Preliminary version of paper to be presented at High Performance Computing '99

An Analysis of Routing Techniques for Mobile and Ad Hoc Networks

Rajendra V. Boppana, Mahesh K. Marina and Satyadeva P. Konduru
Computer Science Division
The University of Texas at San Antonio
San Antonio, TX 78249-0667
{boppana,mmarina,skonduru}@cs.utsa.edu

Abstract. On demand routing algorithms such as the dynamic source routing (DSR) and ad hoc on-demand distance vector (AODV) have received much attention recently and are being promoted as better alternatives to the currently used distance vector algorithms such as the destination sequenced distance vector (DSDV). In this paper, we provide a variation of DSDV and compare it with AODV and DSR in their original proposed forms and with some of the optimizations turned off. While on-demand algorithms are likely to react well to transient conditions such as high node mobility in combination with low network load, they are unlikely to provide the best performance under heavy network loads.

1 Introduction

A mobile and ad hoc network (MANET) facilitates mobile hosts such as laptops with wireless radio networks communicate among themselves even when there is no wired network infrastructure. Currently, there are no major commercial applications which require such impromptu networking capabilities, but there is a real need for MANETs in military situations. In a MANET, most hosts, if not all, are assumed to be moving continually and thus do not have a default router or fixed set of neighbors. So each mobile host should have an Internet Protocol (IP) routing algorithm for building and maintaining routing tables, just like an internet router node. In this paper we analyze some of the most promising IP routing algorithms proposed for MANETs in recent literature for possible sources of overheads and optimizations.

MANETs are made up of shared wireless links or channels, which have low bandwidth and are unreliable owing to external noise and relative movement of nodes. The commercially available WaveLAN [19] is based on the IEEE 802.11 medium access control (MAC) standard [11] for wireless LANs, and provides shared wireless channels with 2 Mb/s bandwidth and a typical radio range of 250m in open field. Since the channels are shared, packet transmissions of two or more neighbors may collide with one another and render the transmissions useless. Using a collision avoidance scheme and handshaking with request-to-send/clear-to-send (RTS/CTS) exchanges, it is feasible to provide fairly reliable unicast communication between neighbors. (By reliability here we mean that the sender can be sure if its packet has been received by the intended receiver correctly or not.) However, broadcasts on a wireless shared channel are unreliable: the sender does not know which, if any, of its neighbors received its broadcast. An advantage of the 802.11 MAC protocol is the RTS/CTS exchange can be used to detect if a neighbor is lost and report the same to the routing algorithm in the network layer. This form of link layer feedback is not available with the most commonly used Ethernet or 802.3 MAC protocol for wired LANs.

Compared to the commonly used Internet routing algorithms, the routing algorithms for MANETs must cope with the following challenges.

- Lack of default router: the routing algorithm should be simple and efficient in its resource usage so that even inexpensive and low-powered mobile nodes can run it.
- Frequent changes in network topology due to both node mobility and temporary losses of wireless channels: the routing algorithm must be highly adaptive to mobility and topology changes so that it provides predictable performance under transient and steady-state conditions.
- Low bandwidth channels: the routing algorithm should minimize the number of hops taken by packets, and minimize overhead in collecting and disseminating routing information with other nodes.
- Unreliable broadcasts: since a broadcast is the primary mechanism by which route discovery and propagation of routing information is done, the routing algorithm must be robust and should not display erratic behavior if packets carrying routing information are lost.

The traditional Internet routing algorithms have too much overhead and are optimized for wired networks. The most commonly used routing algorithms are the open shortest-path first (OSPF)[13], which is based on the link-state (LS) algorithm, and routing information protocol (RIP)[10], based on the distance vector (DV) algorithm.

Of these, the OSPF has even more overhead than DV-based algorithms and will not be considered in this paper. We consider a DV-based algorithm called the destination-sequenced distance vector (DSDV) proposed by Perkins and Bhagwat [15].

The LS and DV based algorithms are classified as pro-active routing algorithms, since they build and maintain routing information about all the nodes in a network by propagating routing information packets even when there are no changes in routing paths or the paths are unused. Since the channel bandwidth is at a premium, many researchers proposed on-demand routing algorithms [16, 12, 14, 9, 18]. The on-demand routing algorithms build or maintain only the routing paths that have changed and are needed to send the data packets currently in the network.

