# EFFICIENT QUEUEING POLICIES FOR MOBILE AD HOC NETWORKS

Avinash Joshi and Samir R. Das
*Department of Electrical & Computer Engineering and Computer Science*
*University of Cincinnati*
*Cincinnati, OH 45221-0030*
*U.S.A.*
*Email: samir.das@uc.edu*

We investigate efficient queueing paradigms to be used at the radio interfaces for nodes in a mobile ad hoc network. Such a network employs dynamic routing to ensure communication between remote nodes. Low bandwidth of wireless links makes efficient queueing paradigms critical for the performance of routing protocols. We investigate various packet drop policies and priority scheduling policies targeted to improve aggregate performance measures for best-effort traffic. We evaluate these policies with an on-demand routing protocol, called AODV, and demonstrate that effective queueing paradigms can improve the fraction of packets delivered, and reduce delay and routing load.

## 1 INTRODUCTION

A mobile ad hoc network (or MANET) [1] is a group of mobile, wireless nodes which cooperatively and spontaneously form a network independent of any fixed infrastructure (e.g., base stations or access points) or centralized administration. A node communicates directly with the nodes within its radio range and indirectly with all others using a dynamically-determined multi-hop route. The MANET environment is typically characterized by energy-constrained nodes, variable-capacity and bandwidth-constrained wireless links and dynamic topology. Many dynamic routing protocols for MANET have been proposed and evaluated in recent literature. See, for example, [1]. However, little attention has been paid in developing efficient queueing mechanisms at the radio interfaces for effective operations of these routing protocols. Most MANETS use low-bandwidth radios. Thus, efficient queueing mechanisms can improve the effectiveness of routing protocols significantly.

In a traditional network protocol stack implementation, simple queueing mechanisms are employed at the network interface. See Figure 1. For example, packets are scheduled on the air in a *first-come-first-serve or FCFS* basis. Also, when the the interface queue buffer is full, all incoming packets are dropped (*drop-tail* policy). Though such schemes work well for high-bandwidth wired networks, we show that simple changes that give priority to certain types of packets in the scheduling and drop policies can significantly improve the per-

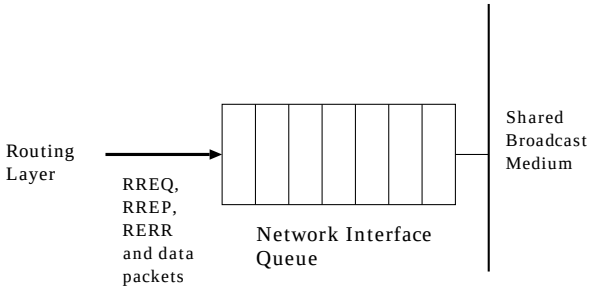



Figure 1: Queueing at the radio interface.

formance of mobile ad hoc network routing protocols. Our work is directed towards best-effort traffic and we try to optimize aggregate performance measures (e.g, aggregate network throughput or average delay). We do not consider queueing issues related to any sort of guaranteed service traffic. Neither do we consider fairness issues between flows [2].

In our knowledge, scheduling issues have not been investigated much for ad hoc networks. Channel-state dependent scheduling [3] has been investigated for wireless LANs, where packets are scheduled on the air-interface based on the link quality. Packets seeing a better link quality gets higher priority. This scheme has been targeted for access-point based LANs, but can potentially be used in ad hoc networks. More recently, message length based scheduling has been investigated in ad hoc networks [4].

For our investigation, we have chosen a popular on-demand routing protocol called AODV (ad hoc on-demand distance vector routing) [5,6]. On-demand routing protocols have received significant interest in the MANET community, because of their ability to perform routing tasks only on an "as-needed" basis and thus to keep the routing overheads low. While our investigations are currently on AODV, the policies that we implement are general in nature and should be applicable for other on-demand routing protocols such as Dynamic Source Routing (DSR) [7,8].

The rest of this paper is organized as follows. In the next section, we briefly describe the AODV routing protocol on which all our later studies are based. In section 3 we present several scheduling and drop policies. We evaluate the performance of our proposed policies against a baseline policy via simulations in section 4. We conclude in section 5.

