Impact of Failures on Routing in Mobile Ad Hoc Networks Using DSR

Xiaobing Hou and David Tipper
Department of Information Science and Telecommunications
University of Pittsburgh
Pittsburgh, PA 15260
{xiaobing, dtipper}@mail.sis.pitt.edu

Abstract - In this paper, we study the transient behavior of DSR protocol in case of path failures in mobile ad hoc networks. Transient analysis shows that after packets en route reach the failure point, they may block the interface queue and cause significant waste of wireless resources resulting in poor network performance. We present a scheme called Failure Record to prevent this situation and improve various performances of the network. Simulation results show that our scheme is very efficient.

Keywords – Ad hoc networks, routing protocols, simulation, DSR, failure record, queuing, transient analysis.

1. Introduction

In an ad hoc network, mobile nodes must cooperate to dynamically establish routes using multihop wireless links. There is no stationary infrastructure, and each node acts as a router. A packet may have to be forwarded by a sequence of nodes to reach its destination. With the mobile nature of ad hoc networks, routes are supposed to break very often and packet loss rate is supposed to be higher than wireline networks. So ad hoc networks require highly adaptive routing protocols and efficient failure recovery strategies to deal with the frequent topology changes. A *mobile ad hoc networking* (MANET) working group [1] has been formed within the Internet Engineering Task Force (IETF) to develop a routing framework for ad hoc networks. Many routing protocols have been proposed to solve the dynamic multihop routing problem in ad hoc networks.

Traditional routing protocols, such as distance vector and link state, were designed for static infrastructured networks, and a dynamic topology was not considered in their design. Obviously, it will be unsuitable to use any non-adaptive protocol for a highly mobile network. Basically, there are two types of routing protocols for ad hoc networks, *proactive* routing and *reactive* routing. Proactive routing protocols attempt to maintain consistent, up-to-date routing information from each node to every other node in the network. Such protocols are termed proactive because they store route information even before it is needed. Proactive protocols suffer the disadvantage of additional control traffic that is needed to continually update stale routing entries. Some of the most popular proactive protocols are DSDV [2], WRP [3], OLSR [4] and FSR [5].

Reactive routing creates and maintains routes only when desired by the source node. Therefore, it's also known as *on-demand*, *source-initiated*, or *demand-driven* routing [6]. When a node requires a route to a destination, it initiates a route discovery process within the network, typically, by some form of flooding. This process is completed once a route is found or all possible route permutations have been examined. Once a route has been established, it is maintained by a route maintenance procedure until either the destination becomes inaccessible along every path from the source or until the route is no longer desired. Compared to proactive routing, reactive routing consumes far less bandwidth for maintaining the routing tables at each node when only a small subset of all available routes is in use at any time. Proposed reactive routing protocols include DSR [7][8][9], AODV [10][11], TORA [12][13], etc.. A review of ad hoc routing protocols is given in [6].

Our goal in this paper is to study the transient behavior of DSR protocol, one of the most popular ad hoc network routing protocols, when path failures occur. We present an improvement scheme termed the Failure Record to increase the packet delivery fraction and decrease the average end-to-end delay of data packets. The rest of the paper is organized as follows. Section 2 provides a brief overview of the DSR protocol and the manet simulation model used. In section 3, we study the transient behavior of the DSR and indicate the impact of failures on DSR. We present a scheme to improve the performance in section 4. Section 5 shows the simulation results. Conclusions and our future work are given in section 6.

2. DSR Overview and Simulation Model

In the Dynamic Source Routing (DSR) [7][8][9] protocol, to send a packet to another node, the sender constructs a source route in the packet's header, giving the address of each node in the network through which the packet should be forwarded to reach the destination. The sender then transmits the packet to the first hop identified in the source route. When a node receives a packet, if this node is not the final destination of the packet, it simply transmits the packet to the next hop identified in the source route in the packet's header. DSR maintains a route cache in each node to contain the source routes the node is aware of. The protocol consists of two major phases: *route discovery* and *route maintenance*. When a node has a packet to send and has no valid route in the route cache, it initiates route discovery by broadcasting a *route request* packet. The intermediate nodes add their own address to the *route record* of the packet until the packet reaches the destination or an intermediate node that has a valid route to the destination in its route cache. Typically, the first copy of the route request packet reaching the destination contains the shortest route. A *route reply* is generated and sent back to the initiator via the reversed route in the route record if symmetric links are supported, or a new route discovered in a similar way. In route maintenance, acknowledgements are used to verify the correct operation of the route links and *route erro*r

packets are used to report link or node failures to the original sender.