In this paper, we investigate the performance of the ad hoc on-demand distance vector (AODV) and the dynamic source routing (DSR) algorithms [16, 12], which have received extensive attention from many independent researchers. We also propose an adaptive version of DSDV algorithm and compare its performance with the two on-demand algorithms. While our analysis confirms some of the previously seen claims [1, 6] for AODV and DSR, it shows weaknesses of the two algorithms under a variety of traffic and mobility conditions. Based on this analysis, we describe features that are likely to work well and those that cause high overhead.

The rest of the paper is organized as follows. Section 2 will describe the basics and unique features of the algorithms considered in this study. Section 3 provides an analysis of the algorithms. Section 4 concludes the paper.

2 Routing Algorithms

We consider mainly three algorithms in this study: DSDV as an example of pro-active algorithms, and AODV and DSR as examples of two distinct approaches in designing on-demand algorithms. In addition, we considered a few variations of these algorithms. In particular, we propose an adaptive DSDV (or ADSDV) algorithm that retains the basic characteristics of DSDV but reduces routing overhead, while improving the performance. Each of the three main algorithms and their variants considered are described below.

2.1 Destination-sequenced distance vector (DSDV)

In the DV algorithms, each node maintains a table of routing entries, with each entry indicating destination node number, number of hops required (distance) to reach the destination, and the next node in the path to the destination. In the beginning, the routing tables do not contain valid (finite distance) entries to all nodes in the network. At periodic intervals, nodes broadcast their routing tables to their neighbors. A node incorporates any routing table information received from its neighbor into its routing table by comparing the entries and picking the routes with better hop counts. These periodic updates to routing entries are done for all paths, even for those that may not be in use. In addition, when a node detects the loss of a neighbor, it invalidates (by setting the hop metric to infinity) all of its routing entries that have the lost neighbor as the next node and broadcasts the changed entries to its neighbors. These triggered updates to routing tables may result in invalidation of routing entries in its neighbors and additional triggered updates by them. The major problem with the DV based algorithms is propagation of fallacious routing information among nodes, which leads to loops in routing paths. There have been many solutions proposed to avoid this problem [2, 8, 15].

DSDV [15] solves the looping problem by attaching sequence numbers to routing entries. A node broadcasts an even sequence number that is higher than the one used before along with its periodic updates and triggered updates resulting from losses of one or more of its neighbors. Any node that invalidates its entry to a destination because of loss of next hop node, will increment the sequence number (which becomes an odd sequence number) and uses the new sequence number in its triggered updates or broadcasts. A node invalidates or modifies its routing entry if a neighbor broadcasts a routing entry to the same destination with a higher sequence number. An invalidated entry can only become valid by the routing information propagated by the destination node with the next even sequence number.

The routing table entries in all the nodes for a given destination collectively specify a *virtual destination-based tree* to send packets to that destination. A simplistic view of DSDV may be that it maintains such one destination tree for each node in a distributed manner.

With 15-second intervals [15, 1, 3], the periodic updates do not cause that much overhead. It is the triggered updates that consume a lot of channel bandwidth and, worse, they invalidate too many routing entries needlessly. To see this, consider the virtual destination-tree for some node, say x. When a node, say y, loses its path to x, it invalidates its path to x and propagates this information to its neighbors, who duly invalidate their entries to x without regard to whether they need to use y to reach x or not. Such route

invalidations can destroy the part of the destination tree that is unaffected by y's loss of path to x, and cause needless triggered updates.

We modify DSDV to limit the impact of triggered updates. Now, a node invalidates its routing entry based on a neighbor's update only if the neighbor's entry has a higher and odd sequence number and the neighbor is currently the specified next hop. A routing entry may be modified based on that of neighbor's if the neighbor has the next higher even sequence number or better metric with the same sequence number. (It is noteworthy that only valid routing entries have even sequence numbers.) Also, invalidations or changes to routing entries do not trigger an update. However, a node performs a triggered update if more than a specified number of packets are queued up at a node. This reduces the routing overhead substantially for high traffic loads and high node mobility cases. To cope up with the reduced number of triggered updates, we decreased the periodic update interval to 5 seconds from 15 seconds in the original DSDV. We call the modified algorithm the adaptive destination-sequenced distance vector (ADSDV) algorithm.

