Improving TCP Performance in Ad Hoc Networks using Signal Strength based Link Management

Fabius Klemm[1], Zhenqiang Ye[2], Srikanth V. Krishnamurthy[3], Satish K. Tripathi[4]

[1]School of Computer and Communication Sciences,

Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland

Fabius.Klemm@epfl.ch

[2]Department of Electrical Engineering,

University of California, Riverside, CA 92521

zye@cs.ucr.edu

[3]Department of Computer Science and Engineering,

University of California, Riverside, CA 92521

krish@cs.ucr.edu

[4]University at Buffalo,

State University of New York

tripathi@engr.ucr.edu

Abstract

Mobility in ad hoc networks causes frequent link failures, which in turn causes packet losses. TCP attributes these packet losses to congestion. This incorrect inference results in frequent TCP retransmission time-outs and therefore a degradation in TCP performance even at light loads. We propose mechanisms that are based on signal strength measurements to alleviate such packet losses due to mobility. Our key ideas are (a) if the signal strength measurements indicate that a link failure is most likely due to a neighbor moving out of range, *in reaction*, facilitate the use of temporary higher transmission power to keep the link alive and, (b) if the signal strength measurements indicate that a link is likely to fail, initiate a route re-discovery *proactively* before the link actually fails. We make changes at the MAC and the routing layers to predict link failures and estimate if a link failure is due to mobility. We also propose a simple mechanism at the MAC layer that can help alleviate *false link failures*, which occur due to congestion when the IEEE 802.11 MAC protocol is used. We compare the above proactive and reactive schemes and also demonstrate the benefits of using them together and along with our MAC layer extension. We show that,

This work was partially funded by DARPA under contract number: F30602-01-2-0535 and NSF CAREER Award: ANI-0237920. A preliminary version of this paper appeared in IFIP Personal and Wireless Communications (PWC), Venice, Italy, 2003.

in high mobility, the goodput of a TCP session can be improved by as much as 75% at light loads (when there is only one TCP session in the network) when our methods are incorporated. When the network is heavily loaded (i.e., there are multiple TCP sessions in the network), the proposed schemes can improve the aggregate goodput of the TCP sessions by about 14% - 30%, on average.

Key Words: **Power Management, Ad Hoc Networks, TCP, Signal Strength, Cross Layer Design**

I. Introduction

TCP performs poorly in wireless ad hoc networks as demonstrated in [1][2][3][4][5][9][10]. The main reason for this poor performance is a high level of packet losses and a resulting high number of TCP retransmission time-outs. First, a node drops a packet if it cannot forward the packet to the next hop of the route on which the packet is to be relayed, as the next hop node has moved out of transmission range. A second reason for packet loss is congestion in the shared medium. In the second case, a node cannot reach the next hop node because there are too many nodes trying to access the channel at the same time. The contention could even result in a single node capturing the medium, if the IEEE 802.11 MAC protocol is used [5]. While congestion can degrade the observed performance of TCP even in wire-line networks, mobility causes a degradation of performance of TCP in ad hoc networks even at very light loads.

Our objective in this paper is to mainly stem the degradation in TCP performance due to mobility. Towards this goal, we propose mechanisms to reduce the number of packet losses. These mechanisms are based on signal strength measurements at the physical layer. Based on these signal strength measurements, when a node fails to communicate with a neighbor, the MAC layer at the node estimate whether the failure is due to congestion or due to the neighbor moving out of range. If the MAC layer deems that the neighbor has just moved out of range, then, it stimulates the physical layer to increase the transmission power and attempts to temporarily keep the link to the neighbor alive. It also prompts the routing layer to search for a new route. The signal strength measurements can also be used to predict possible link failures to a neighbor that is about to move out of range. Thus, if the measurements indicate that the signal strength is diminishing and the link is likely to break, a search for a new route can be proactively initiated before the link actually fails. While searching for the new route, the routing layer should take care to avoid temporary high power links as well as weak links (links that might fail soon). We modify the Ad-hoc On-demand Distance Vector (AODV) routing protocol [14] such that it precludes the use of such links when searching for a new route. In order to cope with failures that are due to congestion, we propose a simple mechanism by which, the MAC layer, based on its estimate of whether a neighbor is still within range, *persists* in its attempt to reach that neighbor for a longer period of time. We reiterate that our goals are mainly to cope with the effects of mobility on TCP performance. If the network is heavily loaded, it is more likely that congestion dominates packet losses. In the simulation experiments that we perform to evaluate our schemes, we observe that when the network is lightly loaded and node mobility is high, our schemes can improve the performance of a TCP session by as much as 75%. When the network is heavily loaded, our schemes can still improve the aggregate TCP goodput by about 14%-30%. The schemes that we propose can be used with the User Datagram Protocol (UDP) as well. However, since the effects of mobility on UDP are unlikely to be as profound as on TCP we do not consider UDP in this paper.