All packets with a valid source route are put in the network interface queue, which is an output queue of packets from the network protocol stack waiting to be transmitted by the network interface. This queue is used to hold packets while the network interface is in the process of transmitting another packet.

DSR has many advantages, including simplicity, correctness, and flexibility [14]. Since all routing decisions for a packet are made by the sender, intermediate nodes do not need to maintain up-to-date, consistent routing information for the destination. By including the source route in the packet header, the route over which a packet is forwarded can be guaranteed to be loop free. In addition, for reasons such as load balancing, the perceived longevity and reliability of the route, the security of the nodes, or differentiated treatment of different types or classes of packets for QoS, it is possible for the sender to use different routes for different packets, without requiring coordination or explicit support by the intermediate nodes.

In this paper, we utilized ns-2 network simulator [16], with CMU Monarch Project wireless and mobile ns-2 extensions, to study the performance of DSR and our scheme. The distributed coordination function (DCF) of IEEE 802.11(b) for wireless LANs is used as the MAC layer. It uses Request-to-send (RTS) and Clear-to-send (CTS) messages and virtual carrier sensing for data transmission to reduce the impact of the hidden terminal problem. The radio model is similar to Lucent's WaveLAN, which is a shared media radio with a nominal bit rate of 2Mb/sec and a nominal radio range of 250 meters. Link/node failure is detected by the feedback from the MAC layer. In case of failing to deliver a packet to the next hop, the MAC layer informs the routing layer of the failure. No additional IP routing layer schemes such as hello messages are needed.

Routing protocols maintain a *send buffer* of 64 packets, which buffers all data packets without a source route. The packets waiting in the send buffer for more than 30 sec are dropped. All packets from the routing layer are queued at the *interface queue* waiting for MAC layer to transfer. The interface queue is FIFO with simple priority, i.e., routing packets have higher priority and will be put in the front of the queue. The size of the queue is 64 packets.

The simulation results presented in this paper are based on scenarios randomly generated by CMU ns-2 extensions. CBR traffic based on UDP is used and the source-destination pairs are picked randomly from the network. Each packet carries 512 bytes of data payload, making the packet size 532 bytes including an IP header. The number of source-destination pairs and the packet rate is varied in the simulation. We use 20 and 30 traffic sources and a packet rate of 4 packets/sec. The network is of 50 nodes randomly placed within a 1500 m × 300 m area. The node mobility model is Random Waypoint [8][14], in which each node begins at a random position, picks a new random position to which to move, and moves there in a straight line at a random speed. Each node independently repeats this behavior and the average degree of mobility is varied by making each node remain stationary for a period called *pause time* every time before it moves to the next position. The smaller the pause time, the higher the average mobility. In our simulation, the maximum speed of the nodes is 20 m/s and the pause time is varied between 0 s and 900 s. Simulations are run for 900 simulated seconds, and at least five runs for each data point to get an average. The above simulation parameters are summarized in Table 1, for additional details, please refer to [17]. We use this simulation model to study the general performance of different schemes in section 5 and some transient behavior in section 3 and 4. In section 3 and 4, we use another simple simulation model to do more transient analysis, which is described in section 3.

Table 1
Simulation Parameters

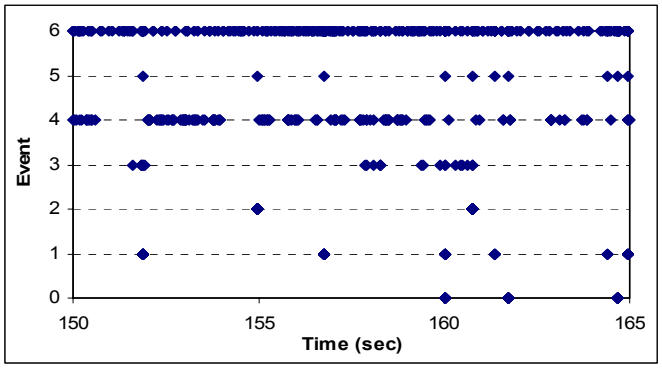
Parameters	Values	Parameters	Values
Simulation time	900 sec	Bandwidth	2 Mb/s
MAC layer	IEEE 802.11	Radio range	250 m
Physical layer		Send buffer size	64
Traffic model	CBR	Send buffer timeout	30 sec
Packet size	512 bytes	Interface queue size	64
Network size	50 nodes	Source number	20, 30
Area	1500 m × 300 m	Packet rate	4 packets/s
Max. speed	20 m/s	Pause time	0 – 900 s

