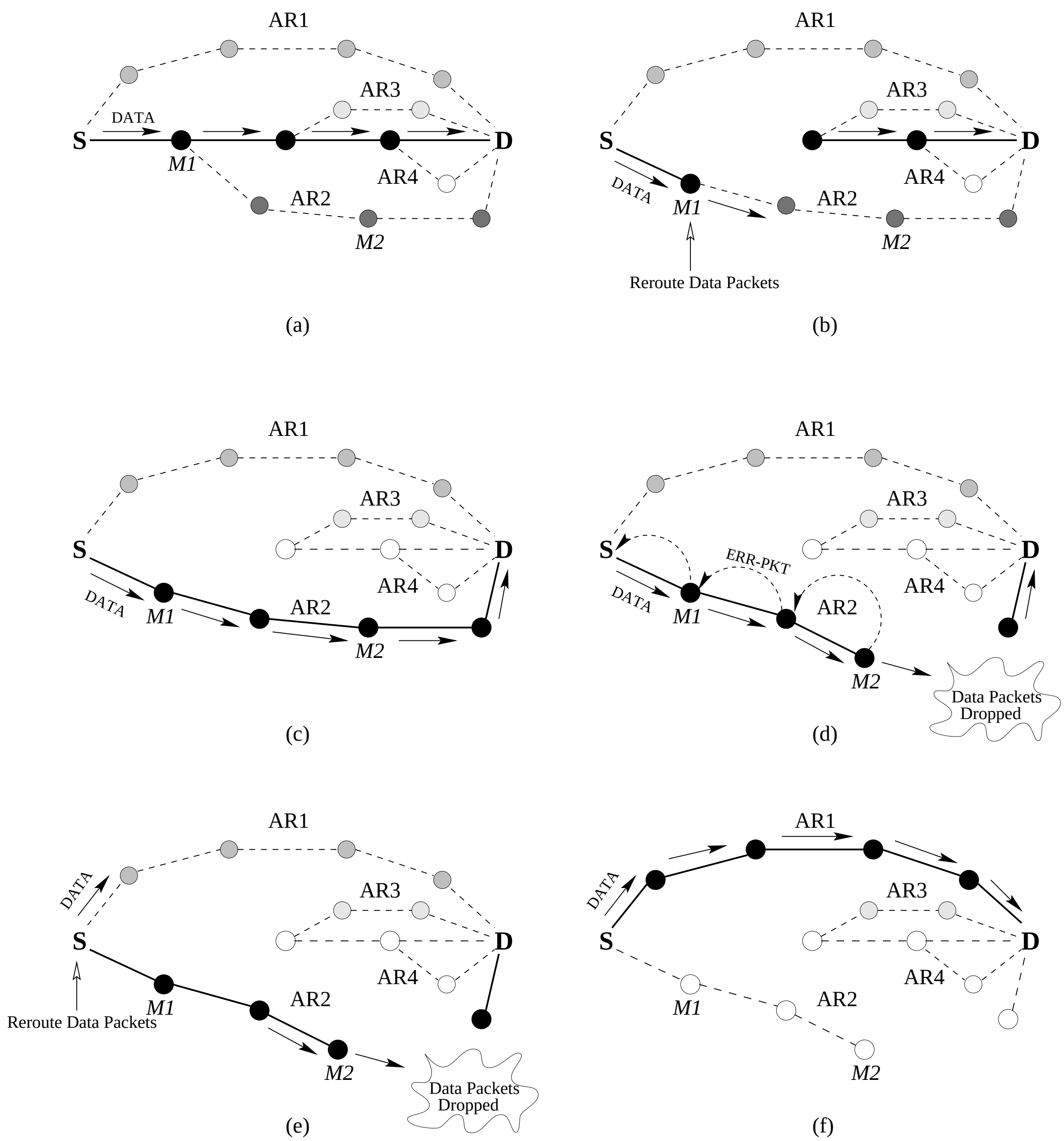


Figure 4.2: Multipath protocol 2.

Four important performance metrics are evaluated:

- *Fraction of packets dropped:* measured as the ratio of the number of data packets dropped *en route* to the number of data packets sent by the source.

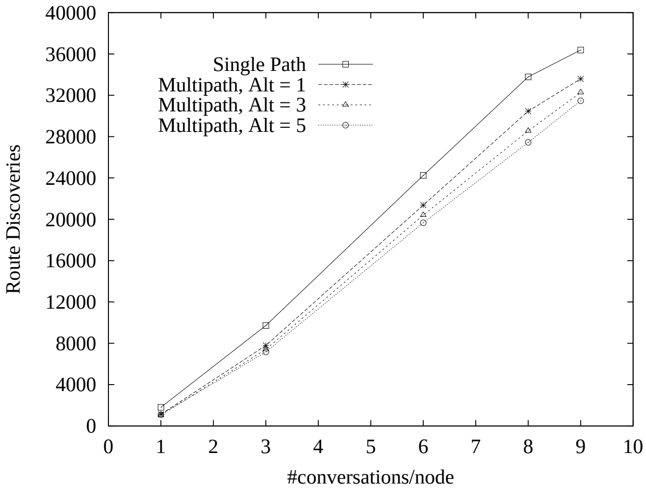
- *End-to-end delay:* measured as the average end-to-end delay latency of data packets; given in ms (milliseconds).

- *Number of route discoveries:* measured as the number of times a route discovery is initiated by any node during the simulation run (it is inversely proportional to the time interval between route discoveries).

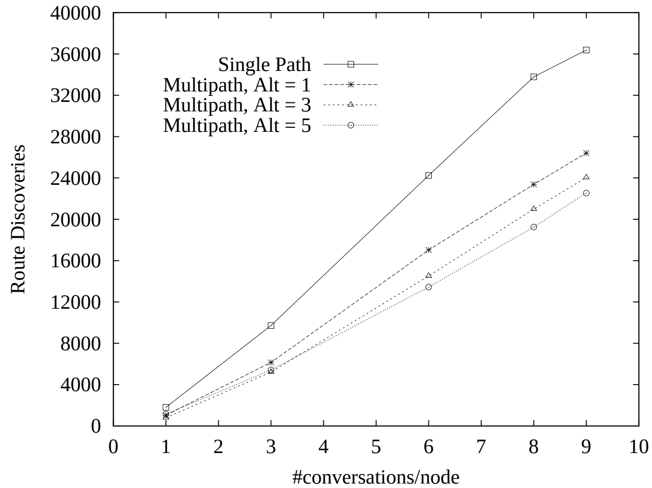
- *Routing load:* measured in terms of the total number of routing packets transmitted (broadcast transmissions are counted as a single transmission).

In the first set of experimental data (Figure 4.3), both multipath protocols are evaluated with various chosen numbers of disjoint alternate routes. As in all previous experiments, data packets are not buffered when a route discovery is in progress. When compared individually, both protocols benefit little in reduction of route discoveries when using 3 or 5 alternate routes in comparison to only using one alternate route. The same type of relative performance can be seen in the amount of data packets that are dropped. This observation can be attributed to the low percentage of alternate (disjoint) routes that may be actually sent to a node once the number is increased beyond one. Alternate routes being unique, automatically make those nodes within a route ineligible from being reused for other possible routes to be sent to the source (protocol 1) or to any other nodes along the primary route (protocol 2).

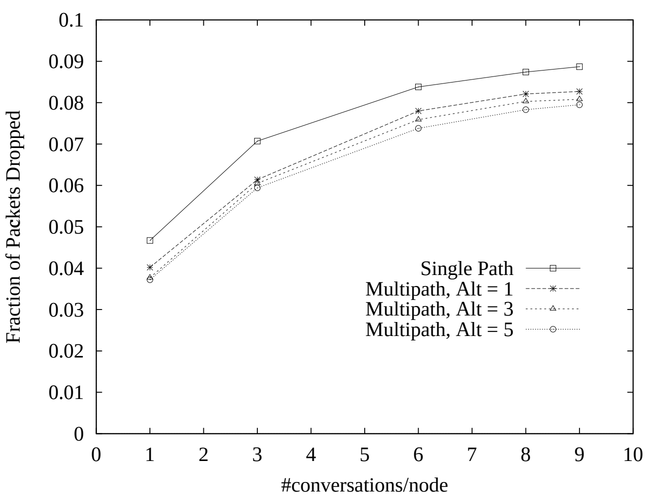
When the performances of the two multipath protocols are compared against each other, we can see that protocol 2 has a clear advantage over protocol 1. The credit for this is in how the alternate routes are being reciprocated. Recall that in protocol 1 only the source will receive alternate routes to the intended destination. In protocol 2, the source *and* the intermediate nodes along the primary route are receivers of alternate routes to the intended destination. During simulation runs for protocol 1, when a link failure occurs along the primary route data packets can only be salvaged when the source has been notified of the failed



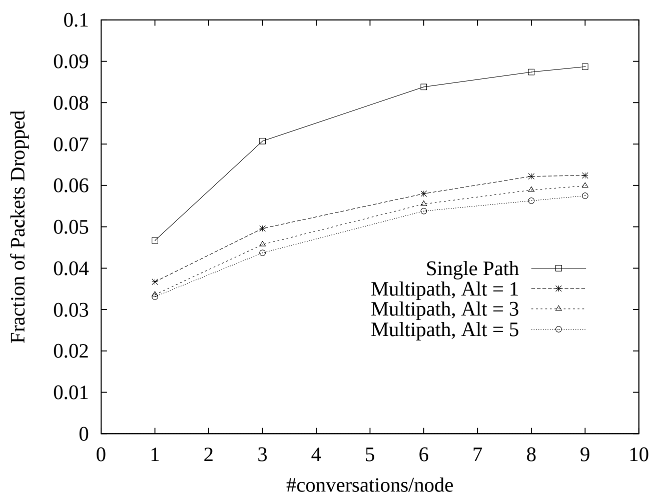
(a) Route Discoveries, Protocol 1



(b) Route Discoveries, Protocol 2



(c) Data Packets Dropped, Protocol 1



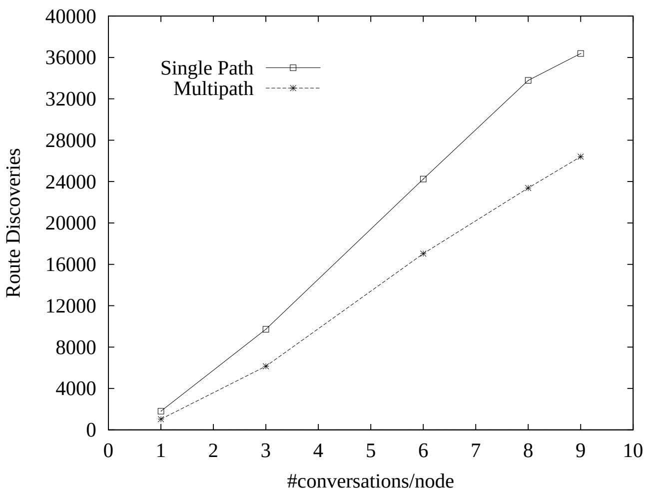
(d) Data Packets Dropped, Protocol 2

Figure 4.3: Number of route discoveries and fraction of data packets dropped for the single path and both multipath DSR protocols using 1, 3, and 5 alternate routes.

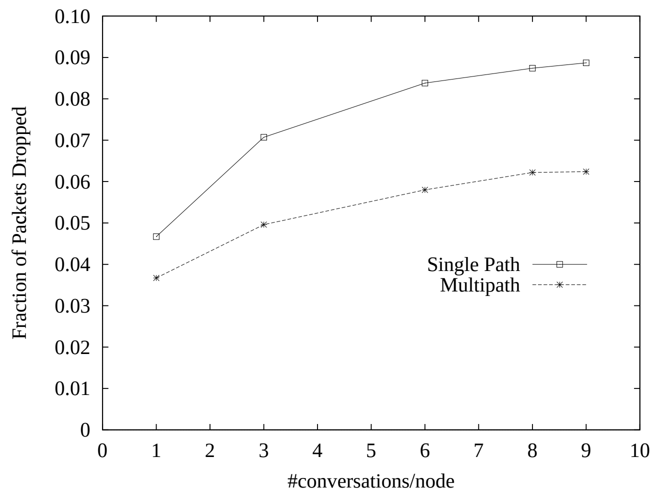
link *and* the source has an alternate route to the intended destination. All data packets in transit will be dropped. If the source does not have an alternate route, then a fresh route discovery will be induced. During this route discovery phase, all data packets will be dropped. Thus, the time period from when a link failure occurs and the source is notified to be given the opportunity to switch to an alternate route is crucial to the overall performance. This problem is further compounded if this time period is extended to include the time it takes for a route discovery to obtain a new route to the destination if there is no alternate route at the source.

