An Analysis of Routing Techniques for Mobile and Ad Hoc Networks

R. V. Boppana, M. K. Marina and S. P. Konduru
CS Division, The University of Texas at San Antonio
San Antonio, TX 78249-0667
{`boppana,mmarina,skonduru`}`@cs.utsa.edu`

Abstract. In this paper, we compare the performance of two on-demand and one pro-active algorithms for multi-hop, ad hoc networks and variants of the algorithms. We show that on-demand algorithms react well to transient conditions such as high node mobility in combination with low network load, but 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. 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 characterized by shared wireless links or channels, which have low bandwidth and are unreliable owing to external noise and relative movement of nodes. Each node runs a medium access control protocol based on the IEEE 802.11 MAC standard.

The routing algorithms can be classified into pro-active and reactive protocols. The traditional IP routing algorithms such as distance vector (RIP [9]) and link-state (OSPF [11]) come under the category of pro-active algorithms. They build and maintain routing information about all the nodes in a network regardless of the usage or changes in the routes which appears to be wasteful in the context of ad hoc networks. Since the channel bandwidth is at a premium in these networks, many researchers proposed on-demand routing algorithms [14, 10, 12] which maintain only routes that are needed to send 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 [14, 10] and compare them with the destination-sequenced distance vector (DSDV) proposed by Perkins and Bhagwat [13]. We also propose an adaptive version of DSDV algorithm and analyze its performance. While our analysis confirms some of the previously seen claims [2, 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.

International Conference on High Performance Computing, Dec. 1999

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 the original and a variant of DSDV (pro-active), and AODV and DSR (on-demand) algorithms.

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 address, hop count and the next hop to reach the destination. Nodes broadcast their routing tables periodically to their neighbors, and incorporate any shorter routes obtained from neighbors. The major problem with the DV based algorithms is propagation of fallacious routing information among nodes, which leads to loops in routing paths. There are several solutions proposed to avoid this problem [3, 8, 13].

DSDV [13] solves the looping problem by attaching sequence numbers to routing entries. A node includes a next higher even sequence number for itself in its periodic and triggered updates. Any node that invalidates its entry to a destination, because of link failure, will increment the sequence number (which becomes an odd sequence number) and uses the new sequence number in its subsequent updates. Neighboring nodes having smaller sequence number for that destination than in the received update modify/invalidate their entry. An invalidated entry becomes valid only by the routing information propagated by the destination node with a higher even sequence number.

The triggered updates in DSDV consume a lot of channel bandwidth and, worse, they invalidate too many routing entries needlessly. 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 higher even sequence number or better metric with the same sequence number. Also a node performs a triggered update only 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. 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 hop count, destination sequence number, and the next hop. Unlike DSDV, however, it does not use periodic or triggered updates to disseminate routing information. When a node needs to send a packet to a destination for which it has an invalid routing entry, a route is obtained by flooding the network with route request (RREQ) packets and obtaining responses (RREP packets) from nodes with routes to the destination.

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. Invalidated routes are propagated to upstream 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 to detect the loss of a neighbor.

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 as 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 [5]). If the source does not have a route to the destination, it performs a route discovery as in the case of AODV, except that RREQ packets contain complete path information known up to that point and the RREP packets have the entire source route put by the destination.

DSR [10, 4] makes use of several optimizations to improve its performance. One of them is that the nodes can operate their interfaces in promiscuous mode and snoop on the source routes in the data packets and unicast routing packets. Also, an intermediate node may correct the path specified in a packet header if that path breaks. Snooping, which requires the device to be on all the time, may not be a feasible optimization for low-powered devices. To evaluate the effect of this optimization, we simulated two 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).

3 Performance Analysis

We used the *ns-2* network simulator [7] with the CMU extensions by Johnson et al. [4] for our analysis. The CMU extensions include implementations of DSDV, AODV, and DSR. The various parameters, timeouts and threshold values and optimizations used, unless explicitly specified, are exactly as described in the paper by Broch et al. [2]. For the sake of completeness, we outline the most relevant parameters for these algorithms.

For DSDV, the periodic update interval is 15 seconds. A node assumes a neighbor is lost if it does not hear the neighbor's update in 45 seconds (3*periodic update interval). For the proposed ADSDV, the periodic update interval is 5 seconds. Link layer feedback (provided by 802.11 LANs) is used to determine lost neighbors.

For AODV, we used the CMU implementation with one modification: the active route timeout is changed from 300 seconds (AODV300) to 3 seconds (AODV3), which is recommended by the latest specification of AODV [15].

In addition to the CMU implementation of DSR, we 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 simulated a network of 50 nodes randomly placed on a field of 1500m x 300m at the beginning of the simulation. Each node moves continuously, in a randomly chosen direction each time, at an average speed of 1 m/s, uniformly varied between 0-2 ms/, or 10 m/s, varied between 0-20 m/s [2].

A wireless channel based on 802.11 wireless LAN has 2 Mb/s bandwidth and a circular radio range with 250 meters radius. All protocols except for the original DSDV use the link-layer feedback to speedup detection of link breakages.

We used constant bit rate (CBR) traffic with 20 connections and 40 connections in our simulations. In each connection, the source sends 64-byte data packets at an average rate of 0.25-8 packets/second. Each simulation is started cold and the total simulation period is 1000 seconds.

Performance metrics. We use the average data packet latency and throughput (total number of data bits delivered) in Kb/s. To study the overheads of various routing algorithms, we plot routing packets transmitted/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 DSDV algorithms. Our simulations of DSDV and ADSDV indicate that ADSDV gives much improved latency and overhead than DSDV when the network load is high. ADSDV has more predictable routing overhead compared to DSDV, especially for the high node mobility case. Also the overhead in DSDV shoots up even at moderate loads, which explains why it has difficulty in sustaining throughput at high network loads. See [1] for more details.