The use of signal strength and a count of the transmitted packets in the local neighborhood (nodes can overhear other packet transmissions) can provide an estimate of whether there is congestion in the local vicinity of a node. A node should only increase its transmission power if the network is not heavily loaded. If there is heavy congestion, temporary increases in power levels can actually increase the number of collisions and increase the congestion. This could degrade the performance further. However, congestion estimation mechanisms are focus of further study and are beyond the scope of this paper.

The rest of this paper is organized as follows: In Section II, we present related work on improving TCP performance in wireless ad hoc networks. In Section III, we discuss the reasons for packet loss in wireless ad hoc networks and the effects of such losses on TCP performance. In Section IV we describe our proposed methods that help reduce packet losses in ad hoc networks. Section V presents our simulation setup and provides a discussion of our simulation results. Finally, we present our conclusions in Section

VI.

II. Related Work

Various approaches have been proposed to improve TCP performance at the transport layer [1][2][3][4][6][7]. In [1][2][3] and [4], explicit link failure notifications are used to freeze TCP state upon the occurrence of a route failure. Explicit route establishment notifications are used to resume TCP transmissions when a new route is established. A fixed-RTO approach is proposed in [6] to deal with packet losses due to link failures and route changes. In [7], a new transport layer protocol that is based on end-to-end rate control is proposed. Various mechanisms have also been proposed to improve TCP performance at the routing layer [8][9][10][11][12]. The COPAS protocol [8] uses node-disjoint paths for TCP-DATA packets in the forward direction and the TCP-ACK packets in the reverse direction to eliminate interference between TCP-DATA packets and TCP-ACK packets of the same TCP session. In contrast to COPAS, the authors of [10] propose the use of the same route for both TCP-DATA and TCP-ACK packets in order to reduce the total number of links that may stall the connection. A Route Failure Prediction (RFP) scheme is also proposed in [10] to predict the occurrence of a link failure based on the trends observed in the signal strengths of packet receptions from neighbors. In [9], the authors propose to split long TCP sessions into multiple segments. By doing so, even if a link failure occurs in one of these segments, data flow can be sustained on other segments. In [11], it is shown that the use of multiple paths, concurrently, does not help in improving TCP performance. The authors of [11] propose using the shortest path as the primary path and the shortest delay path as a backup path to improve TCP performance. In [12], it is observed that a preemptive routing scheme, in which, link failures are predicted before they actually break, can improve TCP performance. In [13], the authors propose Link-RED and adaptive-pacing approaches for improving TCP performance. The Link-RED mechanism marks or drops packets when the number of MAC layer retries exceeds a certain threshold, which in turn is taken to be an indication of congestion. The adaptive-pacing approach introduces an additional MAC layer delay equal to one packet transmission time in order to alleviate inter-packet interference.

In all the aforementioned previous work, packets in transit are dropped if a route breaks (either due to mobility or due to congestion). None of the approaches salvage transit packets. The loss of transit packets can severely degrade TCP performance. Our proposed framework can salvage the packets in transit if a route breaks either due to mobility or due to congestion. We point out that one of the schemes considered (the proactive link breakage prediction scheme) is similar to that in [10] and [12]. However, the proactive method is only one of the components in our framework. While the scheme by itself does provide certain benefits, the combination of the various components of our framework provide significant better performance benefits.

III. Packet Losses in Ad Hoc Networks

Packet losses affect TCP performance. Node mobility and link layer congestion are the two main reasons for packet losses. A link failure on an active TCP path due to mobility causes the MAC protocol to report a link failure to the routing layer. The routing layer will then have to re-compute routes to the appropriate destinations. With the IEEE 802.11 MAC protocol [15], which is the popularly advocated MAC protocol for ad hoc networks, *false link failures* may be induced when congestion occurs. Since our methods should be invoked only when there is a true link failure due to mobility, it is important to correctly identify such failures.