In protocol 2, when a link failure occurs along the primary route data packets are given the opportunity to be salvaged at any node along the primary route (assuming the nodes have alternate routes). This supplemental opportunity undeniably gives protocol 2 the advantage in salvaging a higher percentage of data packets over protocol 1. Furthermore, given that a particular node does not have an alternate route, an error packet is generated and passed back to its upstream neighbor in order to give it a chance to salvage impending data packets. If so, then the error packet is not propagated any further, thus preventing an unnecessary route discovery from the source. In the worst case scenario, the error packet will be sent back to the source where it is giving the last opportunity to salvage data packets (i.e. protocol 1). Protocol 2's advantageous characteristics show up in Figure 4.3 (b) and (d) when compared to protocol 1, Figure 4.3 (a) and (c). The contingency that protocol 1 was faced with (discussed above) is alleviated in protocol 2 due to the help of the intermediate nodes. Route discoveries are reduced considerably in protocol 2 relative to protocol 1 as are the amount of data packets that are dropped. Based upon these results, protocol 2 was chosen as the protocol with the most potential. Since little is gained by going beyond one alternate route per node, this number was chosen as the fixed parameter for protocol 2. Henceforth, all experiments conducted are based upon protocol 2 with one alternate route.

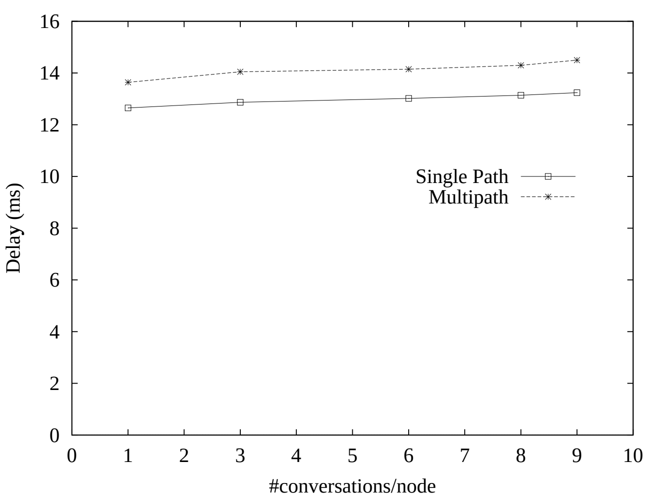
Figure 4.4 (a) shows significant improvements in the number of route discoveries for different number of conversations, demonstrating that the multipath protocol is able to reduce the number of route discoveries by roughly one-third. Significant savings in data packets dropped shows to be between 20-30% as shown in Figure 4.4 (b). Detailed instrumentation of the simulator reveals that 30-40% of delivered data packets use alternate routes. This shows up to be significant in our experimental results despite the fact that availability



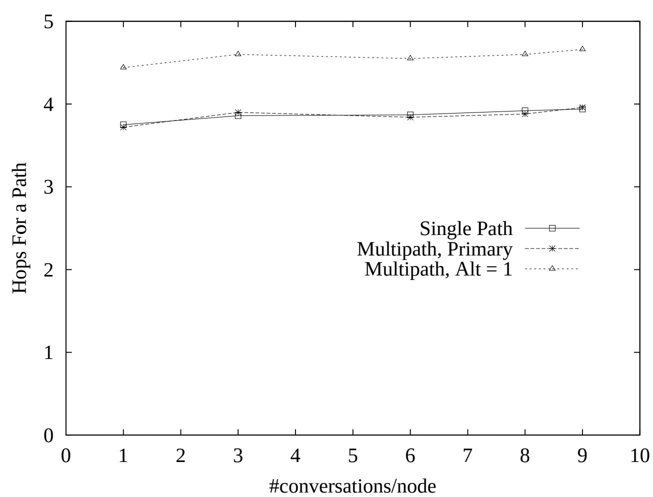
(a) Route Discoveries



(b) Data Packets Dropped



(c) End-to-end Delay



(d) Average Hops For a Path

Figure 4.4: Number of route discoveries, fraction of data packets dropped, end-to-end delay, and average hop length for the single path and multipath DSR protocol. The multipath protocol (protocol 2) uses one disjoint alternate route from each intermediate node (including the source) to the destination.

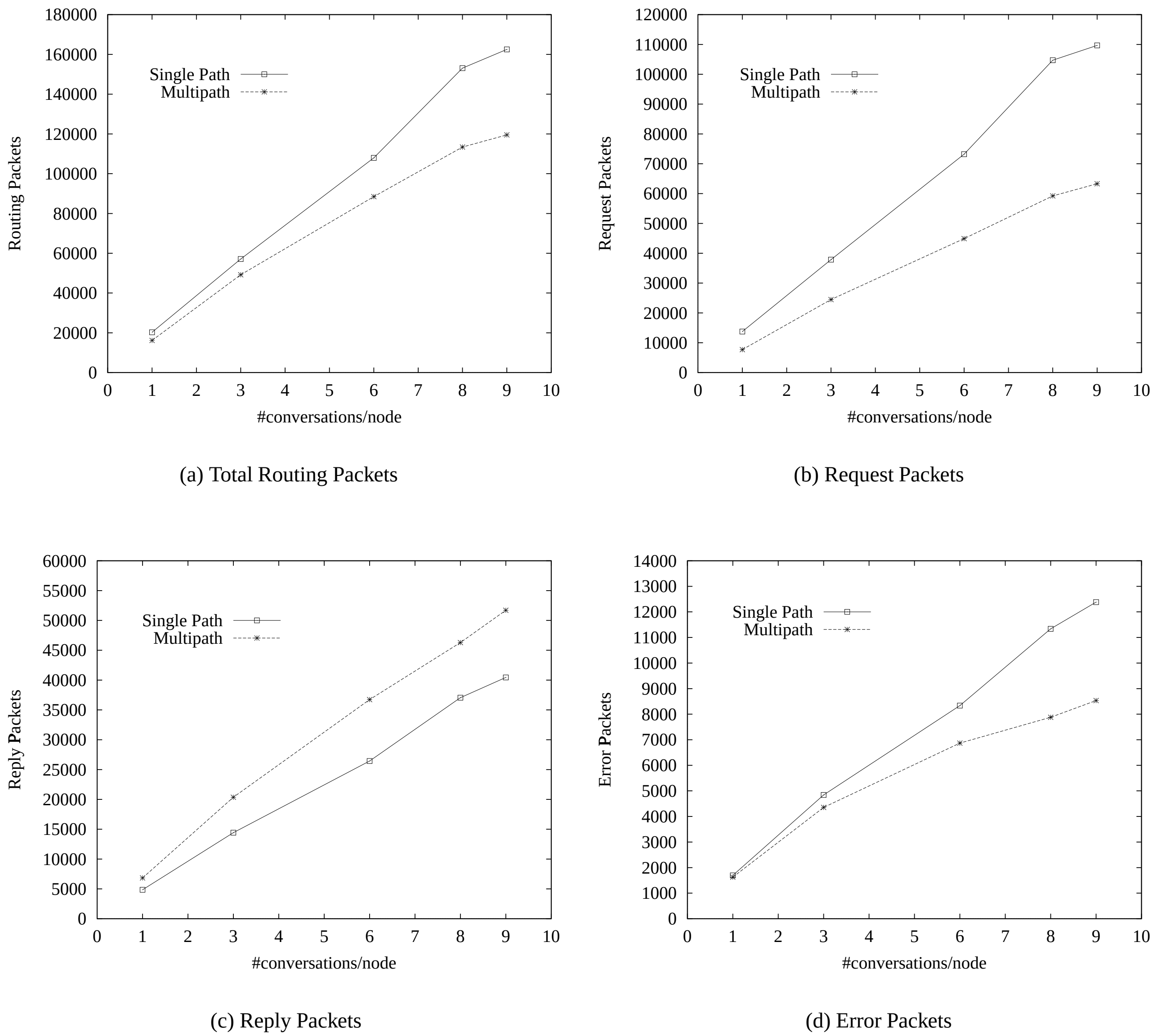
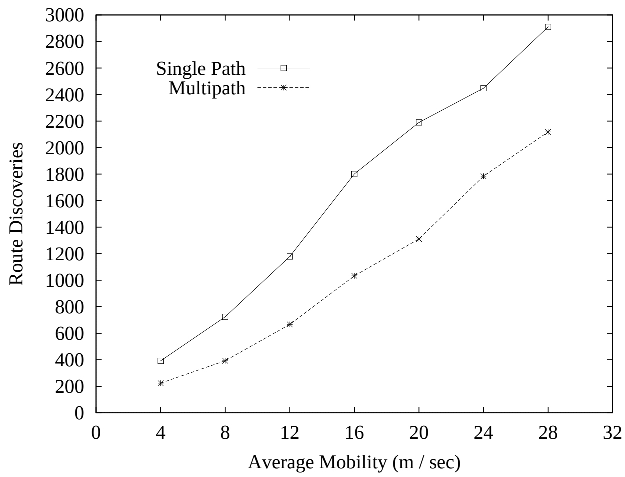


Figure 4.5: (a): Total number of routing packets transmitted during the simulation run for the same set of simulations as in Figure 4.4. (b)-(d): Breakdown of the routing packets in request, reply, and error packets.

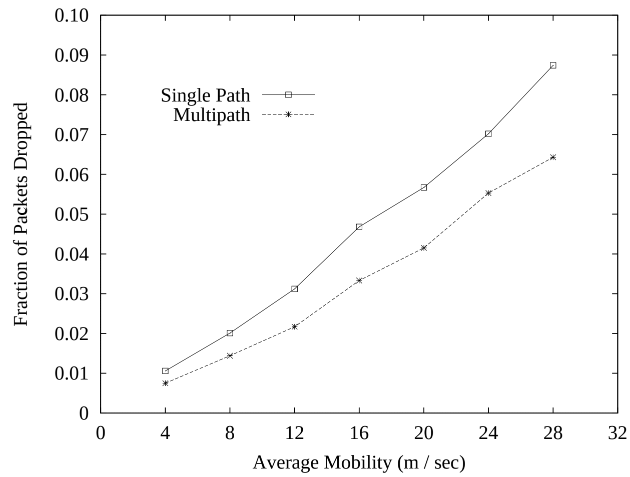
of alternate routes are not always guaranteed. However, as can be seen in Figure 4.4 (c), end-to-end delay has increased slightly in the multipath protocol. This is due to the fact that the alternate routes are typically longer than the primary routes (see Figure 4.4 (d)). The alternate routes are on an average about 1 hop longer than the primary route. The primary route length was found to be about 4 average wise. Recall that using disjoint alternate routes leaves any node to be used only once in any alternate reply to any node within a primary route for any given *source,destination* pair. As nodes that are close to the vicinity of the nodes of the primary route are used in alternate replys, the availability of nodes further away from these *used nodes* can only be used. This in turn increases the length of alternate routes which also extends the latency of the end-to-end delay.

Figure 4.5 (a-d) shows the routing load and its breakdown into request, reply, and error packets. Note that the number of route request packets (Figure 4.5 (b)) come down for the multipath protocol almost proportionate to the number of route discoveries. The replies, on the other hand, increase somewhat as shown in Figure 4.5 (c). This is expected because of multiple replies per route discovery cycle. This slight increase in reply (unicast) packets are countered off by the more significant reduction of request (broadcast) packets. The error packets also come down because of fewer route errors generated, as seen in Figure 4.5 (d). Recall that protocol 2 can have its intermediate nodes along the primary route salvage data packets, unlike protocol 1. This in turn prohibits an error packet from traveling any further than what is needed. Overall these translate to savings on the routing load. The savings are more significant on high loads (close to a quarter) which shows that these alternate routes are helpful even for a higher ratio of conversations per node. Thus, preventing an overabundant amount of route discoveries that would normally occur for such a high load.