3. Transient Analysis of DSR

Link failures or node failures can cause packet loss in ad hoc networks and are typically seen as a major reason for packet loss in ad hoc networks. However, the impacts of a failure on the network performance depend on a complex interaction of several factors and one of them is routing protocol [15]. In this section, we present a result of transient simulation analysis to determine some aspects of failure effects of DSR.

Fig. 1 is an observation from a simulation on the behavior of a random node during a randomly chosen period. The parameters of the simulation are given in section 2. Fig. 1(a) shows the transient behavior of the node in case of failures. Fig. 1(b) shows the queue length of the same node during the same period with queue capacity of 64. From the figures we can see that the major reason of packet dropping is not because of lack of valid routes but full queues. In fact, from simulation results we observed that full queue causes more packet loss than any other reason. Further analysis of Fig.1 indicates that the full queue is caused by the interface queue management and the shared channel access mechanism of the MAC protocol. In Fig. 1(a), every point represents an event happened at a specific time instance. After it enters the interface queue, a packet either is transmitted successfully or returned to the routing layer because of a failure in the MAC layer. The DSR will try to salvage the returned packet by searching the route cache for an alternate path. If there is no alternate path available, the packet is either put into send buffer for a new route discovery process if the node is the source of this packet or dropped otherwise. If the interface

queue is full, the incoming packet will be dropped. In a single channel scenario, a single queue is typically used and that's also the case of our simulation. In case of a failure, a node loses connection to one of its neighbor nodes. The packet going to that node will block all packets behind even they are going to other neighbors that are still connected to this node. Before the MAC layer concludes that it really is a failure, it will retry a couple of times (seven in 802.11). During this period, the long blank period at the line of "successful transmission" events in the figure, packets are still coming and making the queue longer. If this happens very often the queue will keep growing until it's full. In this simulation the mobility is not high (maximum speed is 20 m/sec, pause time is 120 sec) and the number of physical layer failures is not necessary equal to the number of MAC layer failures. After a physical layer failure (link/node failure), if there are more than one packet using this link it will cause more than one MAC failures. Although DSR will take all such packets off the queue after a failure, new packets are still coming in and put into queue immediately without any checking. If the new incoming packets use the failed link, they will block all other packets causing network-wide low throughput and long delay.



Event Value	Event
0	Pkt generated by itself, retry
1	Successful salvage
2	Dropped without alternate path
3	Dropped due to full queue
4	Successful transmission
5	Failure in MAC layer
6	Entering interface queue

(a)

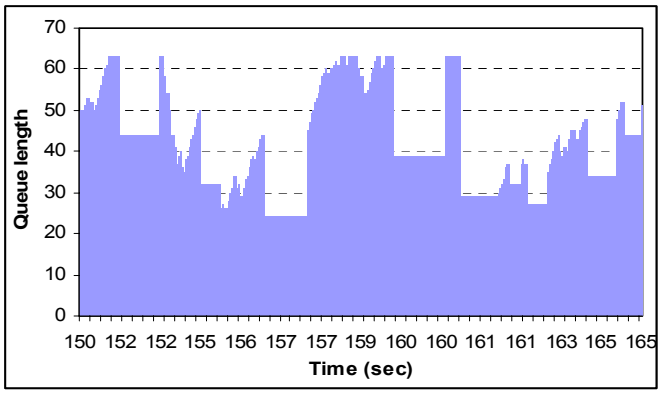
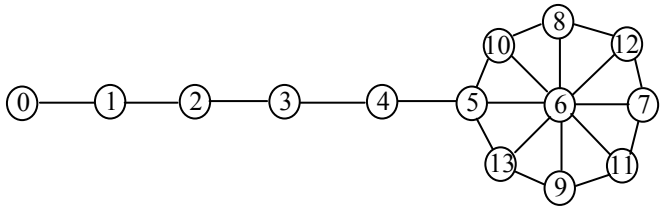
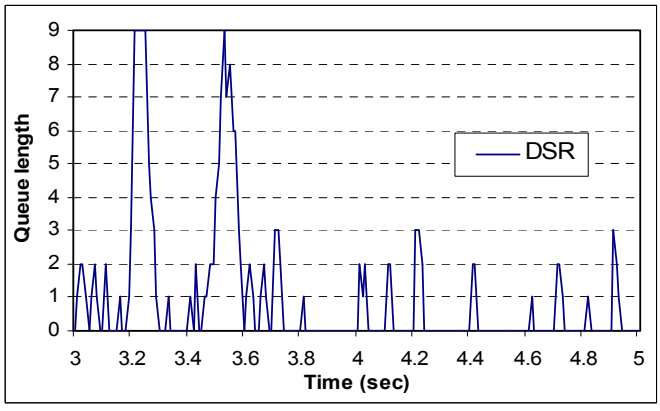