2.2 Ad hoc on-demand distance vector (AODV)

AODV builds and maintains routing entries containing next hop, distance, and destination sequence number. Unlike DSDV, however, it does not use periodic or triggered updates to disseminate routing information. Routing entries are built using route discovery techniques. When a node needs to send a packet to a destination to which it does not have a routing entry, it floods the network with a route request (RREQ) packet. Nodes receiving RREQs set up reverse paths to sources of RREQs in their routing tables, and either reply to the RREQ if they already have an entry for the destination in question or forward the RREQ. In the worst-case, the destination will reply, and the source may receive more than one reply to its RREQ. An existing route entry may be invalidated if it is unused within the specified time interval (active route timeout period) or the next hop node is no longer a viable node to reach the destination. In that case, the invalidation is propagated to neighbors that have used this node as their next hop. AODV requires the neighbors to exchange hello messages periodically or feedback from the link layer when a loss of a neighbor is detected.

The main sources of overhead in AODV are the route requests, replies, and invalidation packets used for route discovery, and the hello packets if link layer feedback is not used. Also, the active route timeout period affects both the routing overhead and performance. Broch et al. [3] used 300-second periods, while the new specification of AODV recommends 3-second active timeout periods. More recently, Das [5] has modified the route discovery using the expanding rings approach (RREQs are sent initially with hop limit of 1, then with a 2-hop limit if no reply is received, and so on). This approach can improve AODV's performance for low traffic loads dramatically. The original AODV permits an intermediate node to drop a packet if its routing entry is invalid. It is possible, however, to improve the performance by letting intermediate nodes do route discoveries in such situations. We have experimented with the expanding rings approach and found it to be beneficial at low traffic loads, but not at high loads. We have not evaluated the impact of the second optimization.

2.3 Dynamic source routing (DSR)

A routing entry in DSR contains all the intermediate nodes to be visited by a packet rather than just the next hop information maintained in DSDV and AODV. A source puts the entire routing path in the data packet, and the packet is sent through the intermediate nodes specified in the path (similar to the IP strict source routing option [4]). If the source does not have a routing path the destination, then it performs a route discovery by flooding the network with a route request (RREQ) packet. The RREQs record route information as they visit intermediate nodes on the way to the destination. Any node that has a path to the destination in question can reply to the RREQ packet. The reply is sent using the route recorded in the RREQ packet. A node that receives a RREQ can use the path recorded to improve its path to the source.

To limit the need for route discovery, DSR also allows nodes to operate their network interfaces in promiscuous mode and snoop all (including data) packets sent by their neighbors. Since complete paths are indicated in data packets, snooping can be very helpful in keeping the paths fresh. So DSR attempts to exploit the shared channels better than AODV or DSDV. To further reduce the cost of route discovery, the RREQs are initially broadcasted to neighbors only (1-hop limit), and then to the entire network if no reply is received. Another optimization feasible with DSR is the gratuitous route replies; when a node overhears a packet containing its address in the unused portion of the path in the packet header, it sends the shorter path information to the source of the packet. Also, an intermediate node may replace a packet's current path specification with an alternate path if it is unable to send the packet to the original next hop node.

DSR specification and implementation [3] contains the

above mentioned optimizations. Of these, the route replacement and gratuitous route replies are specific to DSR. But the expanding rings (or controlled route discovery) can be applied to AODV with suitable modifications. Another optimization specific to DSR is snooping and learning better paths. This requires a mobile node to continually process all packets being transmitted within its listening area. This may not be a feasible optimization for low-powered devices. To evaluate the effect of this optimization, we simulated two additional variants of DSR. In one variation, no snooping is allowed (non-snooping DSR or NSDSR) and in another a node is allowed to snoop only when it is doing route discovery (selective snooping DSR or SSDSR).

2.4 Other algorithms

There are many other routing algorithms proposed for MANETs. The temporally ordered routing algorithm (TORA) [14] maintains multiple routes to destinations to cope up with changes in network topologies. However, this algorithm is sensitive to routing packet losses and has been shown to perform poorly compared to AODV and DSR in previous studies [1, 6]. The zone routing protocol (ZRP) views a MANET as a collection of zones or clusters and uses a pro-active protocol within a zone. Each zone has a zone leader which also runs a reactive protocol for maintaining routing tables and information across the zones. An opposite approach is taken by the CEDAR algorithm [18], in which a collection of high bandwidth links form a spine and a pro-active algorithm is run among the nodes connected to the spine, and a reactive algorithm to reach all other nodes.

3 Performance Analysis

We have used the NS-2 network simulator [7] with the CMU extensions by Johnson et al. [3] for our analysis. The CMU extensions include implementations of DSDV, AODV, and DSR. The various parameter, timeout and threshold values and optimizations used are exactly as described in the paper by Broch et al. [1]. For the sake of completeness, we outline the most relevant parameters for these algorithms.