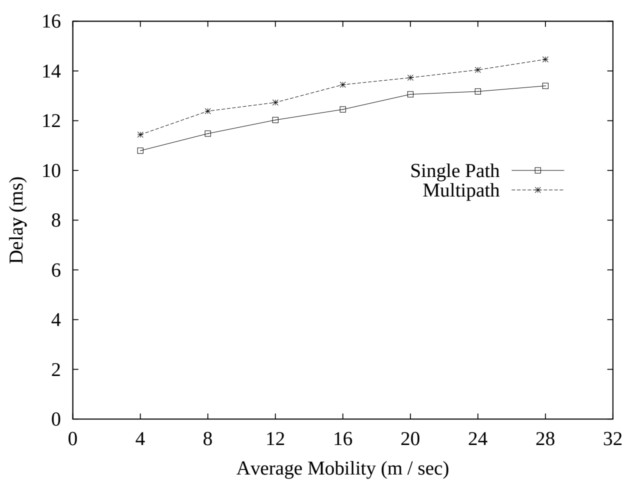
Figures 4.6 - 4.9 show results from a similar set of experiments with varying mobility but constant traffic loads at 1 (low traffic load) and 9 conversations (high traffic load), respectively, per node. General observations are the same as above, except that we find a smaller fractional improvement of the routing load and number of route discoveries for higher mobilities. This is due to the fact that for higher mobilities it is more likely that a link failure of an existing route is accompanied by a link failure of its alternate route. The



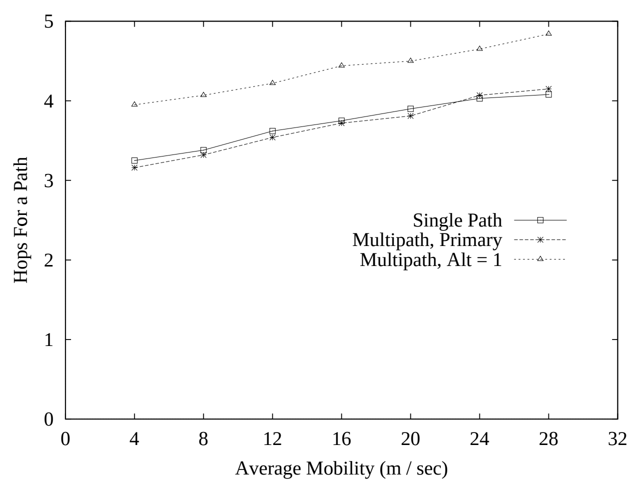
(a) Route Discoveries



(b) Data Packets Dropped

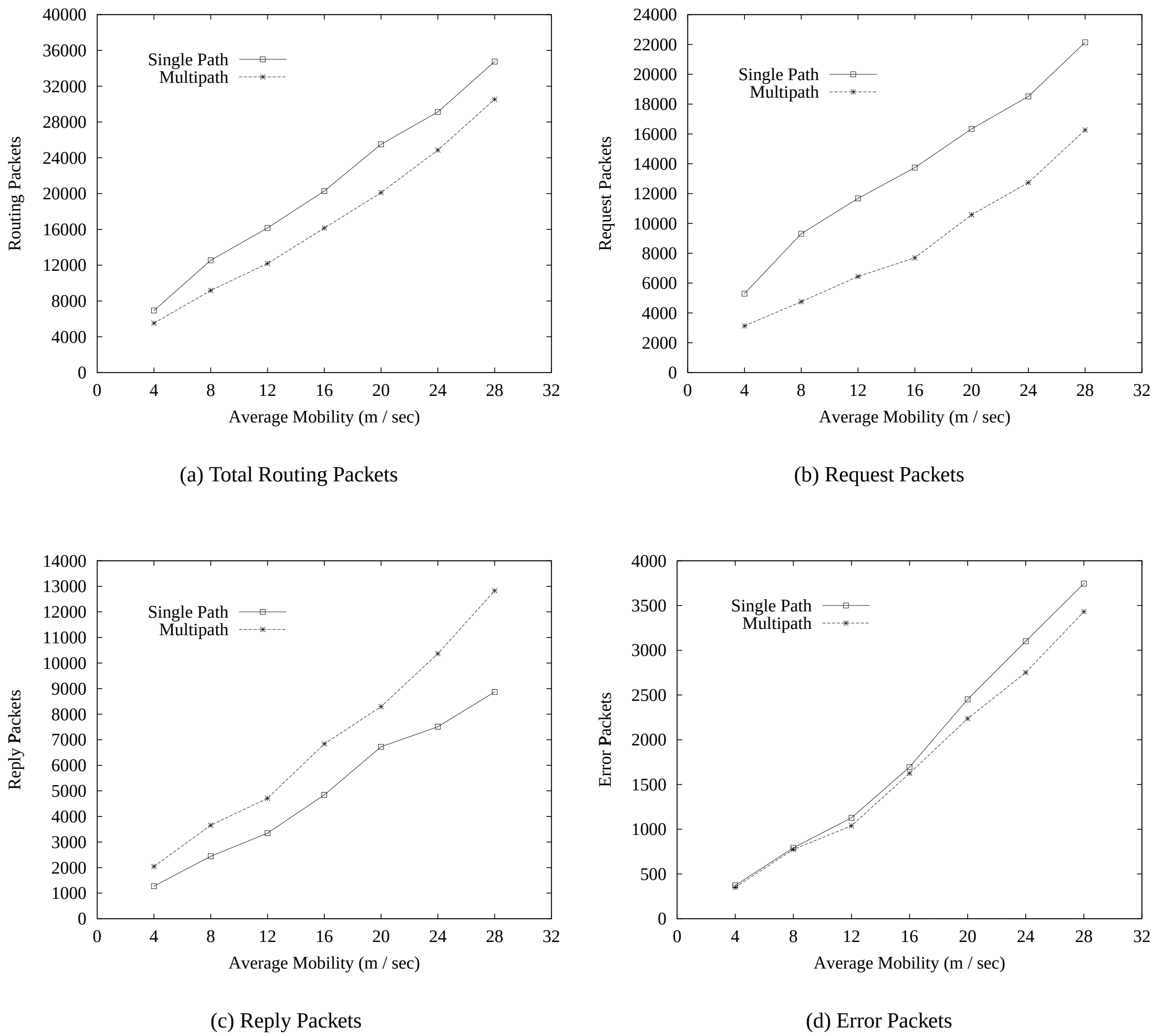


(c) End-to-end Delay



(d) Average Hops For a Path

Figure 4.6: Number of route discoveries, fraction of data packets dropped, end-to-end delay, and average hop length for the same network with varying mobility. Now the traffic load is fixed at 1 conversation per node.



(a) Total Routing Packets

(b) Request Packets

(c) Reply Packets

(d) Error Packets

Figure 4.7: (a): Total number of routing packets transmitted during the simulation run for the same set of simulations as in Figure 4.6. (b)-(d): Breakdown of routing packets in request, reply, and error packets.

breakages of the existing route and the alternate route may be correlated, for instance, when the existing route breaks due to movement of the destination node.

4.3 Related Work

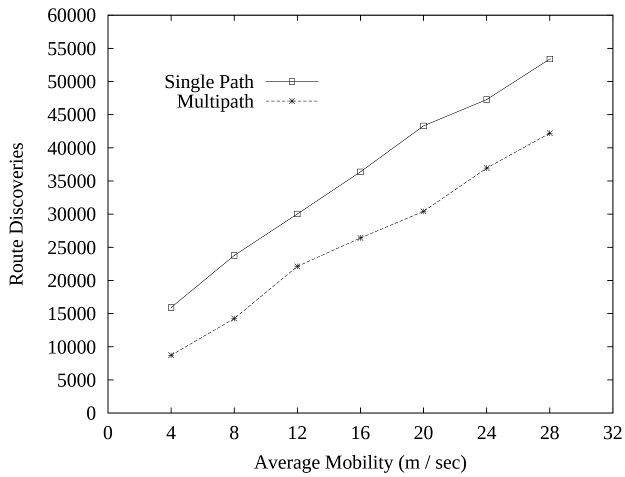
As already mentioned, the idea of multipath routing is not new but incorporating the idea into mobile ad hoc networks needs to be investigated. The Temporally Ordered Routing Algorithm or TORA [54] provides multiple alternate paths by maintaining a "destination-oriented" directed acyclic graph (DAG) from the source. However, TORA does not have an easy mechanism to evaluate the "quality" of these multiple routes. Because of the nature of the protocol, it is hard to determine which route is the shortest. In addition, TORA did not perform well in comparison to other on-demand protocols as our simulation results in Chapter 2 have shown.[1]

The Dynamic Source Routing (DSR) protocol [5, 38] also has an option of maintaining multiple routes, so that an alternate route can be used upon failure of the primary one. In DSR, the quality of routes (i.e. hop-wise lengths) can be easily evaluated and the best (i.e. the shortest) one can be used. But in DSR [38] too many routes are maintained in a trivial manner, without any regard to their ultimate usefulness. Further, if alternate routes have a lot of overlap with the primary route, a link failure in the primary route may also force them to become invalid.

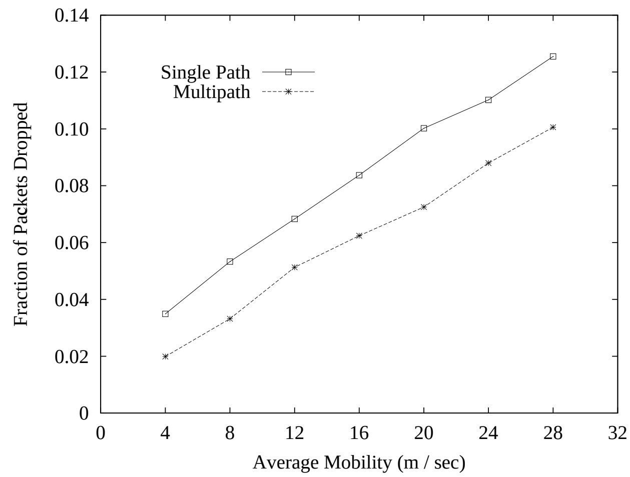
4.4 Concluding Remarks

We proposed two multipath extensions for the popular on-demand routing protocol DSR. The extension explores alternate, disjoint routes that can be useful in case the primary route breaks. Two variations were explored. In the first, only the source gets multiple alternate routes. In the second, each intermediate node on the primary route gets an alternate route (aside from the source). The key advantage is the reduction in the frequency of route discovery floods, which is recognized as a major overhead in on-demand protocols.

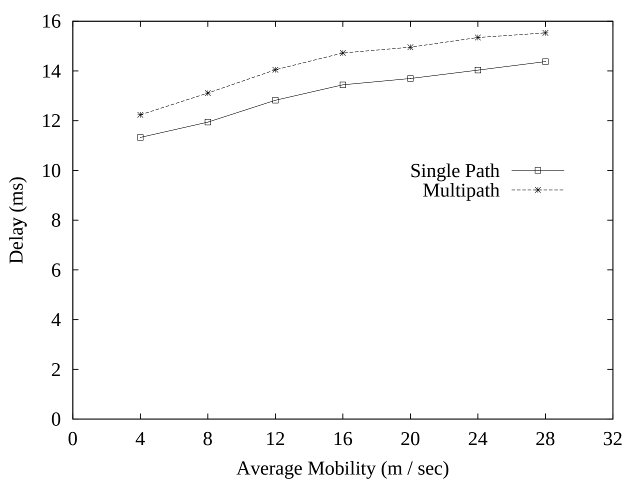
[1] The contemporary work by CMU in [6] also showed TORA to be deficient in performance relative to other on-demand protocols.