Fig. 1. Transient Behavior of DSR

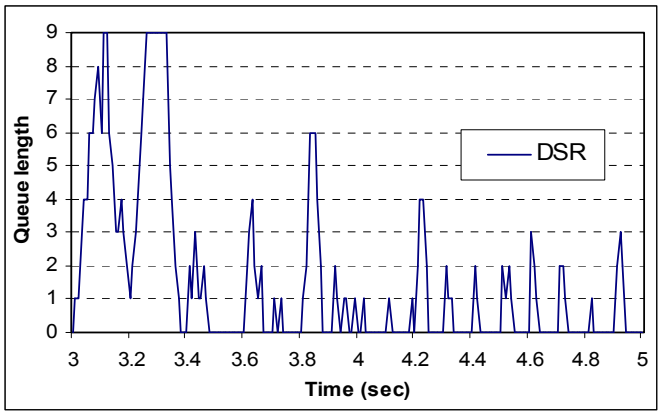
To better understand the transient behavior of DSR after a failure, we adopt the nonstationary simulation methodology of [18] to observe the queue length versus time by conducting a simple simulation with only one failure. The topology used is shown in Fig. 2(a). Background traffic is 6 flows of 10 packet/sec (8-9, 9-8, 10-11, 11-10, 12-13, 13-12) and one flow of 20 packet/sec from node 6 to node 0. The traffic flow facing failure is from node 0 to node 7 with a rate of 20 packet/sec. All traffic flows are CBR over UDP. The failure happens at simulation time instance 3 when node 7 moves out of the transmission range of all other nodes and separated from the network. Every node has a FIFO queue with a capacity of 9. For other simulation parameters, please refer to section 2. Fig. 2(b) shows the observation to node 6 in two typical runs (each run uses different seed). We can see, with DSR, typically there are two or more peaks in the figure. Each peak represents a packet using link 6-7 blocking the queue of node 6, causing full queue and packet loss. It's clear that a single physical failure can generate multiple MAC failures using DSR protocol. We repeat the simulation for 100 times and the ensemble average is shown in Fig. 2(c). Since the confidence interval is very small, we take it off from the figure for clarity. Since there is only one failure in the simulation and it's handled very quickly by the MAC protocol, the width of the peak in each simulation run is very small. Also, the random arrival time of the packets using link 6-7 shifts the position of the peaks. So the average of the peak values is much smaller than the queue capacity. But we still can see there are two or more peaks in Fig. (c).



(a)

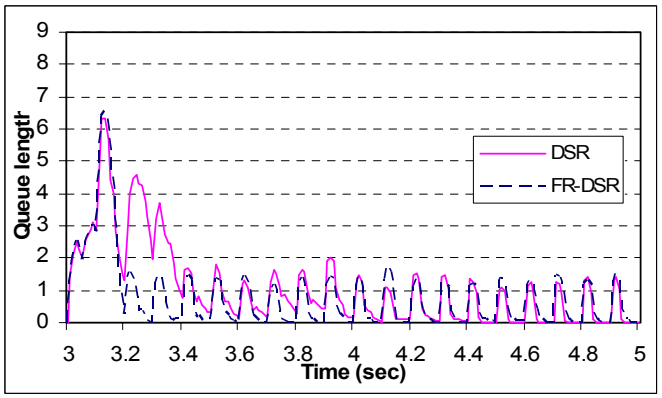


Run 1



Run 2

(b)



(c)