## 2 ON-DEMAND ROUTING AND AODV

Ad hoc On Demand Distance Vector (AODV) [5,6] uses traditional hop-by-hop routing, but it constructs the routing tables on demand. AODV initiates a route discovery process when the source of a data packet does not have a route to the intended destination in its routing table. Route discovery works by flooding a routing query in the network. This is implemented as follows. The source broadcasts a route request (RREQ) packet. Nodes receiving RREQ record a *reverse* route back towards the source, using the node from which the RREQ was received as the next-hop, and then re-broadcasts the RREQ. Duplicate copies of the RREQ received via alternate paths are ignored.

When the RREQ packet reaches the destination or any node having a route to the destination, it sends a route reply (RREP) packet back to the source, using the reverse route set up earlier. As the RREP packet goes back to the source, a corresponding *forward* route is created at each intermediate node towards the destination. Once the RREP packet reaches the source, data traffic can now flow along this forward route.

To prevent routing loops, AODV maintains a sequence number on each node. The sequence numbers always increase and help implement a form of logical clock. Any routing information transmitted on routing packets or maintained on a node is tagged with the last known sequence number for the destination of the route. The AODV protocol maintains an invariant that the destination sequence numbers in the routing table entries on the nodes along a valid route are always monotonically increasing. This guarantees loop freedom. Other than preventing loops, sequence numbers also ensure freshness of routes. Given a choice of multiple routes, the one with a newer sequence number is always chosen.

A set of predecessor nodes is maintained for each routing table entry, indicating a set of neighboring nodes that use that entry to route data packets. These nodes are notified with route error (RERR) packets when the next hop link breaks. Each predecessor node, in turn, forwards the RERR to its own set of predecessors, thus effectively erasing all routes using the broken link. This RERR is thus propagated to each source routing traffic through the failed link, causing the route discovery process to be re-initiated if routes are still needed. An important feature of AODV is maintenance of timer based states in each node, regarding utilization of individual routes. A route is "expired" if not used recently.

## 3 SCHEDULING AND DROP POLICIES

Most current simulation results [9,10] for ad hoc networks use a simple form priority scheduling, where all routing packets (RREQ, RREP and RERR) are given priority over data packets for transmission at the network interface queue. See Figure 1. The rationale is that route set up and maintenance activities are always critical; without such activities data packets cannot be routed any way. However, a closer investigation reveals that such a simple priority scheme is not always very effective. We explain this in the following subsection.

### *3.1 Simple Priority Scheduling*

Flooding of RREQ packets in search of routes in the network is a highly redundant process. Duplicate copies of RREQ are received by nodes via alternate paths only to be discarded later. Only the path taken by the first RREQ received by any node is useful. While the intuition is that the first arriving copy of RREQ defines the shortest delay path and hence the path that should be used for routing data packets, the delays experienced and RREQ and data packets along the same path are actually different. This is because of priority scheduling of all routing packets. Ideally, a network designer would like the RREQ packets to experience similar delays as data packets along the same path, so that the first-arriving RREQ automatically defines the shortest delay path for the data packets. To solve this problem, we propose to treat RREQ similarly as data packets.

However, both route replies and errors (RREP and RERR) should be given priority as before, as they carry crucial routing information which has much less redundancy. Fast transmittal of RREP helps in completing the route discovery process fast. Similarly, fast transmittal of RERR helps in erasing stale routes starting a new route discovery faster. In summary, treating data and RREQ packets similarly and giving priority only to RREP and RERR packets in the interface queue can improve performance by helping the routing protocol discover the least-congested route.

### *3.2 Drop Policies*

In the existing simulation studies [9,10] the incoming packets are dropped regardless of the packet type whenever the the interface queue buffer is full. With this drop-tail policy, an RERR, for example, may be dropped while the buffer may be full with data packets. A successful transmission of an RERR can save a large number of misdirected data packets later on. Clearly, a well-designed drop policy has potential to improve routing performance.

We explore two drop policies. To simplify description, we assume that the incoming packet is alway accepted. However, if that causes a buffer overflow, one or more packets are dropped from the buffer by following a specific policy to make enough room. The policy may require the newly arrived packet to be dropped.