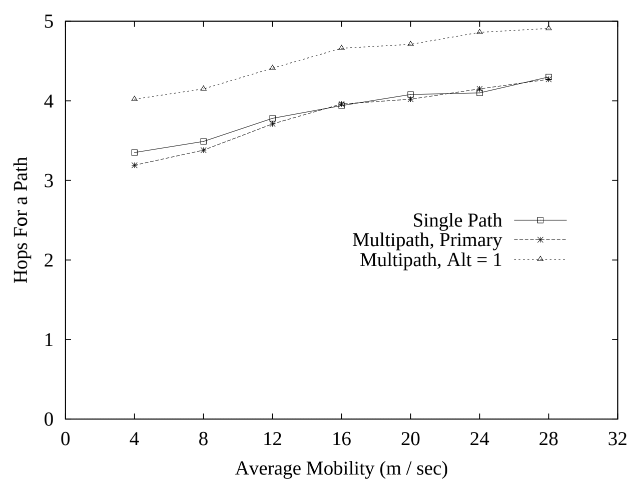
(a) Route Discoveries



(b) Data Packets Dropped

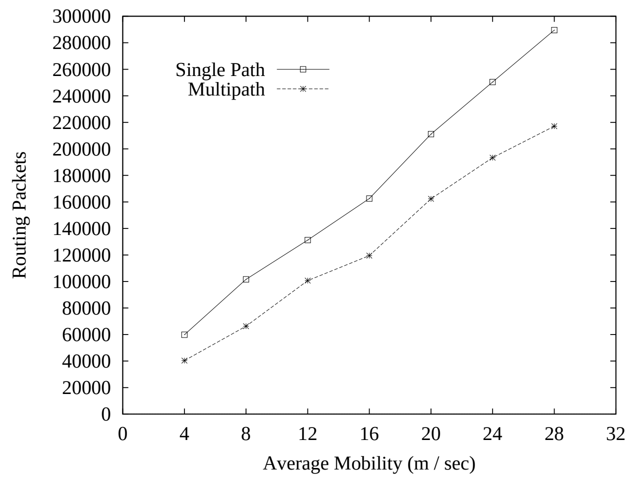


(c) End-to-end Delay

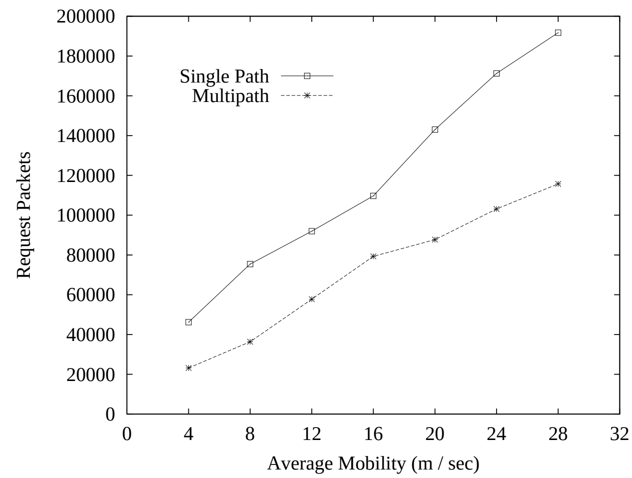


(d) Average Hops For a Path

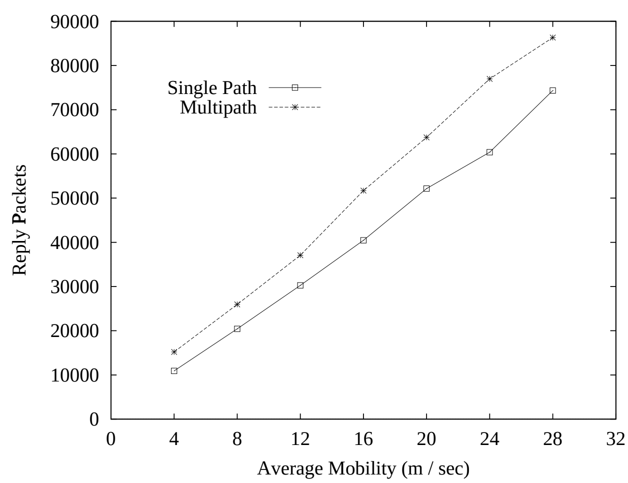
Figure 4.8: Number of route discoveries, fraction of data packets dropped, end-to-end delay, and average hop length for the same network with varying mobility. Now the traffic load is fixed at 9 conversations per node.



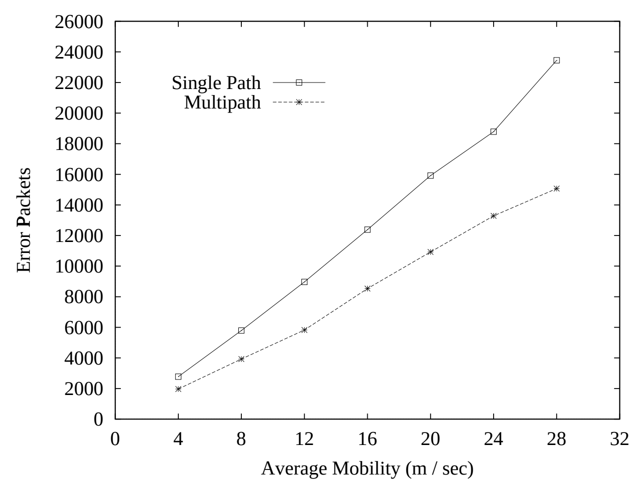
(a) Total Routing Packets



(b) Request Packets



(c) Reply Packets



(d) Error Packets

Figure 4.9: (a): Total number of routing packets transmitted during the simulation run for the same set of simulations as in Figure 4.8. (b)-(d): Breakdown of routing packets in request, reply, and error packets.

Performance evaluation using a packet-level routing simulator shows significant reductions in the number of route discoveries and hence in routing load from the use of multipath routing. The reductions could be up to about one half for route discoveries and similarly for the routing load. Also, as a consequence of alternate disjoint routes, fraction of dropped packets improved by up to about one-third. As was shown, alternate routes tend to be longer than the primary route. This in turn, leads to a slightly longer end-to-end delay for the delivered data packets. This results in a subtle trade off between end-to-end delay and routing load. Our simulations do not model the radio link layer and thus no multiple access interference is captured. We believe that in a real network, reduction in the routing load will also contribute to reduced end-to-end delay because of the reduced interference. Thus, the trade off will be more on the side of reduction of the routing load.

Chapter 5

On-Demand Power-Aware Routing

Even though much effort has been spent in understanding the behavior of the routing protocols and their comparative performance characteristics (see, for example, [6, 17, 16]), less attention has been paid to make these protocols energy-efficient. In critical environments such as military or rescue operations, where ad hoc networks will be typically used, conserving of battery power will be vital in order to make the network operational for long durations. Recharging or replacing batteries will often not be possible. This makes the study in power-aware routing critical. The challenge in ad hoc networks is that even if a host does not communicate on its own, it still frequently forwards data and routing packets for others, which drains its battery. Switching off a non-communicating node may not be always a good idea, as it may disconnect the network.

In this chapter we again focus on the on-demand routing protocol called Dynamic Source Routing or DSR [38, 7] and explore simple techniques to make this protocol power-aware. Even though we developed and evaluated these techniques for DSR, they should be applicable for other on-demand protocols of similar nature, such as AODV [55, 58]. On-demand protocols are chosen for power-awareness, as they typically have lower routing overheads than proactive distributed shortest path protocols (our simulations confirmed this in Section 2.2 of Chapter 2) and thus have lower baseline energy consumption.

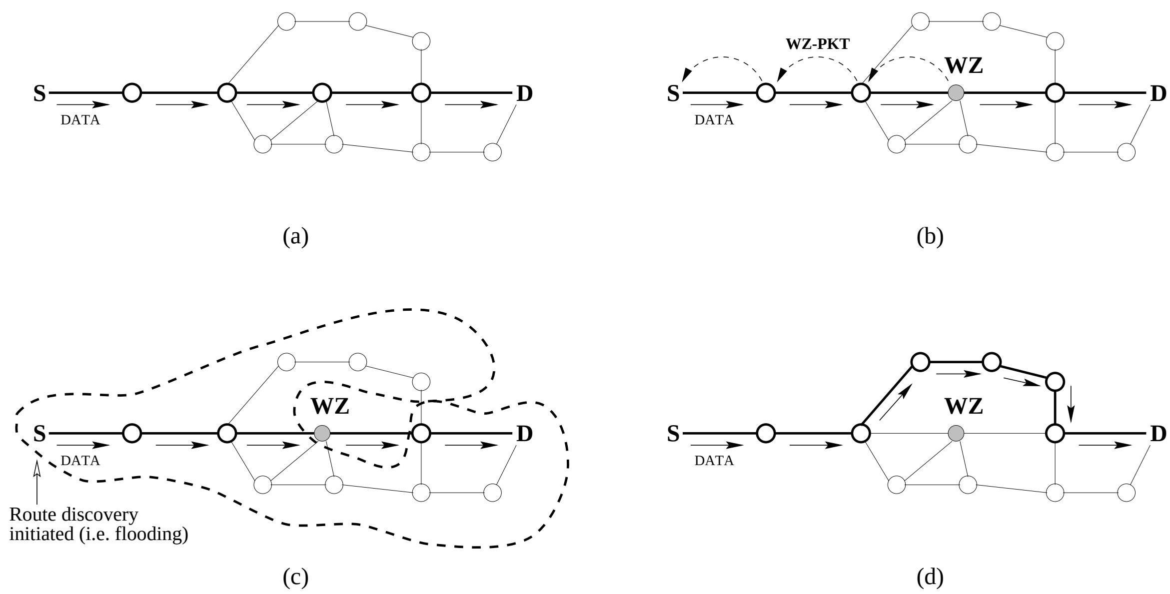


Figure 5.1: Warning Zone State

5.1 Power-Aware Routing

The technique we propose is built around two battery energy levels. We assume that the nodes can always determine the remaining battery energy and take appropriate routing decisions. Two energy thresholds are defined. Below the first one, the node enters a the *warning_zone* state, signifying that its battery is running low, but is still adequate to keep it running. However, at this point it wishes to relieve itself of the routing activities if alternative routers can be found. If alternatives can be found, they take over the routing. Otherwise, the *warning_zone* node continues routing as usual. If the battery of the *warning_zone* node continues to be depleted it eventually crosses another lower threshold, below which it enters the *low_zone* state. Here, the battery is low enough, that this node refuses to route packets for other nodes. But it can still function as a source or destination of packets. Eventually as the battery continues to be depleted, the node dies.

For the following description, refer to Figure 5.1. Figure 5.1 (a) shows an established route for the given source S, destination D pair. Upon entering the *warning_zone* state, a node (labeled WZ) sends back a warning zone packet (WZ-PKT) back to the source S to give it a warning that it should find another route in place of the existing route that *does not* include this warning zone node. However, this being a “courtesy”

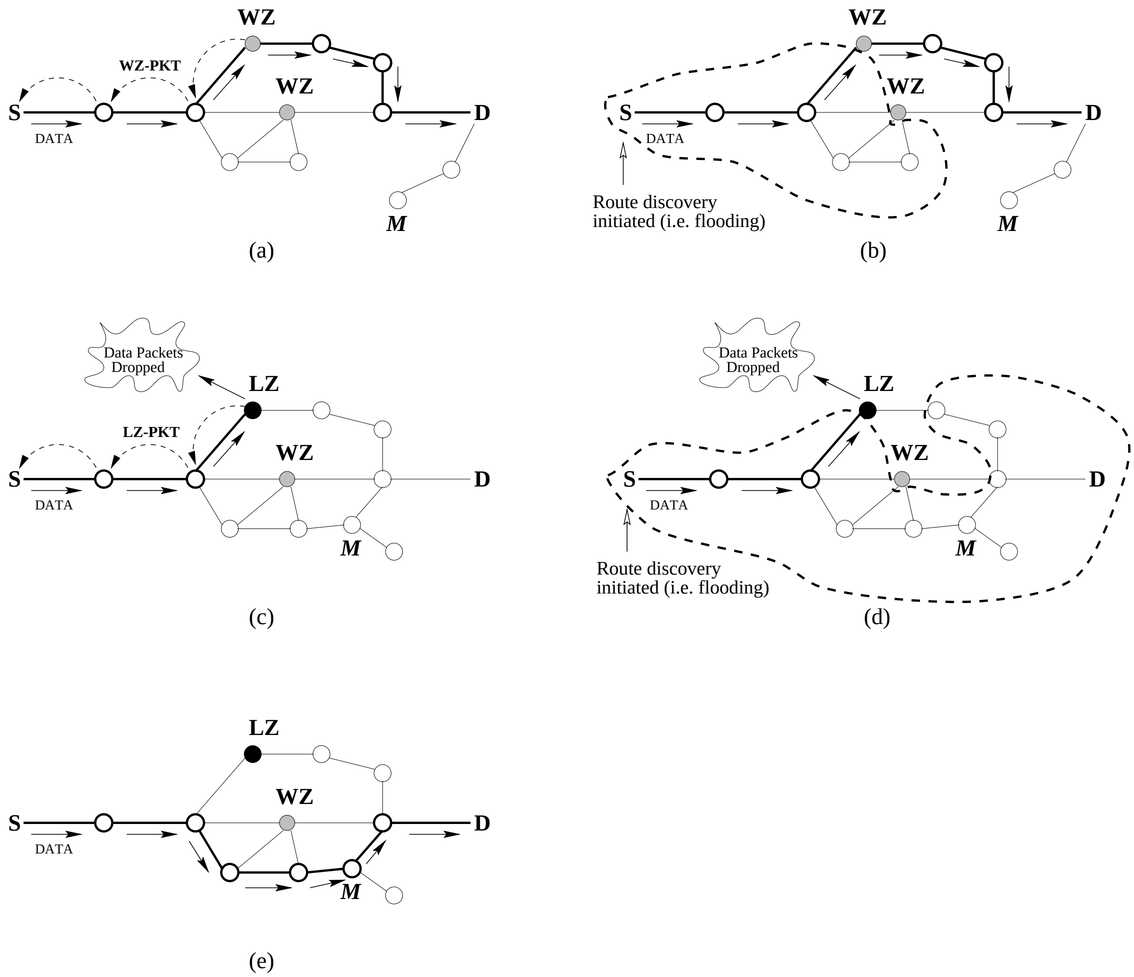


Figure 5.2: Low Zone State

warning, the WZ node will continue to route data packets for the source (shown in solid lines leading to the destination D) until it has found another route. This is shown in Figure 5.1 (b). Upon receiving an WZ-PKT, the source S starts using an alternative route from its cache or, if there is no such route, it initiates a route discovery to find alternate routes. The latter scenario is shown in Figure 5.1 (c). The entire region encapsulated by the dashed line signifys RREQ packets that are disseminated due to the route discovery originated at the source S. Notice that the WZ node does not participate in this route discovery. After receiving RREQ packets, the WZ node will not forward them, thus guaranteeing that it will not be included in the new route. Figure 5.1 (d) shows the new route (solid lines) that the source S has obtained via a route discovery to the destination D.