A false link failure occurs when the MAC protocol at a node, say N_0, declares that the link to a neighbor N_1 is broken, even though N_1 is within its transmission range [17]. The MAC protocol at N_0 fails to establish an RTS-CTS handshake because N_1 cannot respond to N_0's RTS messages since it senses another transmission in its vicinity. This failure is a direct result of the following: In the models used, it is typically assumed that each node has a transmission range of 250 meters and an interference range of 550 meters[1]. Nodes that are within the *transmission range* of a node N_0, can receive packets from N_0. Nodes that are not within the transmission range but are within the *interference range* of N_0 can sense a transmission from N_0 but cannot successfully receive packets from it. These nodes are also precluded

[1]Such values are default setups in *ns-2* for the transmission range and interference range.

from performing transmissions if N_0 is in the process of transmitting a packet. Thus, they will have to ignore any RTS control packets that they may receive from other neighbor nodes.

At the network layer, the routing protocol has to react appropriately to route failures. When the MAC layer reports a link failure to AODV [14], it simply drops the packets that are to be routed on the failed link. Furthermore, AODV brings down the routes to destinations that include the failed link and sends a *route error* message to the source of each connection that uses the failed link.

IV. Reducing Link Failures to Improve TCP Performance

We propose mechanisms that help alleviate packet losses due to mobility. Our mechanisms are based on measuring the signal strength at the physical layer. As pointed out in Section III, it is important to first estimate whether a link failure is caused by mobility or by congestion. False link failures, which we discussed earlier, cannot be overcome by tuning power levels. We propose a simple way to identify and cope with false link failures. The methods we propose, however, only work when the level of congestion in the network is not high and will have to be complemented by other techniques that can estimate the level of congestion in the network. However, we justify the intuitions behind our approach via simulation experiments (in Section V-C). The design of smart techniques to estimate the level of congestion in the network is beyond the scope of this paper and is a topic for future research.

A. Reducing False Link Failures

The IEEE 802.11 MAC protocol reports a link failure if it cannot establish an RTS-CTS handshake with a neighbor within seven RTS attempts [15]. Our idea is to double the number of retransmission attempts if there is a high probability that the neighbor is still within transmission range. We call our version of the MAC protocol (with this increased number of RTS attempts) the *Persistent MAC*.

In order to determine whether a node is still within range, a node keeps a record of the received signal strengths of neighboring nodes. Received signal strength measurements are taken at the physical layer. When a node receives a packet from a neighbor, it measures the received signal strength P_r. The node then observes how P_r changes over time. These signal strength measurements would provide an indication of whether a neighbor is still within range.

For our implementation with the network simulator *ns-2*[2] we used the received signal strength P_r to calculate the distance to the transmitter of the packet. *ns-2* uses the *two-ray ground propagation model* described in [19]. Using this propagation model, the distance d to the transmitter of a packet can be calculated as follows:

$$d = \sqrt[4]{\frac{P_t \cdot G_t \cdot G_r \cdot \ {h_t}^2 \cdot {h_r}^2}{P_r \cdot L}}, \tag{1}$$

where, P_t is the default transmission power and P_r the received signal power; G_t and G_r are the antenna gains of the transmitter and the receiver respectively; h_t and h_r are the heights of the antennas, and L is the system loss, which is set to 1 by default. We assumed that the network is homogeneous, i.e., all nodes use the same parameters P_t, G_t, G_r, h_t, h_r and L. If a node transmits with a different signal power P_t, it must include the value of P_t in the options field of the MAC protocol header.

The MAC protocol keeps a record of the distances to neighboring nodes in a *neighbor table*. A table entry consists of five fields: a neighbor ID, a distance d_1 to the neighbor (estimated using (1)), the time t_1 at which this distance was estimated, and a distance d_2 to the same neighbor estimated at a second, more recent time t_2. When a node receives a packet from a neighbor N_Y, it replaces the older entries of the table, corresponding to that neighbor, with the more recent ones. For simplicity, in our models, we use only two timestamps and assume linear node movement. Thus, at any given time t, we estimate the current distance d_{est} as follows:

$$d_{est} = d_2 + \frac{d_2 - d_1}{t_2 - t_1} \cdot (t - t_2), \text{ for } t_1 < t_2 < t \text{ and } d_1, d_2 \geq 0\ . \tag{2}$$

If a node N_X cannot establish an RTS-CTS handshake with a neighbor N_Y, it uses the neighbor table to estimate the current distance to N_Y. If d_{est} is smaller than the transmission range of N_X, *Persistent MAC* will transmit up to seven[3] additional RTSs to establish a handshake with N_Y. If d_{est} is greater than

[2]The version that we use is ns-2.1b8a.