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 versions of DSR. 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 did additional simu-

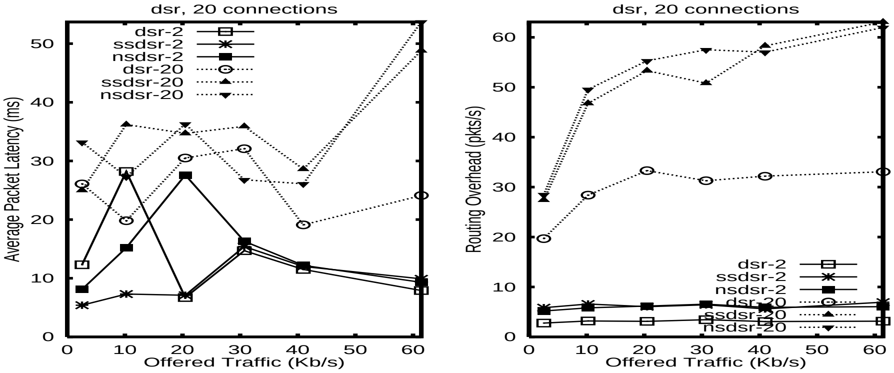

Fig. 1. Data packet latencies and overheads of the original and variants of DSR, algorithms for various traffic loads at low and high speeds. The number after the hyphen indicates the top speed by the mobile nodes.

lations with 40 CBR connections and data rates ranging from 1-8 packets/second (or approximately, 20-160 Kb/s).

We describe here only the results for the high mobility case. (Results for the low mobility case are given in [1]). ADSDV outperforms both DSR and AODV 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 2). ADSDV provides 35-43% higher throughput compared to both AODV and DSR. Furthermore, the routing overhead (Figure 2) 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.

4 Conclusions

The adaptive form of DSDV (ADSDV) presented is still a pro-active algorithm, since it depends mainly on periodic updates and controlled triggered updates, and seems to have superior performance to DSDV for increasing load in 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.

Compared to AODV and DSR, ADSDV performs poorly when the network load is low 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

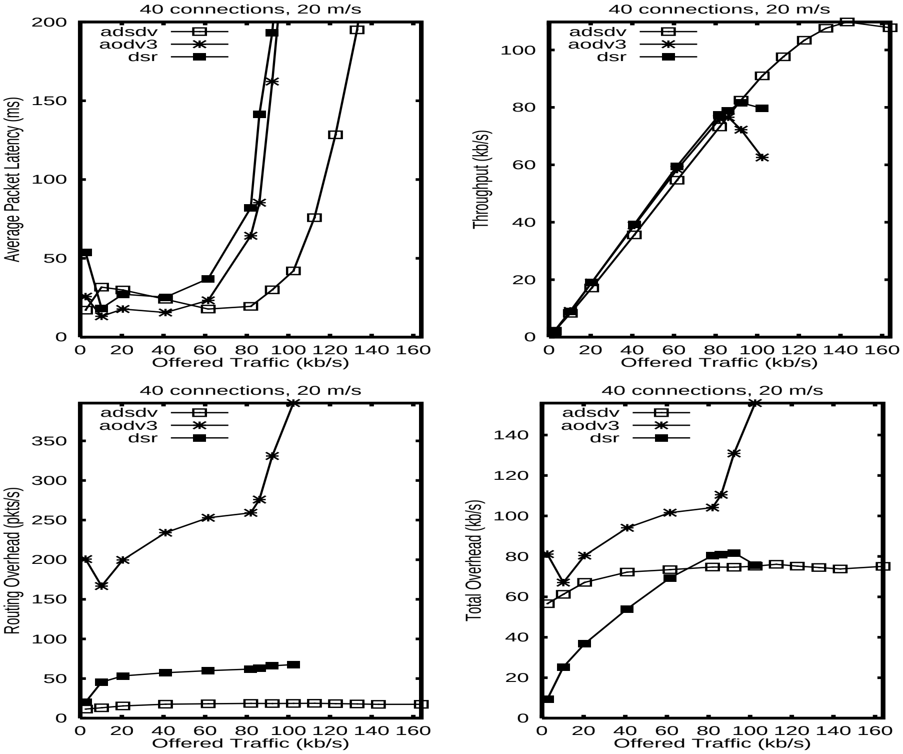

Fig. 2. Data packet latencies, throughputs, and overheads of ADSDV, AODV3 and DSR for the high node mobility case.

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 pro-active algorithms may be suitable or even preferable than pure 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.

Acknowledgments

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. R. V. Boppana, M. K. Marina, and S. P. Konduru, "An analysis of routing techniques for mobile and ad hoc networks." Manuscript http://www.cs.utsa.edu/faculty/boppana/papers/papers.htm, June 1999.
2. J. Broch et al., "A performance comparison of multi-hop wireless ad hoc network routing protocols," in *ACM Mobicom '98*, Oct. 1998.
3. 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.
4. CMU Monarch Group, "CMU Monarch extensions to the NS-2 simulator." Available from http://monarch.cs.cmu.edu/cmu-ns.html, 1998.
5. D. E. Comer, ed., *Internetworking with TCP/IP: volume 1.* Prentice Hall, Inc., 1995.
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.
9. C. Hedrick, "Routing information protocol." RFC 1058, 1988.
10. 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.
11. J. Moy, "Ospf version 2." RFC 1247, July 1991.
12. 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.
13. 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.
14. 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.
15. 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.