**Dropping data packets**

In this policy, one or more data packets are dropped to make room. The data packets that are dropped are the *farthest* (in terms of hop counts) from their destinations. To determine this, the routing table is consulted for each data packet in the buffer. This is based on the rationale that data packets going over longer routes are more prone to be dropped en route because of route errors. This makes the data packets that are farthest from destinations (i.e., with the largest remaining hop counts as per the routing table) good candidates for drops. If there are more candidate packets than necessary (e.g., many data packets with the largest remaining hop counts as determined from the routing table), more recently arrived packets are given priority for drops.

**Dropping route requests**

Here, we extend the above drop policy by giving RREQ packets priority for drops. The rationale is that the RREQs should avoid congested routes. If the interface queue buffer is full, a good idea will be not to attempt establishing new routes through this node. This will provide a better load balance. Thus, we drop RREQ packets from the tail of the interface queue (i.e., more recently arrived packets are dropped first) until enough room is exposed. If not, data packets are dropped following the previous policy.

### *3.3 Comprehensive Priority Scheduling*

So far, we have discussed simple priority scheduling and drop policies. The scheduling policy uses RREQ and data packets in a low priority class and RREP and RERR in a high priority class. FCFS is used within classes. The drop policies extend the simple drop-tail policy, by dropping RREQ packets first and then data packets with highest remaining hops. Here, we propose an additional scheduling policy on top of these policies. RREQ and data packets still have lower priority, but among themselves they they are dispatched in certain order instead of FCFS. Data packets are dispatched in the order of their remaining hop counts. But this is done only upto the next RREQ packet. Then the RREQ packet is dispatched. The rationale is to give less priority to

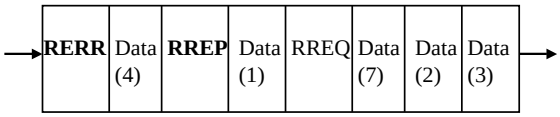



Figure 2: Example illustrating the comprehensive scheduling scheme. The numbers in parentheses with data packets denote the remaining no. of hops. The RREP and RERR are dispatched first in that order. Then Data(2), Data(3) and Data(7) are dispatched in that order. Then the RREQ is dispatched, followed by Data(1) and Data(4).

the data packets that are far from their destinations as they have a higher susceptibility to drops. But this process does not change the order in which the RREQ packets are served. Figure 2 illustrates this scheduling scheme with an example.

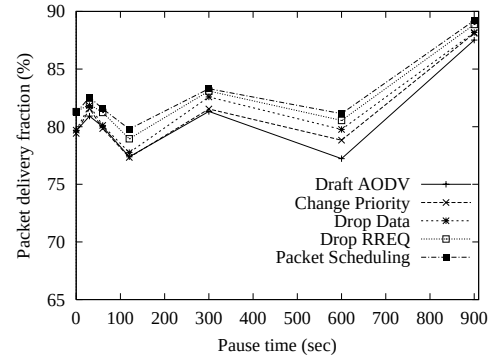
## 4 PERFORMANCE EVALUATION

We have simulated the AODV routing protocol using the above queueing policies at the network interface. The *ns-2* simulator [11] with the wireless stack as described in [9] is used. This simulator, with similar mobility and traffic models, has been used in quite a few recent performance studies for ad hoc networks. See, for example, [9,12,10].

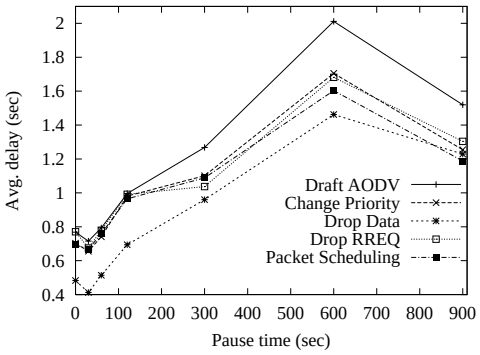
### *4.1 Simulation Scenarios and Performance Metrics*

In our simulation experiments 50 nodes move around in a rectangular area of 1500m $\times$ 300m according to the random waypoint mobility model [9]. In this model, the nodes move to a random destination with a randomly chosen speed (0-20m/s in our experiments), and pause there for certain time before moving to another random destination. Each node uses the IEEE 802.11 standard [13] MAC layer. The radio model is very similar to the first generation WaveLAN [14] radios with nominal radio range of 250m.