[3]This number will typically be a system parameter. The choice will depend on the density of the network and the congestion levels.

the transmission range, or if the information in the neighbor table with regard to N_Y is deemed stale, the *Persistent MAC* will report a link failure to the routing protocol. The *Persistent MAC* will also report a link failure if the additional attempts to establish a handshake with N_Y fail. Note that the increase in the number of RTSs is not likely to increase the actual congestion significantly since the RTS messages are sent only if the channel is sensed idle. Furthermore, for each RTS failure, the node still continues its back-off process, which, in turn, would give ample time for the congestion to abate[4].

One may argue that if the congestion persists for a long time, increasing the number of RTS retries may not help much in salvaging packets since if the seventh RTS fails, it is likely that the eighth RTS will fail too (due to persistent congestion). We show later in Section V-C via simulations that this hypothesis is in fact untrue and the success or failure of an RTS attempt, seems to be independent of the success or failure of previous RTS attempts. We reiterate that increasing the number of RTS attempts is a simple method to reduce the number of false link failures. During periods of high load, a more sophisticated method may be required to estimate and deal with congestion.

We also note that the linear model to estimate distances is a simple method used to evaluate our mechanisms. We expect that the absolute value and the gradient of the received signal strength might be indicative of whether a node is moving out of range and may even be more realistic in practice. However, one might expect similar results with such methods.

B. Signal Strength based link management methods

We propose two mechanisms for alleviating the effects of mobility on TCP performance. We call these the *Proactive* and the *Reactive Link Management (LM)* schemes. These schemes are implemented at the MAC layer. We also provide a modification of AODV at the network layer that can exploit the presence of the link management schemes. *Proactive LM* tries to predict link breakage, whereas *Reactive LM* temporarily keeps a broken link alive with higher transmission power to salvage packets in transit.

[4]We perform simulation experiments to demonstrate this and discuss these in Section V.

The modified AODV allows the forwarding of packets in transit on a route that is going down while simultaneously initiating a search for a new route.

1) Proactive Link Management: The idea of *Proactive LM* is to inform the routing protocol that a link is going to break before the link actually breaks. The *link break prediction mechanism* uses the information from the neighbor table described in Section IV-A. *Proactive LM* estimates the projected distance to a neighbor in the immediate future. For example, if the current time is t, the distance $d_{0.1}$ of a particular neighbor at $(t + 0.1)$ seconds is:

$$d_{0.1} = d_2 + \frac{d_2 - d_1}{t_2 - t_1} \cdot (t + 0.1 - t_2), \text{ for } t_1 < t_2 < t \text{ and } d_1, d_2 \geq 0 \ . \quad (3)$$

Proactive LM informs the routing layer as soon as $d_{0.1}$ is estimated to be greater than the transmission range. The routing protocol then informs the packet source, which stops sending packets and initiates a route discovery[5]. In this example, packets in transit have 0.1 seconds to traverse the *weak* link. If the warning comes too late, the weak link breaks before all the packets in transit can be salvaged. On the other hand, the warning should not come too early as we want to use the link as long as possible.

2) Reactive Link Management: Reactive LM temporarily increases the transmission range of a node to reestablish a broken link. Packets in transit can then traverse the reestablished *high power link*. A node N_X tries to set up a high power link if the RTS-CTS handshake (to a neighbor N_Y) with *default* transmission power is not successful. N_X therefore sends RTSs with high transmission power. N_Y must also switch to high transmission power to send the CTS. Otherwise, the CTS would not reach N_X. The RTS frame must therefore contain the value of the transmission power P_t. N_X and N_Y also send the DATA and the ACK packets with high power. When *Reactive LM* at N_X establishes the temporary high power link to N_Y, it stimulates the routing protocol to begin a new route discovery.

Reactive LM maintains a table that records the default and the high power links. A node should be able to change its transmission power quickly, because it should not use high transmission power to

[5]We will describe in detail how the modified AODV reacts to the notifications from Proactive LM in Section IV-B.3.

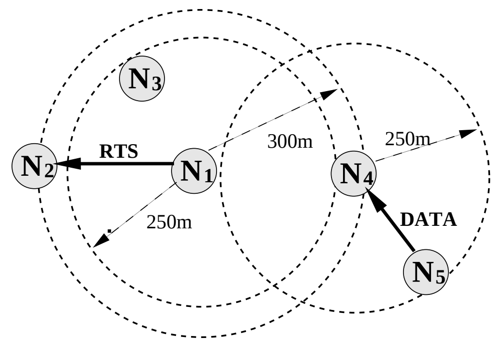

Fig. 1. Effects of power asymmetry

communicate with neighbors that are within default transmission range. Furthermore, nodes must not broadcast *Route Request* messages from AODV with high transmission power since new routes should consist of default power links only. As *Reactive LM* does not use signal strength to estimate the distance to a neighbor, it also raises the transmission power in case of false link failures. However, we combine the scheme with the Persistent MAC, and this helps alleviate this effect to a large extent.