Figure 5.2 depicts such a scenario for a node going into the *low_zone* state from the *warning_zone* state. This sequence, beginning with Figure 5.2 (a), is the node WZ *en route* from S to D. Let us assume that the source S cannot find a new route after it receives WZ-PKT and issues a route discovery as depicted in Figure 5.2 (b). Also, assume that node M has moved from its original location. Node WZ's battery level will continue to be depleted until it reaches the *low_zone* state (now shown as LZ in Figure 5.2 (c)). At this point it has no other option but to send back a low zone packet (LZ-PKT) to the source S telling it that it refuses to participate in aiding its data packets to reach the destination D. At this point forth, any more data packets that the LZ node encounters from S will be dropped. Also, node M is assumed to have moved once again. The LZ node will also refuse to participate in *any other* route discoveries that are currently occurring or will occur on the network. Again, network flooding is depicted as the region encapsulated within the dashed line in Figure 5.2 (d). The LZ node will continue, however, to send and/or receive its own data packets if needed. The final route discovered, which does not include both the LZ node and the WZ node, is shown Figure 5.2 (e).

Note that in either of the cases described above, a node will never completely shut itself down. Thus, a node will still keep its network interface on and receive packets. The reason is that without actually receiving packets it is not possible to know the intent of the packet. It is just the transmitting activity that will be curtailed depending on the state of the node. For example, in the *low_zone* state, a node will not transmit any data packet except its own and will not transmit any routing packet unless it is an error (RERR) packet. This scenario can occur if the source has not yet received the LZ-PKT and a link failure has occurred downstream from the vicinity of the LZ node along this current route. Error packets are always treated with top priority regardless of node's battery status (*low_zone* state or *warning_zone* state) and need to be delivered to the intending source. Note that other nodes along a broken route should also be informed when a link has failed so that they may also take appropriate action.

Note that potential route discovery activities after receiving a WZ-PKT or LZ-PKT will also drain the energy of other nodes. To optimize the overall energy budget, route discovery timeouts are manipulated. This timeout reflects how long the source node should wait for a reply before issuing another route discovery. A much longer timeout is chosen for route discoveries after receiving the WZ-PKT. If a route is not found, then the next route discovery will not occur immediately. Rather, the source waits for a period of time, which we call a *periodic timeout*, before issuing the next route discovery. The reason behind this is that, although a new route should be found, the scenario is not as urgent as when there does not exist any route. Recall, that the WZ node that sent the WZ-PKT is still willing to forward data packets as a courtesy to the source. Thus, for the time that the WZ node is still in the *warning zone* state, the source's data packets will still be forwarded. During the scenario that a source receives an LZ-PKT from an LZ node, however, this will be treated as though it were an RERR packet, to which the source will immediately clear this current route (or any other route for that matter containing this link) from its cache and proceed to begin a route discovery. Here there is no use of the *periodic timeout* since this is treated as a more urgent case in that a new route is needed to be found immediately. Thus, if no route is found soon then after a small timeout period, another route discovery will begin. The *small timeout period* is a predefined time that is the average time that it takes for a source to receive a route reply once it has started a route discovery and is considerably smaller than the *periodic timeout.*

5.1.1 Performance Tradeoffs

It is important to understand that the performance tradeoffs posed by the above power-aware mechanism. The mechanism does not directly attempt to optimize the power consumption of the network as a whole. It just wants to avoid using nodes with low battery levels, thus running the risk of nodes dying prematurely and possibly disconnecting the network. It is actually possible that the alternate route used after receiving a WZ-PKT or LZ-PKT is longer than the original route. This presents several inefficiencies. First, the data packet latency could be higher for longer routes. Second, overall energy expenditure for routing a data packet would be higher for longer routes. Third, additional route discoveries will also consume additional energy. However, the useful lifetime of the network will increase as nodes with low battery levels will be avoided as much as possible and overall energy drain will be fairly distributed all over the network instead of being concentrated in a few nodes. We will see evidence of this in the performance evaluation section.

Various network, traffic and mobility characteristics will have a varying degree of influence on our method. For example, if the network is sparsely connected the nodes will not have as much of a selection of alternate nodes. This will reduce the effectiveness of our technique. Similarly, a large number of conversations covering many different source-destination pairs may not gain very much from this technique. Here, many nodes in the network will act as routers and there will not be many free nodes to shift the routing load to. Lastly, high mobility somewhat balances out the routing load over all nodes as routes are frequently changing. Thus, high mobility scenarios may not gain very much from this technique either. We will see evidence of some of these conjectures in the next section.

5.2 Performance Evaluation

We simulated our power-aware routing technique as an extension to the DSR protocol for a mobile, ad hoc network. The DSR protocol is implemented with the *low_zone* state only, while the power-aware routing protocol is implemented with both the *low_zone* state and the *warning_zone* state as described before. Promiscuous listening was not implemented in either protocol as this has been shown to be a rather expensive power consumption-wise [25]. Again as in previous chapters, the event-driven, packet-level routing simulator MaRS is used for the simulation study. In the model we have simulated, 60 mobile hosts move around a rectangular region of size 800 m $\times$ 800 m according to the following mobility model. Each node chooses a direction, speed and distance of move 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 3.0 – 5.0 m/sec (low mobility experiments) and 7.0 – 9.0 m/sec (medium mobility experiments), distance is exponentially distributed with a mean of 5 m, and the direction is uniformly distributed within $[+90°, -90°]$ with respect to the direction of the previous move. 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.

The workload is the same as described in Subsection 2.1.2 of Chapter 2. We used 5 and 10 conversations with source and destinations selected randomly. All simulations are run until the data packet delivery fraction (defined below) drops below 80%. Each point in a plot represents an average of ten runs with different random number streams.

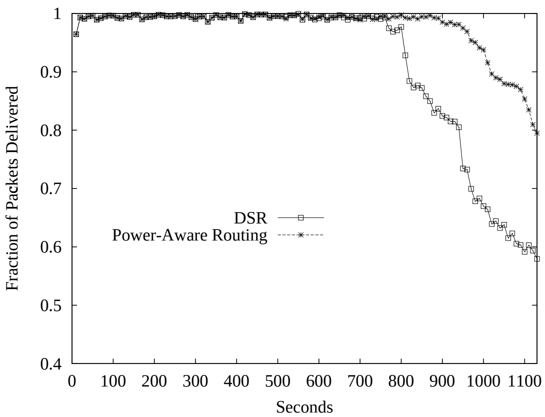
Five important performance metrics are evaluated:

- *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,
- *Battery level* – measured as the amount of remaining battery energy,
- *Fraction of nodes in the low zone state* – measured as the fraction of nodes in the network whose battery level is at or below the threshold of LOW_ZONE,
- *End-to-end delay* – measured as the average end-to-end latency of data packets,
- *Average number of hops* – average number of hops in the routes.

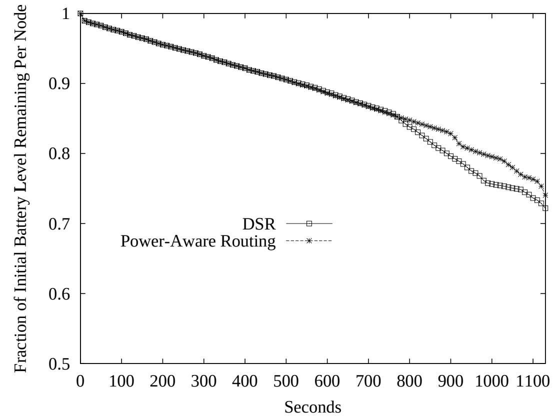
In the performance plots all these metrics are plotted across a time axis averaged over 10 second intervals in order to capture the overall run-time dynamics. For the two battery threshold levels, the following values are found to work well in our model. We have used a value of 45% of the initial battery level for the WARNING_ZONE threshold and a value of 20% of the initial battery level for the LOW_ZONE threshold. Following the published values for power consumption for various commercial radio network interfaces [66], we have assumed that a node spends 2 units of energy for transmitting versus 1 unit of energy for receiving a packet. To make the measurement simpler, we have used the same energy values regardless of the size of the packet. We assume that there is no battery drain when a node is idle – not receiving or transmitting any packets. In reality however, there is a drain whenever the network interface is on, even if it is idle. But this issue shouldn't affect our comparative evaluation. We have initialized the battery to 15,000 units of energy.

5.2.1 Simulation Results

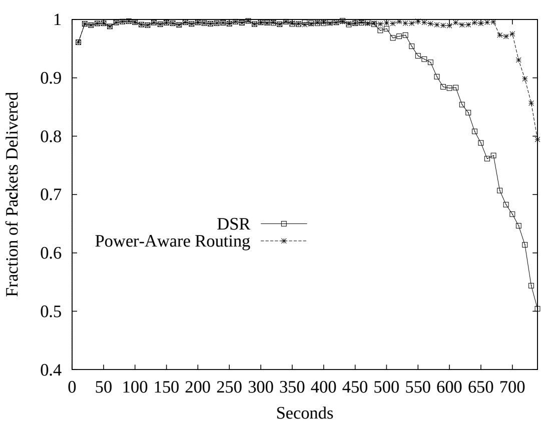
In the first set of experimental data (Figure 5.3), we ran our power-aware routing protocol against the baseline DSR using 5 conversations and 10 conversations, respectively. Both protocols were run using low node



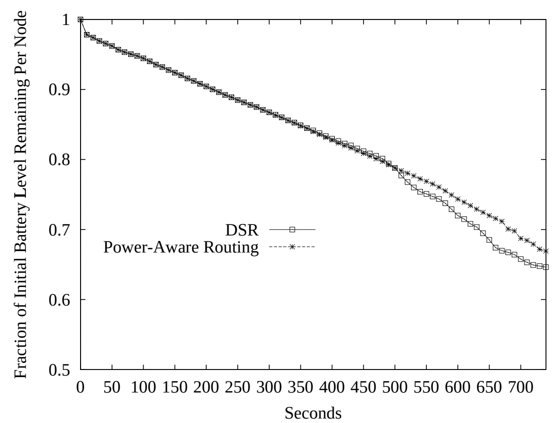
(a) Data Packet Delivery, 5 conversations



(b) Battery Level, 5 conversations



(c) Data Packet Delivery, 10 conversations



(d) Battery Level, 10 conversations

Figure 5.3: Data packet delivery and battery level remaining per node for 5 conversations and 10 conversations, respectively, for low mobility.

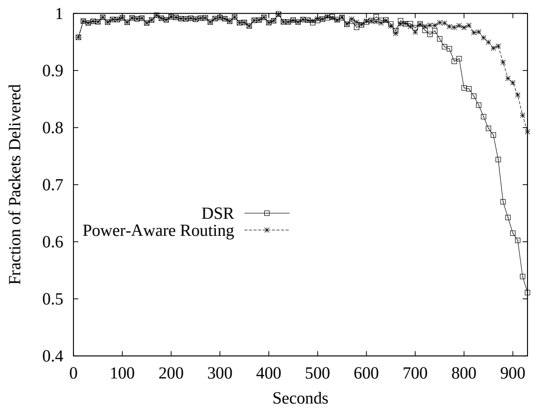
mobility. For both numbers of conversations experimented with, data packet delivery was greatly enhanced over a period of time with the power-aware routing protocol. This metric is shown in Figure 5.3 (a,c). In DSR some nodes started dying early, thus occasionally disconnecting the network. Nodes start dying much later in the power-aware routing protocol as our technique attempts to distribute routing participation once a node enters the WZ state. Overall, a packet delivery ratio of at least 90% was extended between 150 to 200 seconds. Figure 5.3 (b,d) shows that average battery consumption was also noticeably receptive in savings in relation to the DSR protocol over this time. A closer observation reveals a direct correlation between battery level savings and the extended time for high data packet delivery fraction.

Performances for 5 and 10 conversations run at medium mobility are shown in Figures 5.4 and 5.5. respectively. As can be seen for data packet delivery and battery level in both cases, Figures 5.4 (a,b) and 5.5 (a,b) respectively, the longevity for both metrics are extended considerably. Extended time is about 100 seconds for data packet delivery of at least 90% and about 70 seconds for data packet delivery of at least 80%. As before, there is a direct correlation between the average battery consumption and the extended data packet delivery.