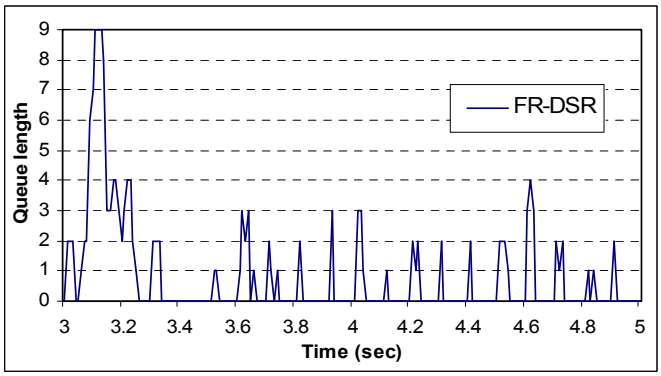
Fig. 2. DSR Transient Analysis to a Simple Scenario

4. Failure Record

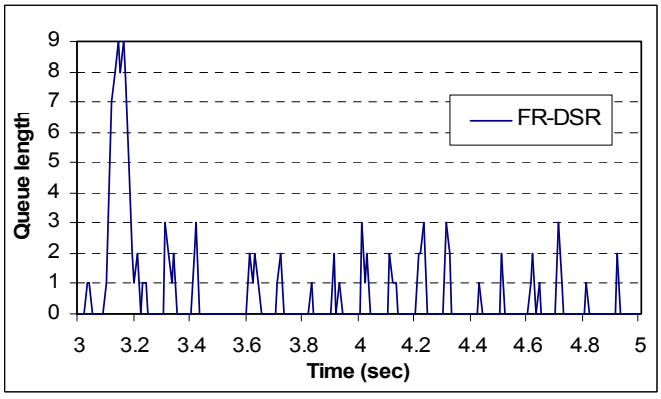
From the simulation results in the last section, we know that a major source of packet loss in DSR is caused by the inappropriate interface queue management between DSR and MAC layers after failures. It's a very bad idea to put all incoming packets into the queue directly and let the packets using the failed link try that link and waste precious wireless resources. In this section, we propose a simple scheme called the Failure Record (FR) to solve the problem and improve the performance of DSR.

The reason that all packets are put into queue directly in DSR is that nodes have no memory about what happened before. So in our scheme, every node creates a failure record to remember which of its links have been broken in the past so that the packets using those links can be appropriately handled before they enter the queue. Specifically, each node has a table with each of its entries standing for a failure happened to one of its links in the past a few seconds. Each entry includes the name of the failed link (represented by the ID of this node and the node ID on the other end of the link) and the expiration time of this entry. The routing protocol checks every incoming packet. If it is not using any link in the table, the packet enters the queue directly. Otherwise, the protocol should salvage the packet without trying to transmit it. The salvage process is similar to the one in DSR. The node searches its route cache for alternate route to the destination. If no such route exists, the packet is dropped. The difference is no error message sent back to the source node since the node already did this when the link is failed and put into the table. The incoming packets using the failed link are typically the ones en route when the failure happened. All packets starting from the source node after the error message reaches the source use an alternate route to avoid the failed link. This is also the reason why we need an expiration time for each entry in the table. After all en route packets pass this node or are dropped by this node, the associated entry should be erased. In the following simulation, we use a fixed expiration time. We expect a variable expiration time adaptive to traffic load and node mobility can improve the performance of this scheme and leave it for future research.

We repeated the simulation described in Section 3 using the FR approach. Fig. 3 shows the result for two typical runs. We can see there is only one peak of queue length after the failure happens. Multiple MAC failures happened in DSR are prevented. The ensemble average of 100 runs of simulation is shown in Fig. 2(c). Compared with DSR results, the transient congestion of node 6 in FR is clearly reduced. Another metric that can show the reduced transient congestion is the probability of full queue after the failure. With DSR, in one second after the failure, 3.7% of the time the queue is full, 16.4% of the time the queue length is greater than 4. With FR, the values are 1.5% and 7.9%, respectively.