The IEEE 802.11 MAC Collision Avoidance (CA) mechanism does not work well if nodes have different transmission ranges [16]. Figure 1 shows an example, in which N_1 increases its transmission range from 250 to 300 meters to reestablish the link to N_2. N_1 does not know about the DATA transmission from N_5 to N_4, because it could not receive N_4's CTS. N_1 therefore disturbs the DATA transfer from N_5 to N_4. Due to this effect, as congestion in the network increases, increasing the transmit power can in fact degrade performance (We shall see the performance of the proposed schemes with different traffic loads in Section V). Thus, this method should be incorporated only when the network is not heavily loaded.

3) Modifications to AODV: Proactive and *Reactive LM* inform the routing protocol of either weak or high power links. In this subsection, we explain how our modified version of AODV reacts to these MAC layer notifications. The routing protocol does not necessarily have to distinguish between *weak* and *high power* links. In both cases, the objective is to inform the packet source of the link failure, initiate a new route discovery, and to salvage packets in transit. In AODV, a route to a destination N_D in the routing table of a node N_X can be in either of two states: The route can be UP, which means N_X forwards packets

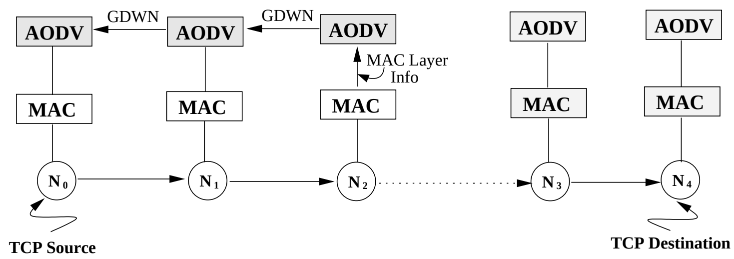

Fig. 2. Modifications to AODV

to N_D. If N_X receives a Route Request (RREQ) for N_D, it will respond with a Route Reply (RREP) because it knows a route to N_D. The second state is DOWN; if the route is in this state, N_X does not have a route for N_D. If N_X wants to send packets to N_D, it will initiate a route discovery. If N_X receives an RREQ for N_D, it will broadcast the RREQ. If N_X receives a packet for N_D, it will drop the packet and respond with a Route Error (RERR).

We added an additional route state to allow nodes to salvage packets in transit. We call this the *Going Down (GDWN)* state. The GDWN state has the following characteristics: If N_X receives a packet for N_D, it will forward the packet. If it receives an RREQ for N_D, in lieu of responding with a RREP it broadcasts the RREQ. If an application at N_X wants to send packets to N_D, the modified AODV will initiate a route discovery. Figure 2 shows an example, in which the MAC protocol at N_2 informs the modified AODV that the link to N_3 is getting weak (or has become a temporary high power link). The modified AODV then, sends a GDWN packet to its active neighbor N_1, which also sends a GDWN packet to N_0. All three nodes N_0, N_1, and N_2 change the route state for destination N_4 from UP to GDWN. N_1 and N_2 keep forwarding transit TCP packets towards N_4, but N_0 stops sending packets and initiates a route discovery. The old route to N_4 via N_1, N_2, and N_3 is then no longer used and finally times out, i.e., the route state is set to DOWN. If the MAC protocol reports a link breakage, the modified AODV behaves like the original AODV, i.e., it brings down the route to destination N_4, sends RERR messages to its active neighbors, and drops all packets in transit to N_4.

4) Transport Layer: The methods we presented in the previous subsections are aimed at reducing the number of packet drops. TCP Tahoe, Reno, and New Reno grow the congestion window until packets are dropped. In wireless ad hoc networks, congestion does not lead to buffer overflow very often as in wired networks, but rather to false link failures, which cause the routing protocol to bring down the route. Therefore, even in static networks, where we would expect stable routes, the excessive growth of the TCP congestion window and false link failures cause repeated route changes. This behavior was shown by Saadawi and Xu [5], who quantified the performance of several versions of TCP in ad hoc networks. They showed that TCP Tahoe, Reno, and New Reno suffer from the "instability problem" due to the excessive growth of the congestion window. They suggested restricting the maximum window size. They also showed that *TCP Vegas* does not suffer from this instability problem, because it uses a more conservative mechanism with *Round Trip Time (RTT)* estimations to control the size of the congestion window. TCP Vegas does not need packet losses to stop the growth of the congestion window. Since our goal is to study the effects of mobility as opposed to congestion, we used TCP Vegas for our simulations with *ns-2* [18][20].