For DSDV, the periodic update interval is 15 seconds. DSDV implementation [3] does not use the link-layer feedback, since this makes triggered updates too frequent. Instead, a node assumes a neighbor is lost if it does not hear the neighbor's periodic update in 45 seconds. Also, DSDV uses 5 buffers per destination in each node. For the ADSDV we proposed, the periodic update interval is 5 seconds. Link layer feedback is used to determine lost neighbors and to invalidate appropriate paths. An update is triggered if a node has five or more packets waiting in its queues. A node invalidates a routing entry based on a neighbor's invalidated routing entry only if the that node is the next hop node. All other parameters are exactly the same as the ones used for DSDV.

The CMU implementation of AODV uses 300-second active route timeouts and link layer feedbacks. We have modified the active timeout value to 3 seconds, since the newer specification of AODV [17] recommends it and it seems to give better performance. These two versions are called AODV300 and AODV3, respectively. All other parameters are the same as given in [1]. AODV implementations flood the network with RREQ packets and packets reaching an intermediate node with no valid entry for the destination are dropped. These are obvious sources of overheads and performance loss that can be minimized with careful optimizations.

DSR implementation has controlled propagation of RREQs, promiscuous operation of network interface to facilitate snooping all packets in the listening area, gratuitous route replies to improve paths, route replacement to repair broken paths by intermediate nodes, and a secondary route storage to keep multiple paths to destinations. In addition to this, we have simulated the non-snooping DSR (NSDSR) and selective snooping DSR (SSDSR), which differ from the original DSR only in their snooping capabilities.

Network and mobility model. We have simulated a network of 50 nodes randomly placed on a field of 1500m x 300m at the beginning of a simulation. Each node moves in a randomly chosen direction at an average speed of 1 m/s, uniformly varied between 0-2 ms/, or 10 m/s, varied between 0-20 m/s. This is to see the impact of node mobility on routing protocol performance. We simulated only continuous mobility (that is, no pause times).

A wireless channel has 2 Mb/s bandwidth and a circular radio range with 250 meters radius. The CMU implementation has modeled the IEEE 802.11 wireless LAN with distributed coordinated function (DCF). Neighbors exchange unicast packets using RTS/CTS exchanges. All algorithms except the original DSDV use the link-layer feedback to speedup detection of loss of neighbors and avoid using hello messages. Broadcasts are unacknowledged.

We have simulated constant bit rate (CBR) traffic with 20 connections and 40 connections. In each connection, the source sends 64-byte data packets at an average rate of 0.25-8 packets/second. With 20 connections, only a part of the network is used, and with 40 connections, almost all

of the network is used. Each simulation is started cold and run for 1000 seconds. This sort of simulation shows the impact of transient conditions on the performance, when the traffic load is low (<20 Kb/s). For high (>60 Kb/s) traffic loads, such a simulation captures mainly the steady-state behavior of a routing algorithm.

Performance metrics. We use the average data packet latency (the time it takes for a data packet to reach its destination from the time it is generated at the source), which includes all the queuing and protocol processing delays in addition to propagation and transmission delays. We also give the network throughput (total number of data bits delivered) in Kb/s. To study the overheads of various routing algorithms, we plot routing information packets transmitted per second and overhead bits/second. The overhead bits/s gives the bits transmitted as routing packets and source routing information (for DSR). All the metrics are plotted with respect to offered (data) load in Kb/s.

Performance of AODV algorithms. Figure 1 gives the average packet latencies and routing overheads with AODV3 and AODV300 for 2 m/s and 20 m/s top node speeds and 20 CBR connections. For light traffic conditions, AODV300 has better latency and overhead. However, for moderate loads and high node mobility, the AODV3 outperforms AODV300. With both algorithms, the overhead increases rapidly with higher network load.

Performance of DSDV algorithms. Figure 1 gives the latencies and routing overheads for ADSDV and DSDV. For low node mobility, ADSDV has better latency and overhead than for low network loads. However in the high node mobility case, for low to moderate network loads DSDV has better latency and overhead than ADSDV. As the load on the network increases, ADSDV gives much improved latency and overhead than DSDV regardless of the mobility. ADSDV has more predictable routing overhead compared to DSDV, especially for the high node mobility case. The big jump in the overhead by DSDV at about 20 Kb/s network load explains why it has difficulty in sustaining throughput at high network loads. The triggered updates consume a lot of the network bandwidth.