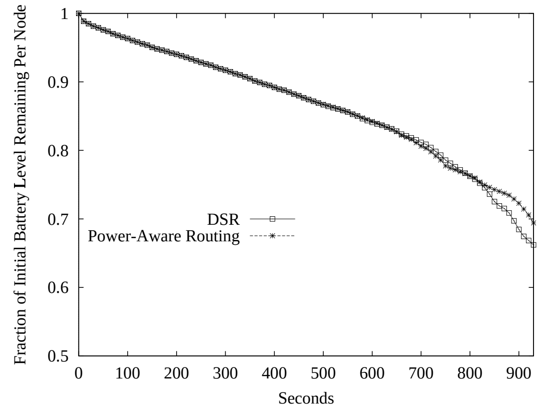
The fraction of nodes that have entered the *low_zone* state has been reduced by as much as 50% for most of the duration of the time as can be seen in Figures 5.4(c) and 5.5(c). Indicating that there are more active nodes available throughout the network with sufficient battery power to keep routing packets.

As time goes on, the average end-to-end delay increases slightly in our protocol. This is not surprising as the number of nodes available to route packets becomes less, route lengths are typically longer. This is shown in Figures 5.4(d,e) and 5.5 (d,e). Also, note that for the DSR protocol the inverse occurs as times goes on. Due to the higher percentage of nodes in the *low_zone* state there is less active nodes with sufficient battery power to re-route data packets. While a shorter end-to-end delay is attractive, this comes at the same time of a higher percentage of data packets being dropped. Thus, only source-destination pairs that are close in vicinity to each other are able to successfully transmit and receive data packets.

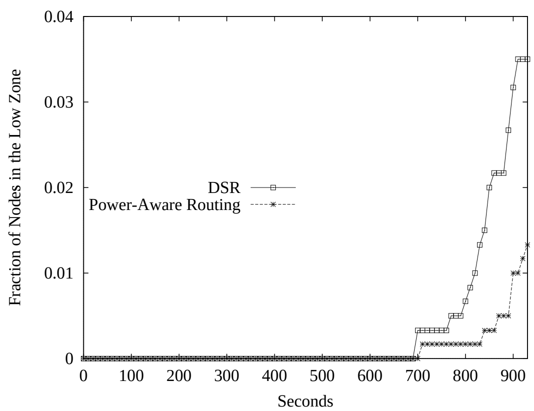
Further investigated experiments reveals a sensitivity to both a higher number of conversations and/or an increase in mobility. Figure 5.6 shows the performance metrics for a run of 20 conversations, once again in the same size network but now with a high mobility of 12 m/s. Note that there is still an increase (although small) in the data packet delivery performance for our power-aware routing protocol in comparison to the DSR protocol for a brief period of time in Figure 5.6 (a). This, in conjunction with the rapidly increase of nodes in the *low_zone* state shown in Figure 5.6 (b), shows that there will be little benefit with such a scenario as time goes on. The reason is two fold. First, as the number of conversations increases throughout the network, there will be more data packets being routed (higher communication pattern). Thus on average, nodes will be doing more work either sending, receiving, or more importantly, routing for others. As more conversations are introduced into the network the amount of room for optimization is reduced. Second, due to an increase in the amount of mobility, links fail more frequently. This increases routing overhead due to



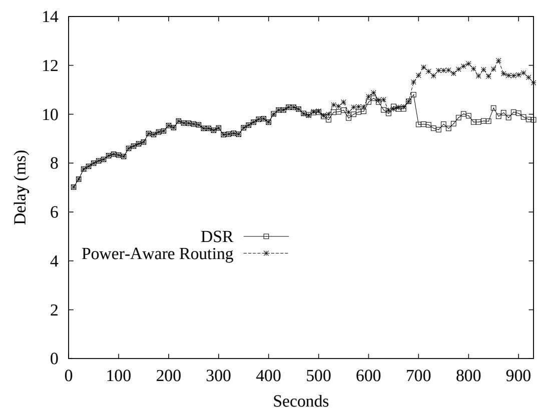
(a) Data Packet Delivery, 5 conversations



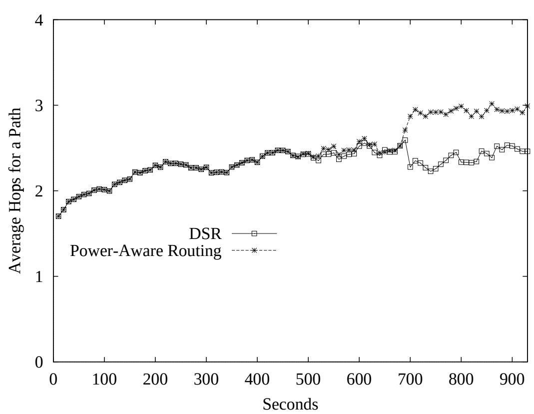
(b) Battery Level, 5 conversations



(c) Low Zone Nodes, 5 conversations

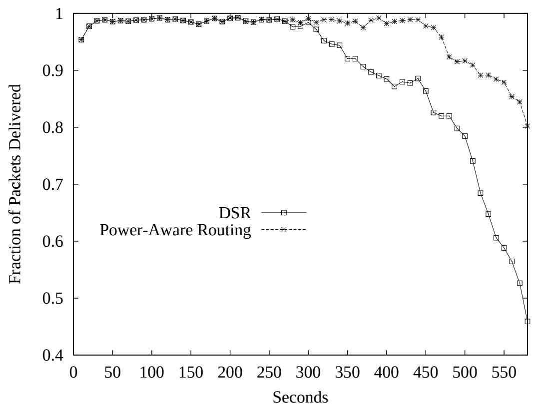


(d) Average Delay, 5 conversations

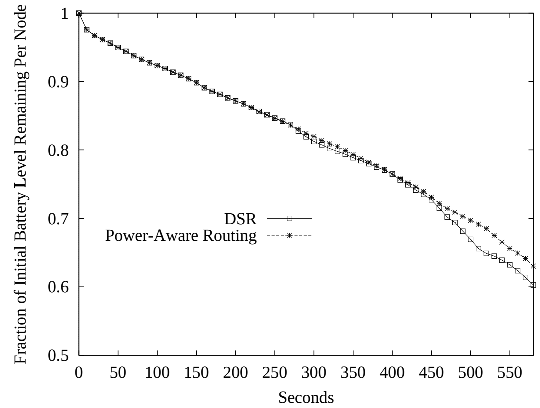


(e) Average Number of Hops, 5 conversations

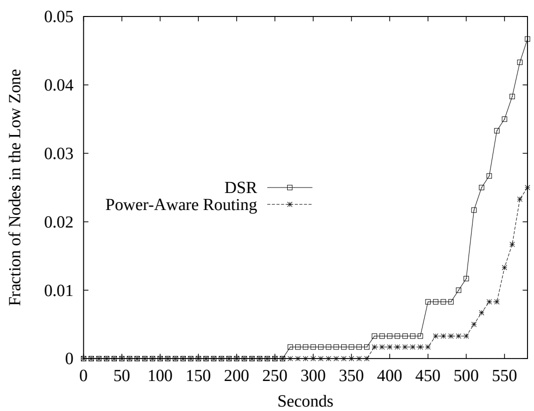
Figure 5.4: Various performance metrics for 5 conversations at medium mobility.



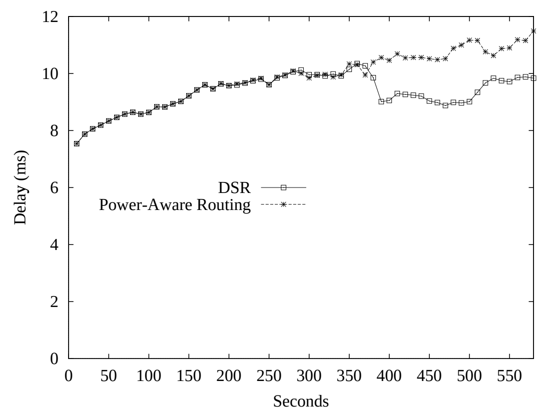
(a) Data Packet Delivery, 10 conversations



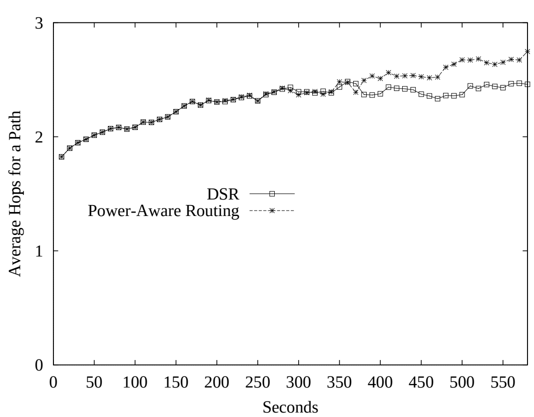
(b) Battery Level, 10 conversations



(c) Low Zone Nodes, 10 conversations

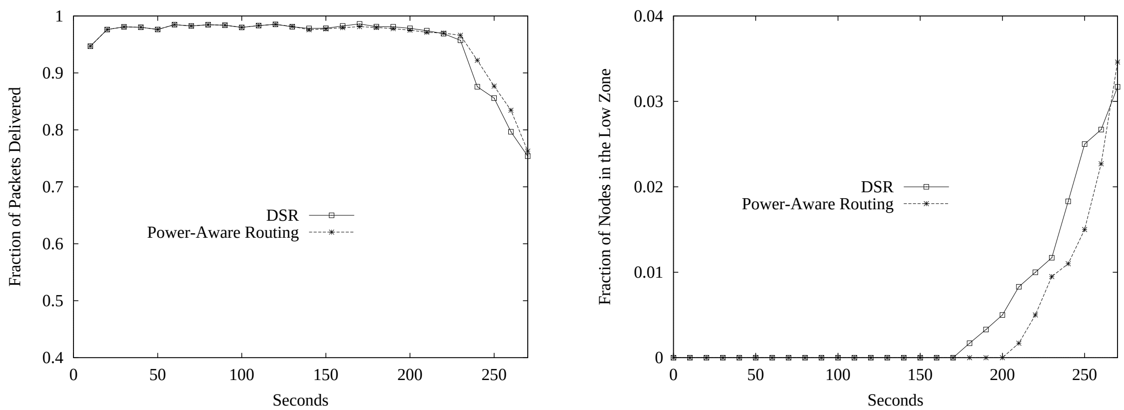


(d) Average Delay, 10 conversations



(e) Average Number of Hops, 10 conversations

Figure 5.5: Various performance metrics for 10 conversations at medium mobility.



(a) Data Packet Delivery, 20 conversations

(b) Low Zone Nodes, 20 conversations

Figure 5.6: Various performance metrics for 20 conversations at high (12 m/s) mobility.

frequent route discoveries, draining battery power of all nodes rapidly. Thus, there is little that can be done to optimize a network with heavy traffic and high mobility.

5.3 Related Work

The problem of trying to conserve battery usage within a mobile ad hoc network is not new. Previous other work has gone into different strategies on how to conserve power usage. In [9], the network longevity was the overall goal in reducing battery consumption. An algorithmic approach of a class of flow augmentation algorithms coupled with a flow redirection algorithm was used. Unlike the conventional approach of minimizing the cost of a route from a source to a given destination, their strategy was geared to balancing the battery level usage among the nodes in the network in proportion to their energy reserves.

In [64], power-aware metrics were discussed and simulated for performance evaluation. Results showed that with the use of these power-aware metrics the actual cost per packet was reduced. Though the network sizes used were small, 20 or less nodes, an interesting observation was made that in larger networks savings are higher. It should be noted that this technique was run on top of their MAC layer protocol PAMAS [63]. In PAMAS, nodes turn off their interface, when they know they cannot (because of possible multiple access interference, for example) send or receive packets. This is, in general, a useful technique for power-aware communication in any layer.

In [45], a transport layer protocol is proposed to suspend the communication device during idle periods was addressed. The tradeoff of such a method is in the choosing of the interval to shut down the communication device and when to turn it on. This will incur a delay in delivery of data packets. Experimental results based upon simulation communication patterns show that queuing of data will incur only a small delay but is dependent upon the power management level.

In [25], a performance analysis of the on-demand protocols DSR and AODV [55] were evaluated for their energy consumption. Since no attempt was made to optimize the battery usage of the nodes within the network, the primary work was to evaluate the protocols themselves during operation. One interesting observation was that flooding (the common occurrence among on-demand protocols) was very expensive in regard to the network as a whole. This was true for both protocols but more so for AODV due to its more broadcast traffic stemming from more frequent route discoveries. However, DSR with its promiscuous mode turned on also had a very high price since a node is listening to whatever it can listen to and thus will be receptive to a higher number of packets.