V. Simulations and Discussion

Figure 3 depicts the simulation scenario that we used in our simulations. The scenario consists of 50 mobile nodes and a certain number of static TCP end nodes (TCP sources and TCP sinks). The 50 mobile nodes move in a rectangular area of 300 by 1500 meters according to the *random way-point model*. The pause time is zero seconds and the maximum speed of the mobile nodes is set to 0, 4, 8, 12, 16, and 20 m/s for different simulation runs. In order to avoid some of the potential pitfalls of the random way-point model [21], we modified the $ns-2$ random way-point generation code to set the minimum speed to be 10% of maximum speed, rather than the default value of 0 m/s. In each simulation run, in order to reach a steady state, the system warms up for 300 seconds before the TCP sessions are established. In order to test the mechanisms in scenarios of high mobility, we also perform certain experiments wherein the minimum possible speed is chosen to be 50% of that of the maximum speed (specified explicitly when the experiments are discussed). The TCP end nodes are placed at the edges of the rectangular area and they are static during the simulations. We consider three setups in our simulations. We call these setup I, II, and III, respectively. In setup I, there is one TCP connection between two static nodes. Setup II has three TCP connections (with data flows in different directions) that cross each other, between four static nodes. Similarly, Setup III has five TCP connections that cross each other, between ten static nodes.

The traffic carried by each TCP connection is a file transfer of infinite length, i.e., a TCP source will send TCP data packets for the entire duration of the simulation. The default transmission range of each node is 250 meters and the interference range is 550 meters. A TCP packet travels about 8 hops, on average, to get from a source to the corresponding sink. All TCP sessions last for 600 seconds.

Since the interference range of a node is set to 550 meters in our simulations, when a node is transmitting, a large number of its neighbor nodes (within its interference range) are prevented from initiating transmissions. We see that even with only one TCP connection in setup I, the medium in the whole simulation area will be almost saturated, i.e., the medium is busy for most of the time. If we further increase the number of TCP connections as in setups II and III, the spatial diversity benefits are limited. All the TCP connections contend with each other for the medium in the same region. That is to say, increasing the number of TCP connections causes higher levels of contention. As compared with the *light* traffic load in setup I (one TCP connection only), the traffic load is *moderate* in setup II, and is *heavy* in setup III.

Proactive LM notifies the routing layer when the distance estimate to a neighbor at *(current time + 0.1)* seconds is greater than the default transmission range. This time seems appropriate since we do not want it to be too long (routes are left unused even when the link is fairly stable). The temporary high

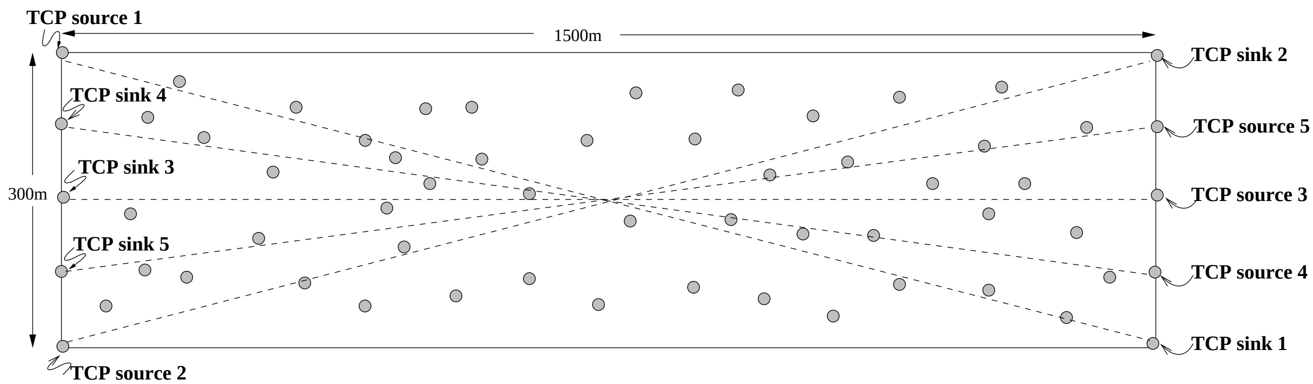

Fig. 3. Simulation Scenario

power transmission range for *Reactive LM* is set to 275 meters in our simulations[6] We use the following metrics to quantify the performance of TCP:

- *Packet loss:* Ratio of the number of dropped TCP packets to the total number of TCP packets being injected by the TCP sources.
- *TCP goodput:* Number of TCP data packets received by the application layer at the TCP sinks. Note that retransmitted packets are not counted while computing the goodput.
- Number of TCP retransmission time-outs per delivered packet.