Run 1



Run 2

Fig. 3. Two Runs of FR-DSR in a Simple Scenario

5. Performance Evaluation

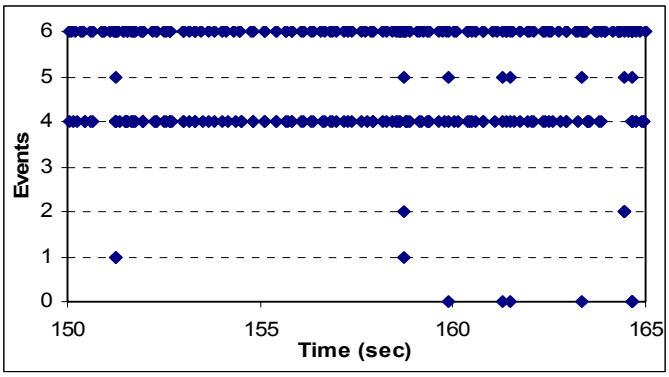
We evaluated three performance metrics defined as follows:

- Packet delivery fraction: ratio of the packets delivered to the destination to those generated by the CBR sources;
- Average end-to-end delay: all delays experienced by each packet, including queuing delays, route discovery delays, retransmission delays at the MAC, DSR and FR-DSR, propagation delays and transfer times;
- Normalized routing load: the ratio between the number of hop-wise transmission of routing packets and the number of data packets delivered at the destination.

Figure 4 shows typical transient behavior of FR-DSR from a single run. Note that although Fig. 4 shows the same node in same time slot as in Fig.1, it's unlikely to have same packet arrival process since it depends on the output process of other nodes. Comparing Fig.1(b) and Fig.4(b), we can see that the interface queue in FR-DSR is never full in the chosen time slot. Actually, the queue is far less than full. This is from the effect of Failure Record scheme. FR can reduce the number of MAC layer failure events so that it can increase the throughput. From the figure, we can see the output process of FR-DSR is smoother and there are less queue blocking periods.

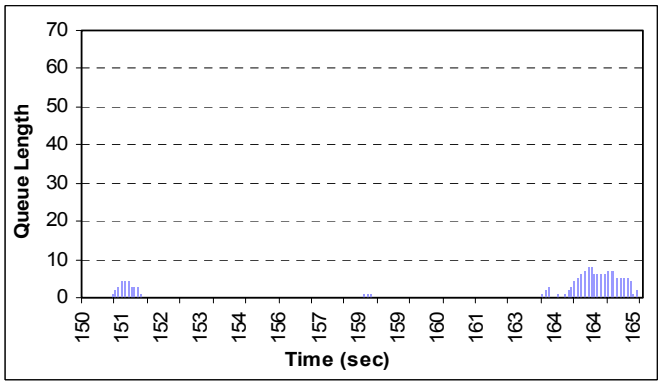
In our simulation, failure record expiration times in FR-DSR are 0.5 and 3 seconds for 20 and 30 sources respectively. Fig. 5 shows the packet delivery fractions for different numbers of sources versus the pause time. In case of 20 sources, the two schemes almost have same performance. When the traffic is increased to 30 sources, FR-DSR apparently outperforms DSR at low pause times. The reason that FR-DSR performs better with higher traffic is there are more en route packets due to either high traffic generation or congestion. From Fig. 6, we can have similar observation with respect to average end-to-end delay. Another observation, from these two figures is that FR-DSR has bigger improvement when mobility is higher (smaller pause time). With higher mobility, path failures happen more often generating more en route packets using broken links.

FR-DSR also has a better normalized routing load as shown in Fig. 7. Basically, Failure Record cannot reduce routing overhead. But with higher packet delivery fraction, the normalized routing load can be reduced.



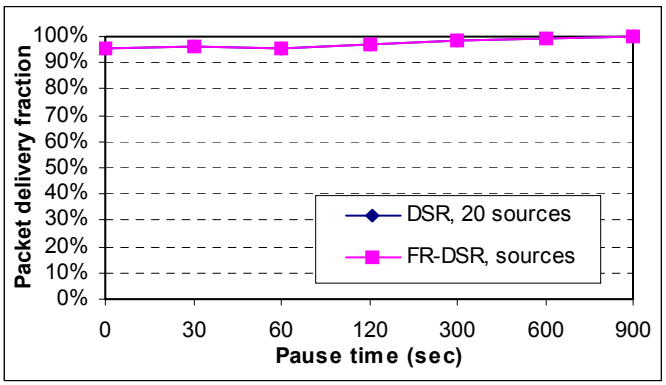
Event Value	Event
0	Pkt generated by itself, retry
1	Successful salvage
2	Dropped without alternate path
3	Dropped due to full queue
4	Successful transmission
5	Failure in MAC layer
6	Entering interface queue

(a)



(b)