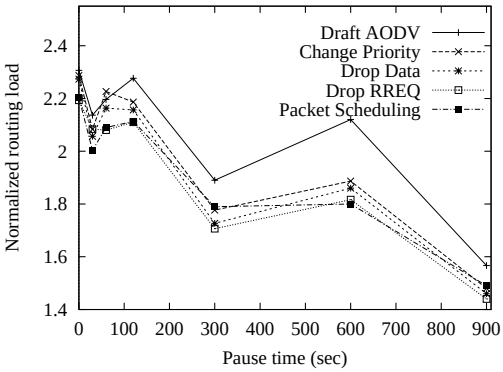
The experiments use different number of sources with moderate packet rate and varying pause times. The results for a 50 node network with 30 and 40 sources and varying pause times from 0 to 900 seconds are shown. Note that a pause time of 0 seconds means constant movement, and pause time of 900 seconds means a stationary network, as our simulations are run for 900 simulated seconds. The sources are CBR (constant bit rate) and generates UDP packets with a packet rate of 4 packets/sec for 30 sources and 3 packet/sec for 40 sources. We use a slower rate with 40 sources, as the network congestion




(a) Fraction Delivery

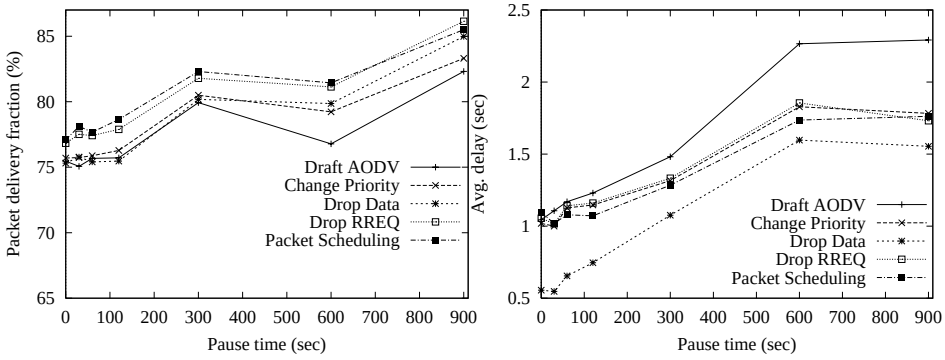



(b) Average Delay




(c) Normalized Routing Load

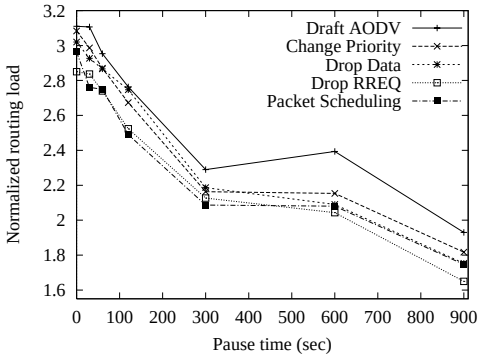
Figure 3: Performance of various policies for 50 node model with 30 sources.




(a) Fraction Delivery




(b) Average Delay




(c) Normalized Routing Load

Figure 4: Performance of various policies for 50 node model with 40 sources.

is too high otherwise for any meaningful comparison. 512 byte data packets are used. Each data point represents an average of at least 5 runs with identical traffic models, but with different randomly generated mobility scenarios. For fairness, identical mobility and traffic scenarios are used across various techniques.

The following 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.
- *End-to-end delay*: measured as the average end-to-end latency of data packets in ms.
- *Normalized routing load*: measured as the number of routing packets transmitted for each data packet delivered at the destination.

### *4.2 Simulation Results*

Figures 3 and 4 demonstrate the relative performance of various policies in a 50 node network with 30 and 40 sources respectively. The labels used in the plots denote various policies. "Base AODV" denotes the original priority scheduling policy where all routing packets are given priority over data packets. "Change Priority" denotes the scheme where only RREP and RERR receive higher priority, and RREQ and data packets receive the same priority. "Drop Data" and "Drop RREQ" denote the two drop policies implemented on top of the "Change Priority" policy. The "Comprehensive Scheduling" is the policy described in subsection 3.3. It is implemented on top of the "Drop RREQ" policy.

As shown in the performance plots, the packet fraction delivery progressively generally improves for more sophisticated queueing policies. The comprehensive scheduling policy works best. It delivers upto about 5% more packets compared to the base policy. The "Drop RREQ" policy follows very closely, indicating that most of the performance improvement is achieved by the load balancing effects. The scheduling based on remaining hop counts for data packets provides only minor benefits.