5.4 Concluding Remarks

On-demand routing protocols are useful for a mobile ad hoc network environment for their low routing overheads. However, if inadequate care of battery usage is not taking into consideration in their design, it may lead to premature depletion of some nodes' battery leading to network frequent network disconnects. We have proposed a power-aware technique as an extension of the DSR protocol that will react appropriately to discover alternate routes as and when a node's battery runs low. Two levels are used to denote two levels of urgency — one just issues a warning (*warning_zone* state) and the other denotes an emergency (*low_zone* state). The source nodes for the routes going via the node with low battery react appropriately by attempting to discover a new route to the destination excluding those nodes whose battery level is low.

Performance evaluation using a packet-level routing simulator shows that the longevity of a network can be extended by a significant amount. Nodes with a battery level in the the *low_zone* state is also reduced by as much as 50% using our technique. This is significant in that this allows the network to use these nodes to forward data packets or aid in finding other routes, that would die quickly in the baseline DSR protocol.

Our power-aware technique can be used for other on-demand protocols that do not use source routing. Only the nodes monitoring their own battery level need be implemented the same way as described here with WZ-PKTs and LZ-PKTs causing a response at the source as before. Our future plan is to optimize or make the searching of a new route "smarter" by having a new route consisting only of nodes with a certain adequate amount of battery level. For example, knowing that a route discovery is initiated by a WZ-PKT, then only those nodes that have a battery level of at least 5% or more than the actual *WARNING_ZONE* level will be accepted into the viable route. We feel that this should further extend the longevity of the network with still a high data packet delivery. We also plan to evaluate the effectiveness of our technique in various other mobility and traffic scenarios, and with more elaborate simulations that model all protocol layers.

Chapter 6

Conclusions

Mobile ad hoc networks are characterized by the lack of any stationary infrastructure, multihop wireless links and frequently changing network topology. Such networks are useful where a mobile network must be set up quickly without the aid of an existing infrastructure, such as military or search-and-rescue operations, emergency management and exploration missions. In addition, future commercial use is likely in the area of home- or personal-area networks where using an infrastructure may not be cost-effective.

Development of efficient dynamic routing protocols is a central challenge in ad hoc network design. Limited bandwidth of wireless networks necessitates that the routing overhead be kept low. However, this is hard to achieve as changing mobility requires that significant routing and re-routing activities are performed very frequently. Another important resource in such networks is the individual battery levels of each node that limits the useful lifetime of the network. Even a few nodes dying out quickly can significantly limit the lifetime of the network if the network gets disconnected. This is very likely to happen in a sparse network.

The work presented in this dissertation addresses the above challenges in the following way. It compares the performance of several existing routing protocols in a common framework and evaluates several useful metrics including the routing overhead. It then focuses on the routing overhead question further and develops two independent techniques to reduce the routing overheads, *viz.*, query localization and multipath routing. Finally, a routing optimization is developed that increases the lifetime of the network by making the routing protocol power-aware. We summarize the specific contributions and observations in the following.

6.1 Performance Evaluation of Routing Protocols

In the early stages of this dissertation, there was limited literature that evaluated the performance of existing routing protocols in a common framework. Our work was one of first to perform a comprehensive evaluation of a large suite of routing protocols, including protocols of various types (*viz.* proactive and reactive or on-demand) in a common simulation platform. Several useful performance metrics, such as fraction of packets delivered, delay and routing load, were evaluated across various traffic and mobility scenarios. The simulation results show that proactive protocols demonstrate excellent performance in end-to-end delay and data packet delivery albeit with very high routing overhead. Reactive (on-demand) protocols, on the other hand, showed excellent savings in routing load though less than ideal performance in end-to-end delay and data packet delivery. This was because on-demand protocols, unlike their proactive counterparts, create and maintain routes only on an “as needed” basis. This keeps the routing load down, but lets the protocol suffer a modest route discovery latency making it more prone to data packet drops and end-to-end delay.

6.2 Reduction of Routing Overhead

The routing overhead differential between proactive and on-demand protocols was overwhelmingly in favor of on-demand protocols. In a real network, routing overhead will adversely impact the delay and delivery metrics because of additional multiple access interference that our simulator did not capture because of modeling limitations. Also, more routing overhead will translate to more battery usage. This prompted us to focus on the routing overhead question further. Here we only considered on-demand protocols for their already low routing overhead. We developed two techniques to reduce routing overheads, as described below.

6.2.1 Query Localization

The major component of routing load in on-demand protocols is the query flooding for route discovery. A query localization technique is developed that limits the extent of the query flood, thus reducing the routing load. Utilizing valuable information from prior routing histories of the source-destination pairs, we show that new routes are very similar in comparison to the older, stale routes. Thus, flooding can be guided around the vicinity of these prior routes in finding new routes. Simulation results show a reduction of the routing load up to about 50% in comparison to network-wide flood and 20% in comparison to a simple query containment flood mechanism. Reduction in the routing load is also shown to positively impact delay and delivery performances.

6.2.2 Multipath Routing

In addition to controlling the extent of the query flood, it is possible to reduce the frequency of such a flood. We explore this option here by finding multiple routes per route discovery. We focus on *disjoint* routes so that these routes are less likely to fail together. Alternate routes are used when primary routes fail and thus route discovery floods can be postponed until all alternate routes break. Different multipath techniques are explored, for example, those that provide alternate routes only at the source and those that provide all nodes on the primary route with the alternate routes. Simulation results show a reduction in the number of route discoveries by approximately one-third, which contributes to a similar reduction in the routing load. We also discovered that there is little benefit in providing more than one alternate routes.

Multipath techniques also reduces the amount of dropped data packets. But there is slight increase in the end-to-end delay as alternate routes tend to be longer than primary routes. However, we consider this shortcoming to be very minor compared to the benefits of multipath routing in ad hoc networks.

6.3 On-Demand Power-Aware Routing

Finally, we focus on the question of making routing protocols aware of diminishing battery levels at the nodes so that the operational lifetime of the network can be improved. Note that reduction of the routing load also reduces battery usage. However, specific power management protocols need to be devised to offload routing responsibilities from the nodes running low on battery power. We developed a power-aware routing technique for on-demand protocols where all nodes are aware of their own battery levels. The technique works by shifting routing responsibilities towards nodes with higher battery levels from nodes with very low battery levels. This effectively increases the longevity of the network by distributing the battery levels more evenly.

Simulation results show that the longevity of the network can be extended up to 200 seconds while still delivering a high percentage of data packets. The amount of nodes whose battery power was drained to the level of not being able to help in any form of routing was reduced by 50%. Thus, the number of potential participating nodes is increased and battery levels are distributed more evenly among the network.

6.4 Future Work

A couple of research directions are immediately apparent from the above work. First, more elaborate simulators that model all protocol layers in detail will possibly discover performance issues that could not be ascertained from the work presented in this thesis. For example, the interaction of the routing load with other metrics (such as delay and packet delivery fraction) will be more severe in a detailed simulator. Some detailed simulation work is already in progress in our group or elsewhere [37, 41]. We happily note, however, that detailed simulators find far better improvements from using our routing load reduction techniques. See, for example, our recent work reported in [15]. Second, real ad hoc network testbeds need to be developed to understand performance issues further. Some efforts are underway in our research group [20].

Several additional techniques are expected to improve performance of on-demand routing protocols that are not yet explored by the research community. For example, the protocols currently gives preference to shortest route in terms of hop count and not in terms of delay. Using a delay metric is complex, as it is dynamically changing even without any change in route, and also it may give rise to route oscillations. But clearly, a delay metric will be more effective from the application's point of view. Similarly, routes can be constructed by skirting around congestion points which will also emphasize the delay metric. In addition, choice of sophisticated queuing disciplines, for example, those that will give higher priority to certain types of packets (say, error packets more than the route request packets) should be explored.

Bibliography

[1] G. Aggelou and R. Tafazolli. Rdmar: A bandwidth-efficient routing protocol for mobile ad hoc networks. In *Proceedings of ACM MobiCom99WoWMoM99,* August 1999.

[2] 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):629–638, August 1996.

[3] C. Alaettinoglu, A. U. Shankar, K. Dussa-Zieger, and I. Matta. Design and implementation of MaRS: A routing testbed. *Journal of Internetworking: Research and Experience*, 5(1):17–41, 1994.

[4] D. Bertsekas and R. Gallager. *Data Networks.* Prentice-Hall, 1987.

[5] J. Broch, D. Johnson, and D. Maltz. The dynamic source routing protocol for mobile ad hoc networks. `http://www.ietf.org/internet-drafts/ draft-ietf-manet-dsr-01.txt`, Dec 1998. IETF Internet Draft.

[6] 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 the 4th International Conference on Mobile Computing and Networking (ACM MOBICOM'98)*, pages 85–97, October 1998.

[7] Josh Broch, David Johnson, and David Maltz. The dynamic source routing protocol for mobile ad hoc networks. http://www.ietf.org/internet-drafts/ draft-ietf-manet-dsr-03.txt, Oct 1999. IETF Internet Draft (work in progress).

[8] K. Carlberg and J. Crowcroft. Building shared trees using a one-to-many joining mechanism. *ACM Computer Communication Review*, pages 5–11, January 1997.

[9] J.H. Chang and L. Tassiulas. Energy conserving routing in wireless ad-hoc networks. In *Proceedings of the IEEE Infocom The Conference on Computer Communications*, pages 22–31, March 2000.

[10] S. Chen and K. Nahrstedt. Distributed QoS routing with imprecise state information. In *7th Int. Conf. on Computer Communications and Networks (IC3N)*, pages 614–621, October 1998.

[11] C. Cheng, R. Riley, and S. P. R. Kumar. A loop-free extended Bellman-Ford routing protocol without bouncing effect. In *Proc. of ACM SIGCOMM Conf.*, pages 224–236, 1989.

[12] M.S. Corson, S. Batsell, and J. Macker. Architectural considerations for mobile mesh networking. `http://tonnant.itd.nrl.navy.mil/mmnet/ mmnetRFC.txt`, May 1996. Request for Comments Draft.

[13] M.S. Corson and A. Ephremides. A distributed routing algorithm for mobile wireless networks. *Wireless Networks*, 1(1):61–81, 1995.

[14] S. Corson, S. Papademetriou, P. Papadopoulos, V. Park, and A. Qayyum. An internet MANET encapsulation protocol (IMEP) specification. `http://www.ietf.org/internet-drafts/draft-ietf-manet-imep-spec-01.txt`, 1998. Internet Draft.

[15] R. Castañeda S.R. Das. Query localization techniques for on-demand routing protocols in ad hoc networks. *ACM/Baltzer Wireless Networks Journal*, 2000. Submitted.

[16] Samir R. Das, Charles E. Perkins, and Elizabeth M. Royer. Performance comparison of two on-demand routing protocols for ad hoc networks. In *Proceedings of the IEEE INFOCOM 2000 Conference*, March 2000.

[17] S.R. Das, R. Castañeda, J. Yan, and R. Sengupta. Comparative performance evaluation of routing protocols for mobile, ad hoc networks. In *7th Int. Conf. on Computer Communications and Networks (IC3N)*, pages 153–161, October 1998.

[18] S.R. Das, R. Castañeda, J. Yan, and R. Sengupta. Simulation based performance evaluation of routing protocols for mobile, ad hoc networks. *ACM/Baltzer Wireless Networks Journal*, 1998. Submitted.

[19] IEEE Standards Department. Wireless LAN medium access control (MAC) and physical layer (PHY) specifications, IEEE standard 802.11–1997, 1997.

[20] S. Desilva and S. R. Das. Experimental evaluation of a wireless ad hoc network. In *Proceedings of the IEEE International Conference on Computer Communication and Networks (ICCCN'00)*, October 2000.

[21] E. W. Dijkstra. A note on two problems in connection with graphs. *Numerical Mathematics*, 1:269–271, Oct. 1959.

[22] F. Douglis, P. Krishnan, and B. Marsh. Thwarting the power hungry disk. In *Proceedings of the 1994 Winter USENIX Conference*, January 1994.

[23] J. Haarsten et. al. Bluetooth: Vision, goals, and architecture. *ACM SIGMOBILE Mobile Computing and Communications Review*, 2(4):38–45, Oct 1998.

[24] K. Fall and K. Varadhan. ns notes and documentation. `http://www-mash.cs.berkeley.edu/ns/`, November 1997.

[25] L. M. Feeney. An energy consumption model for performance analysis of routing protocols for mobile ad hoc networks. `http://www.sics.se/ lmfeeney/research.html`, July 1999.