Fig. 4. Transient Behavior of FR-DSR



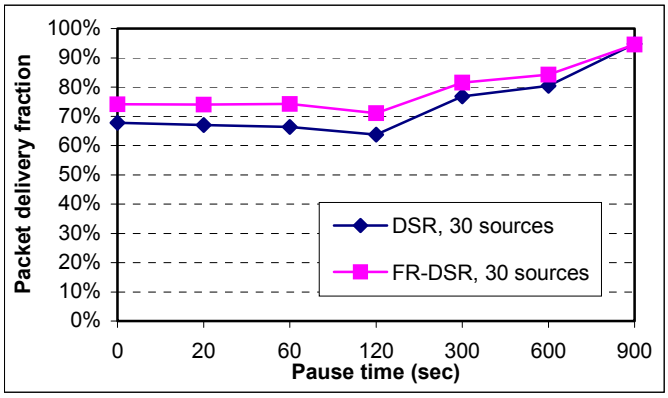
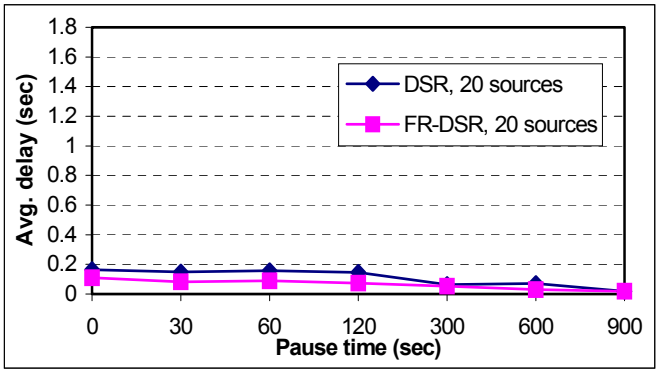


Fig. 5. Packet Delivery Fractions



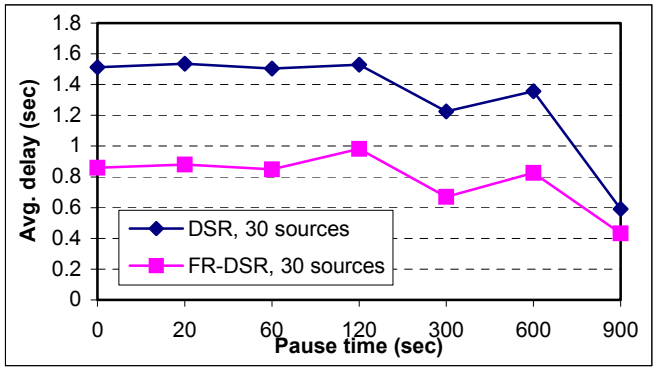
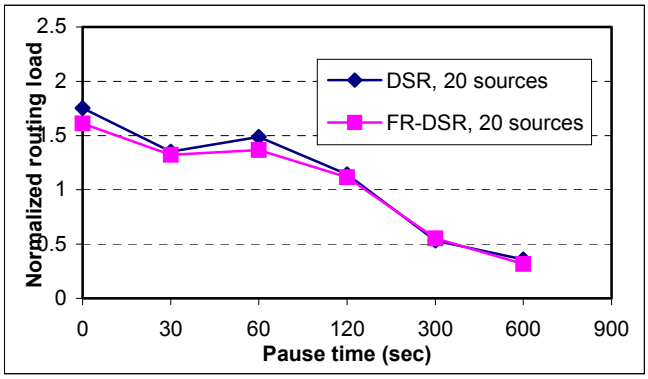