Performance of DSR algorithms. Figure 1 gives the latencies and routing overheads for the original DSR algorithm with all the optimizations turned on and the non-snooping and selective snooping DSRs. The latencies vary very widely with traffic load. Essentially DSR and its variants use a lot of optimizations which lead to unpredictable behavior under transient conditions. At low traffic loads, data packets arrive at nodes infrequently, and most of the optimizations are done using stale routing information. Purging stale routes frequently will make DSR's performance more predictable. At moderate loads however, a clear trend can be seen. SSDSR and NSDSR perform substantially worse than the original DSR for high node mobility and moderate traffic loads. This difference in performance increases with higher traffic load. The overheads are higher when snooping is restricted or prohibited. A noteworthy point is selective snooping is only slightly better than not snooping at all. Since at most 40% of nodes send packets and thus can snoop while they do route discoveries, selective snooping is not very effective. If more nodes send packets, selective snooping can provide performance closer to that of DSR. Also, snooping is counterproductive at low traffic loads, since a lot of routing information becomes stale and causes more harm than good.

Comparison of ADSDV, AODV3, and DSR. To see how the best ones from each group of algorithms compare with one another, we have conducted additional simulations with 40 CBR connections and data rates ranging from 1-8 packets/second (or approximately, 20-160 Kb/s).

Consider the low node mobility case (top node speed of 2 m/s). All three algorithms have comparable latencies and throughputs as shown in Figure 2. ADSDV provides much higher throughput; 45% more than AODV and 15% more than DSR. Also, when the network load is increased beyond saturation, ADSDV exhibits more graceful performance degradation by sustaining its throughput. The overheads incurred by AODV and DSR are comparable (see Figure 3), and rise rapidly with network load. AODV's overhead shoots up once the saturation point is reached, while DSR's overhead goes down beyond the point of saturation. In contrast, ADSDV starts with high overhead (in bits/sec) and increases less rapidly with network load. In terms of routing packets/second, ADSDV is competitive with DSR and is better than AODV.

Now consider the high node mobility case. Once again ADSDV outperforms both DSR and AODV, now by a wider margin. While AODV has lower latency for moderate network load, ADSDV is clearly the more stable algorithm for high network traffic (see Figure 4). ADSDV provides 35-43% higher throughput compared to both AODV and DSR. Furthermore, the routing overhead (Figure 5) is much less in packets/second. ADSDV has higher overhead in bits/sec, since its routing updates are much larger than route requests (RREQs) and route replies (RREPs) used in AODV and DSR. (Though DSR may seem to have less overhead in bits/sec at loads above 80 Kb/s, it is already in saturation and is dropping a lot of data packets at these