All policies reduce delay over the base policy. The "Drop Data" policy has the best delay performance. The reason is somewhat statistical — this policy preferentially drops data packets susceptible to delays (large remaining hop counts) when the buffer is full. This makes the average delay very small. The comprehensive scheme follows second. Considering that it has a higher packet

delivery fraction, we feel that it is still the best policy to use. It reduces delay by up to 30% over the base policy. Routing load-wise it is also close to the best, and improves the load by up to about 50% compared to the base policy.

## 5 CONCLUSIONS

We have proposed two packet dropping policies and one packet scheduling policy for ad hoc networks. The schemes have been evaluated with the AODV routing protocol. We believe that similar on-demand routing protocol (such as DSR) will give qualitatively similar results. These schemes provide a better load balance and prioritizes packet dispatch by giving preference to either critical routing packets or data packets that are less likely to be dropped. It was demonstrated that all policies investigated perform better than the base scheduling policy which gives priority to *all* routing packets, but otherwise does a FCFS scheduling. The comprehensive policy is the policy of choice, as it delivers the largest fraction of data packets – upto about 5% over the base. It also improves delay by up to about 30% and routing load by up to about 50%. This study also demonstrates that intelligent queueing paradigms can lead to improved routing performance.

## ACKNOWLEDGMENTS

This work is partially supported by NSF CAREER grant ACI-0096186 and NSF networking research grant ANI-0096264.

## REFERENCES

1. C. Perkins, editor. *Ad Hoc Networking.* Addison Wesley, 2001.
2. V. Bhargavan, S. Lu, and R. Srikant. Fair scheduling in wireless packet networks. *IEEE/ACM Transactions on Networking*, 7(4):473–489, August 1999.
3. P. Bhagwat, P. Bhattacharya, A. Krishna, and S. Tripathi. Using channel state dependent packet scheduling to improve TCP throughput over wireless LANs. *ACM/Baltzer Wireless Networks Journal*, pages 91–102, 1997.
4. R. Kakaraparthi, R. Duggirala, and Dharma P. Agrawal. Efficient message scheduling in an ad-hoc network. In *Proceedings of the 2nd IEEE Wireless Communications and Networking Conference*, pages 1226–1231, Chicago, September 2000.
5. Charles Perkins and Elizabeth Royer. Ad hoc on-demand distance vector routing. In *Proceedings of the 2nd IEEE Workshop on Mobile Computing*

*Systems and Applications*, pages 90–100, Feb 1999.

6. Charles Perkins, Elizabeth Royer, and Samir R. Das. Ad hoc on demand distance vector (AODV) routing. http://www.ietf.org/internet-drafts/ draft-ietf-manet-aodv-10.txt, Jan 2002. IETF Internet Draft (work in progress).
7. D. Johnson and D. Maltz. Dynamic source routing in ad hoc wireless networks. In T. Imielinski and H. Korth, editors, *Mobile computing*, chapter 5. Kluwer Academic, 1996.
8. David Johnson, David Maltz, Yih-Chun Hu, and Jorjeta Jetcheva. The dynamic source routing protocol for mobile ad hoc networks. http://www.ietf.org/internet-drafts/ draft-ietf-manet-dsr-05.txt, March 2001. IETF Internet Draft (work in progress).
9. 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.
10. 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*, pages 3–12, March 2000.
11. Kevin Fall and Kannan Varadhan (Eds.). *ns* notes and documentation, 1999. available from http://www-mash.cs.berkeley.edu/ns/.
12. Per Johansson, Tony Larsson, Nicklas Hedman, and Bartosz Mielczarek. Routing protocols for mobile ad-hoc networks - a comparative performance analysis. In *Proceedings of the 5th International Conference on Mobile Computing and Networking (ACM MOBICOM'99)*, pages 195–206, August 1999.
13. IEEE Standards Department. Wireless LAN medium access control (MAC) and physical layer (PHY) specifications, IEEE standard 802.11–1997, 1997.
14. Bruce Tuch. Development of WaveLAN, an ISM band wireless LAN. *AT&T Technical Journal*, 72(4):27–33, July/Aug 1993.