In the following two subsections, we shall evaluate our protocols by comparing the performance of TCP with the following schemes at the link layer: (1) The *Original* scheme, which is the unchanged version with the IEEE 802.11 MAC, (2) *Persistent MAC* only, (3) *Proactive LM* only, (4) *Reactive LM* only, and (5) *a Combined* scheme, which includes all of the above methods, i.e., the *Persistent MAC*, the *Proactive LM*, and the *Reactive LM*. For the *Original* scheme, we used the original AODV and for versions (2) to (5), our modified variant of AODV was used. Each point in a graph represents an average

[6]We also perform simulations with various other values chosen for the higher power transmission range. The simulation results show that if the increase in transmission range is too small, it does not help much in salvaging transit packets during mobility. On the other hand, if the increase in the transmission range is large, it causes unacceptable levels of interference and packet collisions in the network, and thus undermines the performance of the *reactive LM* scheme. In reality, this is a system parameter and should be set depending upon the scenario of deployment.

TABLE I

SUMMARY OF SIMULATION PARAMETERS

Parameter	**Value**
Simulation time	600 seconds
Number of mobile nodes	50
Number of static nodes	2 in setup I, 6 in setup II, 10 in setup III
Default transmission range	250 meters
Proactive LM time to link breakage	0.1 seconds
Reactive LM high power range	275 meters
Traffic	
Type	FTP with infinite backlog over TCP
Packet size	1460 bytes
Number of TCP connections	1 in setup I, 3 in setup II, 5 in setup III
Movement	
Pause time	0 second
Maximum speed	0, 4, 8, 12, 16, and 20 m/s

of 300 simulation runs with different random movement patterns described earlier.

Figure 4 plots the packet loss as a function of maximum speed with setup I. The original scheme drops between 2 to 13 percent of the packets. *Persistent MAC* can significantly reduce packet loss in scenarios with low mobility, in which contention-induced link failures dominate. There is no improvement with the *Proactive LM* in static scenarios as all packet losses are caused by false link failures. When node mobility increases, *Proactive LM* can approximately reduce the number of losses by half. *Reactive LM* and a *Combined* scheme can reduce the packet loss to less than 2 percent in all mobile scenarios.

Figure 5 shows the effects of decreased packet loss on TCP goodput with setup I. *Persistent MAC*, *Proactive LM*, *Reactive LM*, and the *Combined* scheme can increase the TCP goodput, especially in high mobile scenarios. The reason for the higher goodput is the decreased number of TCP retransmission time-outs. Figure 6 shows the number of TCP retransmission time-outs per delivered data packet. With

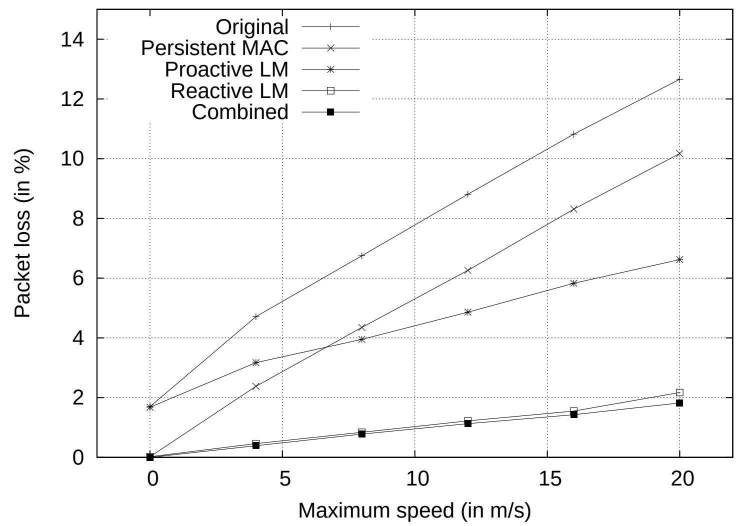

Fig. 4. Performance of the various schemes with one TCP session

the *Original* scheme, TCP times out about 6 times per 100 delivered data packets in scenarios of high mobility, whereas with the *Combined* scheme, there is on average only about 2 time-outs per 100 delivered data packets.