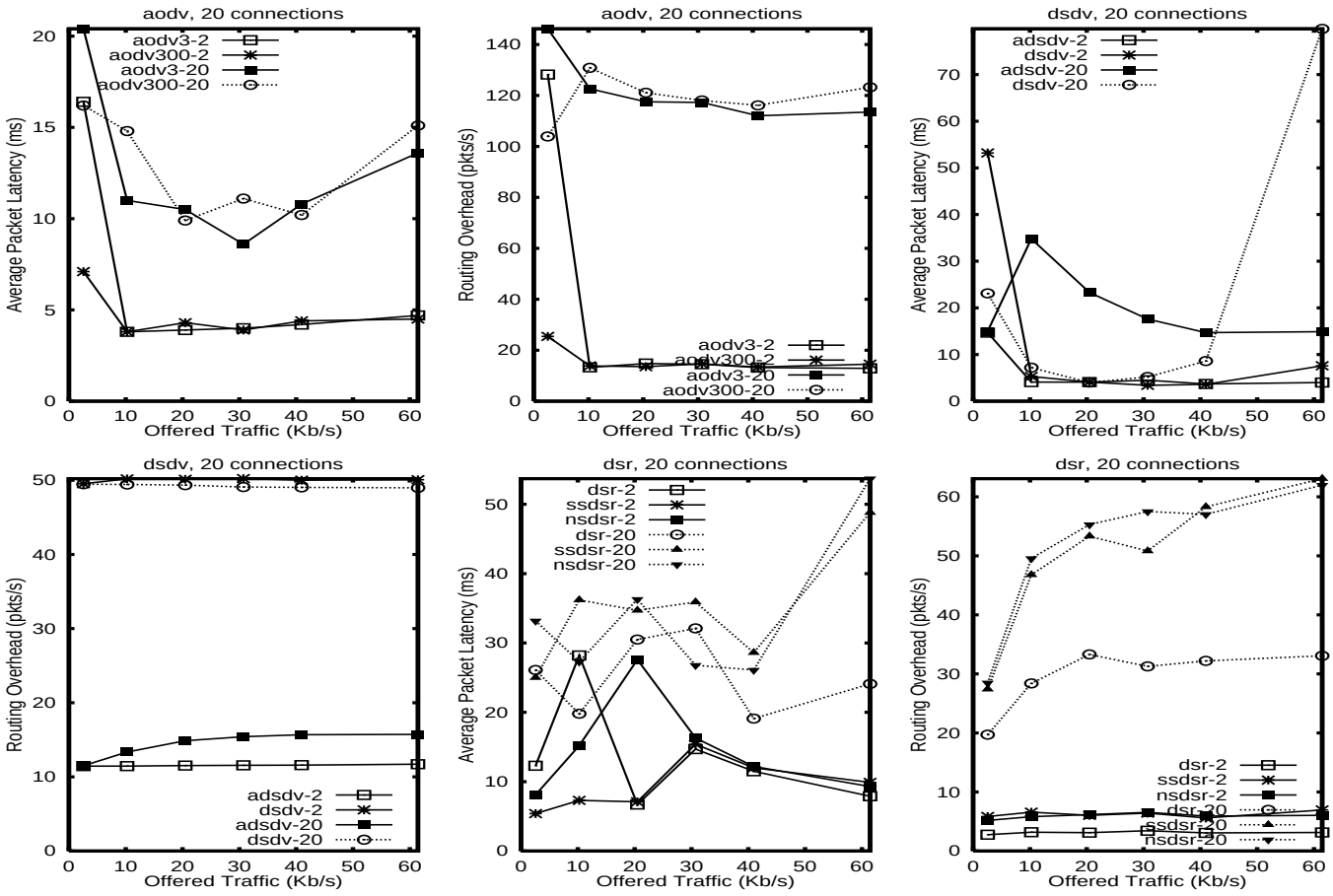


Figure 1: Data packet latencies and overheads of the original and variants of AODV, DSDV, DSR, algorithms for various traffic loads at low and high speeds. The number after the hyphen indicates the top speed by the mobile nodes.

loads.)

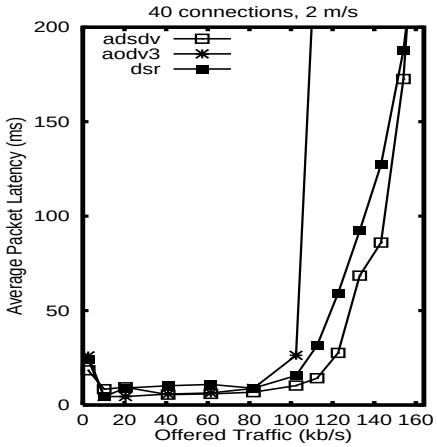
4 Conclusions

We have presented an adaptive form of the well-known DSDV algorithm. This is still a pro-active algorithm, since it depends mainly on periodic updates and controlled triggered updates to gather and disseminate routing information. It does not use route discoveries to construct needed routes. The proposed ADSDV seems to have superior performance to DSDV as the load on the network increases for both low and high node mobility cases.

DSR incorporates too many optimizations. One of them, snooping data packets seems to be counter productive under transient network conditions (low load and high mobility). Overall, DSR is more stable and has lower overhead than AODV. AODV has better latencies than DSR and ADSDV for low traffic load, but has higher overhead when node mobility is high. The CMU implementation of AODV is being revised by Das [5], to incorporate some of the optimizations used in DSR. With these optimizations, AODV may perform as well as DSR for high traffic loads.

Compared to AODV and DSR, ADSDV performs poorly when the network load is light and node mobility is high. The main reason is the on-demand algorithms use route discovery to quickly learn paths, while ADSDV builds routes over a period of time. It seems preferable to use some form of route discovery with ADSDV to improve its performance at low loads. When the network is really stressed, ADSDV outperforms both AODV and DSR. Furthermore, it exhibits graceful degradation of performance when the network is overloaded (beyond the point of saturation). This seems to suggest that carefully designed proactive algorithms may be suitable or even preferable to on-demand techniques for routing in MANETs.

In future, we plan to augment ADSDV with route discovery and compare it with DSR, AODV and other routing algorithms.