[26] B. Freisleben and R. Jansen. Analysis of routing protocols for ad hoc networks of mobile computers. In *Proceedings of the 15th IASTED International Conference on Applied Informatics*, pages 133–136, Innsbruck, Austria, 1997. Iasted-Acta Press.

[27] E. Gafni and D. Bertsekas. Distributed algorithms for generating loop-free routes in networks with frequently changing topology. *IEEE Trans. on Communication*, 29(1):11–18, January 1981.

[28] J. J. Garcia-Luna-Aceves. A unified approach to loop free routing using distance vectors or link states. In *Proc. of ACM SIGCOMM'89 Conf.*, pages 212–223, Sept. 1989.

[29] M. Gerla and J.T.-C. Tsai. Multicluster, mobile, multimedia radio networks. *Wireless Networks*, 1(3):255–265, 1995.

[30] Z. Haas. A new routing protocol for the reconfigurable wireless networks. In *Proc. of the IEEE Int. Conf. on Universal Personal Communications.*, Oct 1997.

[31] Z. J. Haas and M. R. Pearlman. The zone routing protocol (ZRP) for ad hoc networks. `http://www.ietf.org/internet-drafts/draft-ietf- manet-zone-zrp-00.txt`, 1997. IETF Internet Draft.

[32] 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*, pages 167–177, Vancouver, Sept. 1998.

[33] C. Hedrick. Routing information protocol. RFC 1058, June 1988.

[34] D. Helmbold, D.D. E. Long, and B. Sherrod. A dynamic disk spin-down technique for mobile computing. In *Second ACM International Conference on Mobile Computing and Networking (ACM MOBICOM'96)*, pages 130–142, November 1996.

[35] P. Jacquet, P. Muhlethaler, and A. Qayyum. Optimized link state routing protocol. `http://www.ietf.org/internet-drafts/ draft-ietf-manet-olsr-00.txt`, November 1998. IETF Internet Draft.

[36] M. Jiang, J. Li, and Y. C. Tay. Cluster based routing protocol (CBRP). `http://www.ietf.org/internet-drafts/ draft-ietf-manet-cbrp-spec-00.txt`, August 1998. IETF Internet Draft.

[37] P. Johansson, T. Larsson, N. Hedman, B. Mielczarek, and M. Degermark. Scenario-based performance analysis of routing protocols for mobile ad-hoc networks. In *Proceedings of the 5th International Conference on Mobile Computing and Networking (ACM MOBICOM'99)*, pages 195–206, August 1999.

[38] D. Johnson and D. Maltz. Dynamic source routing in ad hoc wireless networks. In T. Imielinski and H. Korth, editors, *Mobile computing.* Kluwer Academic, 1996.

[39] J. Jubin and J. D. Tornow. The DARPA packet radio network protocols. *Proceedings of the IEEE*, 75(1):21–32, January 1987.

[40] P. Karn. Maca - a new channel access method for packet radio. In *ARRL/CRRL Amateur Radio 9th Computer Networking Conference*, pages 134–140, 1990.

[41] B. Karp and H.T. Kung. Gpsr: Greedy perimeter stateless routing for wireless networks. In *Proceedings of the 6th International Conference on Mobile Computing and Networking (ACM MOBICOM'00)*, August 2000. Submitted.

[42] S. Keshav. *An Engineering Approach to Computer Networking : ATM Networks, the Internet, and the Telephone Network*, chapter 11. Addison-Wesley, 1997.

[43] A. Khanna and J. Zinky. A revised ARPANET routing metric. In *Proc. of the ACM SIGCOMM'89 Conf.*, pages 45–56, Sept. 1989.

[44] 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)*, pages 66–75, November 1998.

[45] R. Kravets and P. Krishnan. Power management techniques for mobile communication. In *Proceedings of the 4th International Conference on Mobile Computing and Networking (ACM MOBICOM'98)*, pages 157–168, October 1998.

[46] P. Krishna, N. H. Vaidya, M. Chatterjee, and D. K. Pradhan. A cluster-based approach for routing in dynamic networks. *ACM SIGCOMM Computer Communication Review*, pages 49–65, April 1997.

[47] J. R. Lorch and A. J. Smith. Reducing power processor consumption by improving processor time management in a single-user operating system. In *Second ACM International Conference on Mobile Computing and Networking (ACM MOBICOM'96)*, pages 143–154, November 1996.

[48] J. Macker and S. Corson. Mobile ad hoc networks (MANET). `http://www.ietf.org/html.charters/manet-charter.html`, 1997. IETF Working Group Charter.

[49] J. Moy. OSPF version 2. RFC 1247, July 1991.

[50] S. Murthy and J. J. Garcia-Luna-Aceves. An efficient routing protocol for wireless networks. *Mobile Networks and Applications*, 1(2):183–197, Oct. 1996.

[51] USCG Navigation Center GPS Page. `http://www.navcen.uscg.mil/faq/gpsfaq.htm`, April 1998.

[52] V. Park and S. Corson. A performance comparison of TORA and ideal link state routing. In *Proceedings of IEEE Symposium of Computers and Communication*, June 1998.

[53] V. Park and S. Corson. Temporally ordered routing algorithm (TORA) version 1, functional specification. `http://www.ietf.org/ internet-drafts/ draft-ietf-manet-tora-spec-01. txt`, 1998. IETF Internet Draft.

[54] V. D. Park and M. S. Corson. A highly adaptive distributed routing algorithm for mobile wireless networks. In *Proceedings of IEEE INFOCOM'97 Conf.*, pages 1405–1413, April 1997.

[55] C. Perkins. Ad hoc on demand distance vector (AODV) routing. `http://www.ietf.org/internet-drafts/ draft-ietf-manet-aodv-00.txt`, 1997. IETF Internet Draft.

[56] C. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In *Proceedings of the ACM SIGCOMM '94 Conference*, pages 234–244, August 1994.

[57] C. Perkins and E. Royer. Ad hoc on demand distance vector routing. `http://www.ietf.org/internet-drafts/ draft-ietf-manet-aodv-02.txt`, November 1998. IETF Internet Draft.

[58] Charles Perkins, Elizabeth Royer, and Samir Das. Ad hoc on demand distance vector (AODV) routing. http://www.ietf.org/internet-drafts/ draft-ietf-manet-aodv-04.txt, Oct 1999. IETF Internet Draft (work in progress).

[59] Y. Rekhter and T. Li. A Border Gateway Protocol-4. RFC 827, March 1995.

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

[61] 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*, 6:59–87, 1995. (Preliminary version appeared in Proc. ACM SIGMETRICS/PERFORMANCE Conf., 1992, pages 181-192.).

[62] S. Singh and C. S. Raghavendra. PAMAS - power aware multi-access protocol with signalling for ad hoc networks. In *ACM Computer Communication Review (ACM CCR'98)*, July 1998.

[63] S. Singh and C. S. Raghavendra. PAMAS - power aware multi-access protocol with signalling for ad hoc networks. In *ACM Computer Communication Review (ACM CCR'98)*, July 1998.

[64] S. Singh, M. Woo, and C. S. Raghavendra. Power-aware routing in mobile ad hoc networks. In *Proceedings of the 4th International Conference on Mobile Computing and Networking (ACM MOBICOM'98)*, pages 181–190, October 1998.

[65] R. Sivakumar, P. Sinha, and V. Bharghavan. Core extraction distributed ad hoc routing (CEDAR) specification. `http://www.ietf.org/internet-drafts/draft-ietf-manet-zone-cedar-00.txt`, October 1998. IETF Internet Draft.

[66] M. Stemm, P. Gauthier, D. Harada, and R. Katz. Reducing power consumption of network interfaces in hand-held devices. In *Third International Workshop on Mobile Multimedia Communication (MoMuc-3'96)*, December 1996.

[67] M. Stemm and R. Katz. Reducing power consumption of network interfaces in hand-held devices. In *Third International Workshop on Mobile Multimedia Communication (MoMuc-3'96)*, December 1996.

[68] M. Weiser, B. Welch, A. Demers, and S. Shenker. Scheduling for reduced cpu energy. In *Proceedings fo the First Symposium on Operating System Design and Implementation (OSDI'94)*, November 1994.

[69] W.T. Zaumen and J. J. Garcia-Luna-Aceves. Loop-free multipath routing using generalized diffusing computations. In *Proceedings of the IEEE INFOCOM'98*, pages 1408–1417, March 1998.

Vita

Robert Castañeda was born in San Antonio, Texas in the country of the United States of America on March 16, 1963. He is the third child of Henry G. Castañeda and Romana Castañeda. He grew up in San Antonio and attended West Avenue Elementary School, Nimitz Middle School, and graduated from Robert E. Lee High School in May 1981. For his higher level of education, he attended the University of Texas at San Antonio where he obtained a Bachelor's Degree of Science in Computer Science in December 1990. He later pursued graduate school there and earned a Master's Degree of Science in Computer Science in December 1994. His research collectively lead to a Master's Thesis entitled *"Evaluating Hot Spot Effects and Mutual Exclusion Algorithm Performance on Shared Memory Multiprocessors"* under the supervision of Dr. Xiaodong Zhang. His research also lead to the award *University Life Award for College of Science and Engineering* that was presented to him in April 1994. After graduation he became a Research Assistant under Dr. Zhang to further his research skills. With the advent of the new Computer Science Ph.D program established in the Fall of 1995 at the University of Texas at San Antonio, he again became a student in his quest for his doctoral degree. Under the guidance of his advisor, Dr. Samir R. Das, his research lead to his dissertation entitled *"Protocols for Mobile Ad Hoc Networking"*. He successfully defended his dissertation on August 25, 2000 and formally earned his Doctorate degree in December, 2000. Upon earning his Doctor of Philosophy in Computer Science his interest have turned to industry to pursue a career within the field of wireless data networking. Currently he has one son, Robert Gabriel Castañeda, born July 16, 1986 in San Antonio, Texas.

Teaching

University of Texas at San Antonio, 1997-2000.

Publications stemming from this dissertation are annotated here.

Publications

Journals

"Query Localization Techniques for On-demand Routing Protocols in Ad Hoc Networks", *ACM/Baltzer Wireless Networks*, 2000. (with Samir R. Das and Mahesh K. Marina).

"Performance of Multipath Routing for On-Demand Protocols in Mobile Ad hoc Networks", *Wireless Multicast and Routing in MONET*, 2000. (with Asis Nasipuri and Samir R. Das).

"Simulation Based Performance Evaluation of Mobile, Ad hoc Network Routing Protocols", *ACM/Baltzer Wireless Networks*, 1999. (with Samir R. Das and Jiangtao Yan).

Conferences

"A Power-Aware Routing Technique for On-Demand Routing Protocols in Mobile Ad Hoc Networks", submitted for publication (with Samir R. Das).

"Query Localization Techniques for On-demand Routing Protocols in Ad Hoc Networks", *The Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM)*, August, 1999. (with Samir R. Das).

"Comparative Performance Evaluation of Routing Protocols for Mobile Ad Hoc Networks", *Proceedings of the Seventh International Conference on Computer Communications and Networks (IC3N)*, October, 1998. (with Samir R. Das, Jiangtao Yan, and Rimli Sengupta).

Other publications from previous research are annotated here.

Publications

Journals

"Execution Complexities and Performance of Software Mutual Exclusion Algorithms on Shared-Memory Multiprocessors", *IEEE Parallel & Distributed Technology*, Spring Issue, 1996, pp. 25-42. (with Xiaodong Zhang and Yong Yan)

"Comparative Performance Evaluation of Hot Spot Contention Between MIN-based and Ring-based Shared-Memory Architectures", *IEEE Transactions on Parallel and Distributed Systems*, Vol. 6, No. 8, 1995, pp. 872-886. (with Xiaodong Zhang and Yong Yan).

"Spin-Lock Synchronization on the Butterfly and KSR-1", *IEEE Parallel and Distributed Technology*, Vol. 2, Spring Issue, 1994, pp. 51-63. (with Xiaodong Zhang and Elisa W. Chan).

Conferences

"A Comparative Evaluation of Hierarchical Network Architecture of the HP-Convex Exemplar", *Proceedings of IEEE International Conference on Computer Design*, October, 1997. (with Xiaodong Zhang and James M. Hoover).

"Modeling and measuring hot spots on MIN-based and HR-based shared-memory architectures", *Proceedings of the Fifth IEEE Symposium on Parallel and Distributed Processing (SPDP'93)*, IEEE Computer Society Press, December, 1993. (with Xiaodong Zhang and Yong Yan).

Permanent address: 638 Edgebrook Ln
San Antonio, TX 78213-4514

This dissertation was typeset by Robert Castañeda in LaTeX.