Fig. 6. Average Data Packet Delays



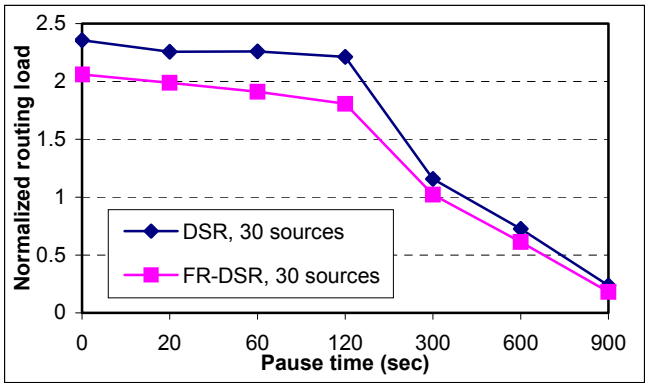


Fig. 7. Normalized Routing Loads

6. Conclusions and Future Work

In this paper we have studied transient behavior of DSR protocol in case of path failures and presented an improvement scheme called Failure Record. Specifically, we observed that in a single channel ad hoc network, path failures often cause packets en route using failed links block the interface queue at the upstream node of the failed link. This queue blocking will decrease the performance of the network. The Failure Record scheme proposed in this paper prevents such packets from blocking the queue by making every node have a short "memory" of the past and drop the packets before they enter the queue. Simulation results show that our scheme can significantly improve the performance of DSR. Our future work is intended to employ an adaptive failure record expiration time to further improve the scheme and provide an alternate path for packets en route to further reduce packet loss in case of failures.

References

[1] J. Macker and S. Corson, "Mobile Ad Hoc Networks (MANET)", *IETF WG Charter*., http://www.ietf.org/html.charters/manet-charter.html, 1997

[2] C. E. Perkins and P. Bhagwat, "Highly Dynamic Destination Sequenced Distance-Vector Routing (DSDV) for Mobile Computers", *ACM SIGCOMM*, pp.234-244, Oct. 1994

[3] S. Murthy and J.J. Garcia-Luna-Aceves, "An efficient Routing Protocol for Wireless networks", *ACM Balzer Mobile Networks and Applications Journal, Special Issue on Routing in Mobile Communications Networks*, 1996

[4] T. Clausen, P. Jacquet, J. Laouiti, P. Minet, P. Muhlethaler, A. Qayyum and L. Viennot, "Optimized Link State Routing Protocol", *IETF Internet Draft*, http://ietf.org/internet-drafts/draft-ietf-manet-olsr-06.txt, Sept. 2001

[5] B. A. Iwata, C. –C. Chiang, G. Pei, M. Gerla and T. –W. Chen, "Scalable Routing Strategies for Ad Hoc Wireless Networks", *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 8, Aug. 1999

[6] E. M. Royer and C. –K. Toh, "A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks", *IEEE Personal Communications*, April 1999

[7] D. Johnson, "Routing in Ad Hoc Networks of Mobile Hosts", *Proc. IEEE Workshop on Mobile Computing Systems and Applications*, Dec. 1995

[8] D. Johnson and D. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks", *Mobile Computing*, T. Imielinski and H. Korth, eds., Kluwer Academic Publishers, Norwell, Mass., 1996

[9] D. Johnson, D. Maltz, Y. –C. Hu and J. G. Jetcheva, "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)", *IETF Internet Draft*, http://ietf.org/internet-drafts/draft-ietf-manet-dsr-07.txt, Feb, 2002

[10] C. E. Perkins and E. M. Royer, "Ad-hoc On-demand Distance Vector Routing", *Proc. 2nd IEEE Workshop on Mobile Computing Systems and Applications*, 1999

[11] C. E. Perkins, E. M. Royer and S. R. Das, "Ad-hoc On-demand Distance Vector (AODV) Routing", *IETF Internet Draft*, http://ietf.org/internet-drafts/draft-ietf-manet-aodv-10.txt, Jan. 2002

[12] V. D. Park and M. S. Corson, "A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks", *Proc. IEEE INFOCOM'97*, April, 1997

[13] V. D. Park and M. S. Corson, "Temporally Ordered Routing Algorithm (TORA) Version 1 Functional Specification", *IETF Internet Draft*, http://ietf.org/internet-drafts/draft-ietf-manet-tora-spec-04.txt, July, 2001

[14] Y. –C Hu and D. B. Johnson, "Implicit Source Routes for On-Demand Ad Hoc Network Routing", *ACM MobiHoc* 2001

[15] W. Wang and D. Tipper, "Fault Recovery Routing in Wide Area Packet Networks", *Proceedings of 15th International Teletraffic Congress*, Washington, DC, June 1997.

[16] Kevin Fall and Kannan Varadhan (Eds.), "The ns Manual", 2002, available from http://www-mash.cs.berkeley.edu/ns/

[17] S. R. Das, C. E. Perkins and E. M. Royer, "Performance Comparison of Two On-demand Routing Protocols for Ad Hoc Networks", INFOCOM, 2000

[18] W. Lovegrove, J. Hammond and D. Tipper, "Simulation Methods for Studying Nonstationary Behavior of Computer Networks", *IEEE Journal on Selected Areas in Communications*, vol. 8, no. 9, Aug. 1990