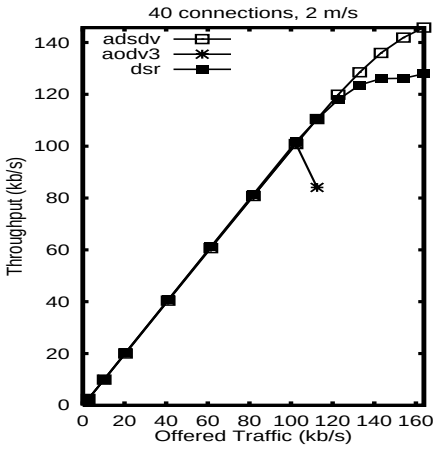
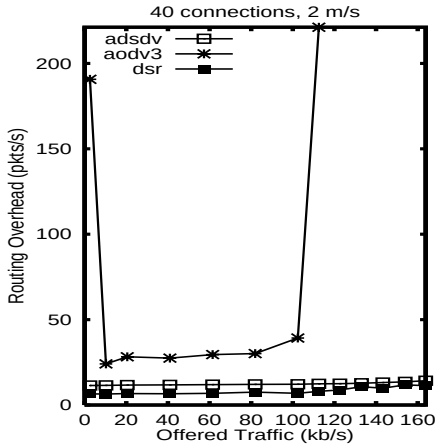


Figure 2: Data packet latencies and throughputs achieved with ADSDV, AODV3 and DSR for the low node mobility case.



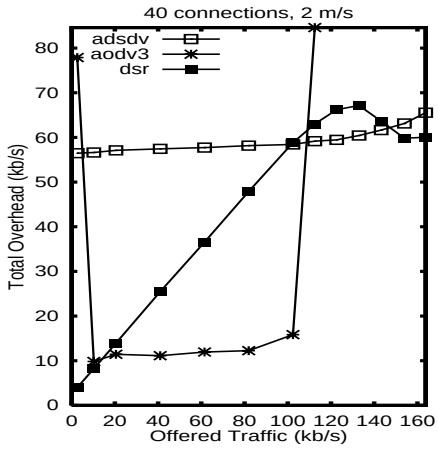


Figure 3: Overheads of ADSDV, AODV3 and DSR for the low node mobility case.

Acknowledgements

This research has been partially supported by DOD/AFOSR grant F49620-96-1-0472 and NSF grant CDA 9633299. Marina and Konduru have also been supported by graduate fellowships from the University of Texas at San Antonio. The authors are grateful to the UC Berkeley/VINT project and the Monarch group at the Carnegie Mellon University for providing the simulator used in this study. The authors would like to thank Samir Das for many stimulating discussions during the course of this work.

References

[1] J. Broch et al., "A performance comparison of multi-hop wireless ad hoc network routing protocols," in *ACM Mobicom '98*, Oct. 1998.

[2] C. Cheng, R. Riley, and S. P. R. Kumar, "A loop-free extended Bellman-Ford routing protocol without bouncing effect," in *ACM SIGCOMM '89*, pp. 224–236, 1989.

[3] CMU Monarch Group, "CMU Monarch extensions to the NS-2 simulator." Available from http://monarch.cs.cmu.edu/cmu-ns.html, 1998.

[4] D. E. Comer, ed., *Internetworking with TCP/IP: volume 1*. Prentice Hall, Inc., 1995.

[5] S. R. Das, "Optimized AODV implementation in NS-2 with CMU Monarch extensions." Personal communication, 1999.

[6] S. R. Das, R. Castaneda, and J. Yan, "Simulation based performance evaluation of mobile, ad hoc network routing protocols," in *Seventh Int'l Conf. on Computer Communication and Networks*, Oct. 1998.

[7] K. Fall and K. Varadhan, "NS notes and documentation." The VINT Project, UC Berkeley, LBL, USC/ISI, and Xerox PARC. Available from http://www-mash.cs.berkeley.edu/ns, Nov. 1997.

[8] J. J. Garcia-Luna-Aceves, "A unified approach to loop-free routing using distance vectors or link states," in *ACM SIGCOMM '89*, pp. 212–223, 1989.

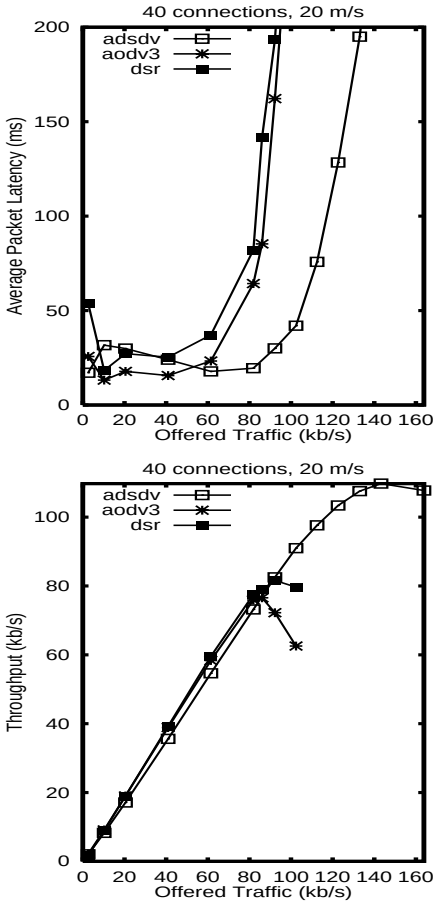


Figure 4: Data packet latencies and throughputs achieved with ADSDV, AODV3 and DSR for the high node mobility case.

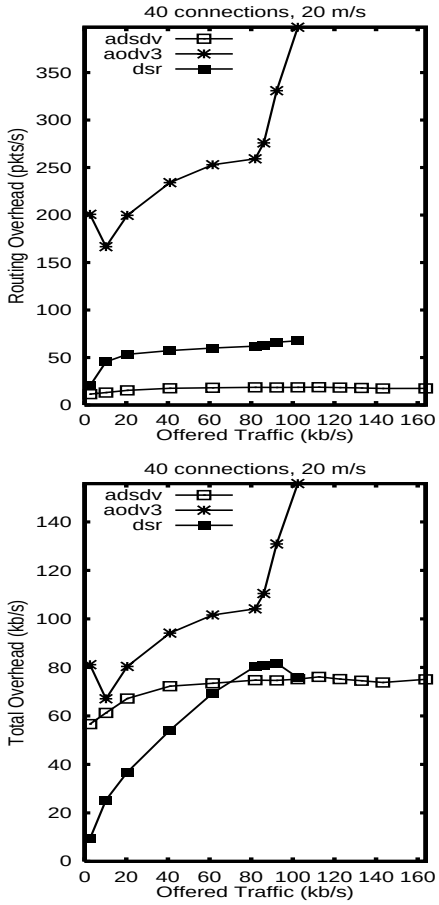


Figure 5: Overheads of ADSDV, AODV3 and DSR for the high node mobility case.

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

[10] C. Hedrick, "Routing information protocol." RFC 1058, 1988.

[11] IEEE Computer Society LAN/MAN Standards Committee, "Wireless LAN medium access control (MAC) and physical layer (PHY) specifications." IEEE Std. 802.11-1997. IEEE, New York, NY 1997.

[12] D. B. Johnson et al., "The dynamic source routing protocol for mobile adhoc networks." IETF Internet Draft. http://www.ietf.org/internet-drafts/draft-ietf-manet-dsr-02.txt, 1999.

[13] J. Moy, "Ospf version 2." RFC 1247, July 1991.

[14] V. D. Park and S. Corson, "A highly adaptive distributed routing algorithm for mobile wireless networks," in *IEEE INFOCOM '97*, pp. 1405–1413, Apr. 1997.

[15] C. E. Perins and P. Bhagwat, "Highly dynamic destination-sequenced distance vector (DSDV) for mobile computers," in *ACM SIGCOMM '94*, pp. 234–244, Aug. 1994.

[16] C. E. Perkins, "Ad hoc on demand distance vector routing." IETF Internet Draft. http://www.ietf.org/internet-drafts/draft-ietf-manet-aodv-02.txt, 1997.

[17] C. E. Perkins, E. M. Royer, and S. R. Das, "Ad hoc on demand distance vector routing." IETF Internet Draft. http://www.ietf.org/internet-drafts/draft-ietf-manet-aodv-03.txt, 1999.

[18] R. Sivakumar et al., "Core extraction distributed ad hoc routing (CEDAR) specification." IETF Internet Draft. http://www.ietf.org/internet-drafts/draft-ietf-manet-cedar-spec-00.txt, 1997.

[19] B. Tuch, "Development of WaveLAN, and ISM band wireless LAN," *AT&T Technical Journal*, vol. 72, pp. 27–33, July 1993.