A. *Effects of Traffic Load*

Figure 7 compares the packet losses with the *Original* scheme and the *Combined* scheme for one, three and five TCP connections. With three and five TCP connections, the percentage of dropped packets is higher than with one TCP connection. The reason for this increase in packet loss is increased link layer contention, which leads to a higher percentage of false link failures (Figure 8). At higher levels of congestion, the percentage of dropped packets increases. Thus, the proposed schemes are beneficial primarily at light loads wherein mobility is predominantly responsible for link failures (see Figure 9).

Figure 9 shows the goodput improvement enjoyed by TCP with the *Combined* scheme with one, three and five TCP connections. The total goodput improvement with three and five TCP connections is lower than that observed with one TCP connection, except when node mobility is low. With increased network contention, it is more difficult for the *Proactive* and *Reactive LM* schemes to salvage packets in transit

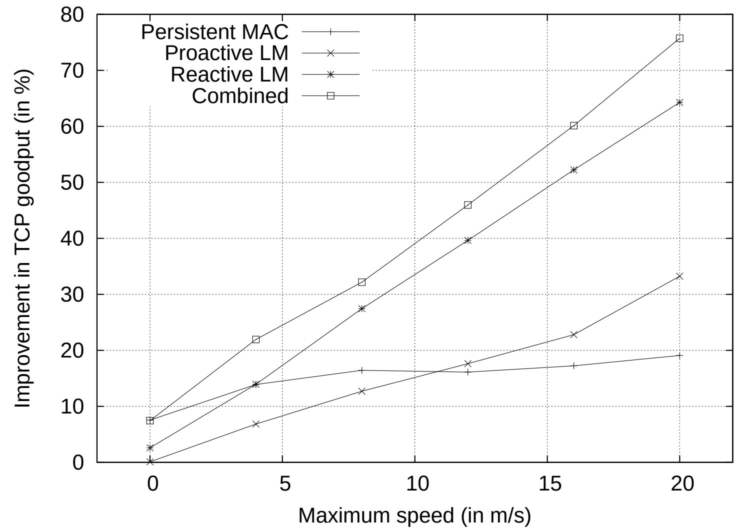

Fig. 5. Improvement in TCP goodput versus maximum speed for one connection

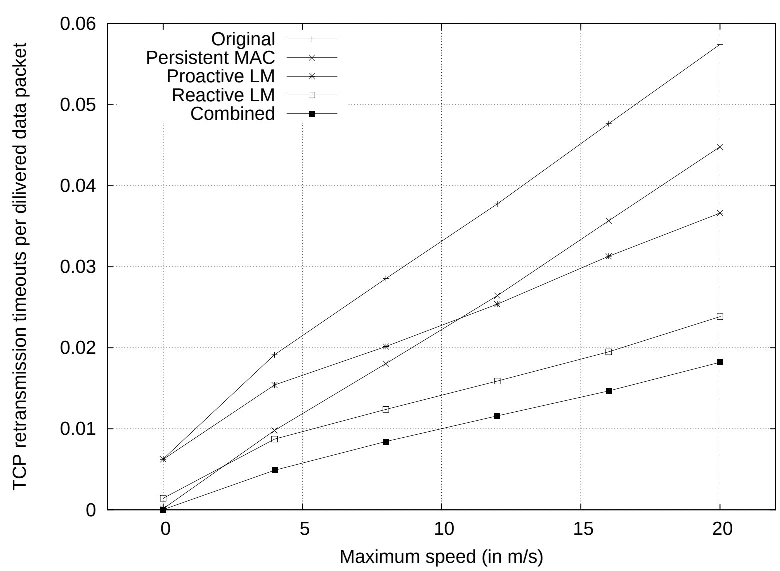

Fig. 6. TCP retransmission time-outs per delivered packet as a function of maximum speed

as it takes a longer time for these packets to traverse the weak or high power link.

Figures 10 and 11 show the *improvement ratio* in TCP goodput with three TCP connections and with five TCP connections respectively. The improvement ratio is defined to be the ratio of the total goodput achieved by TCP with a particular scheme to that achieved with the original AODV protocol and the standard IEEE 802.11 MAC protocol in place. We see that the benefits due to the *Proactive LM* and

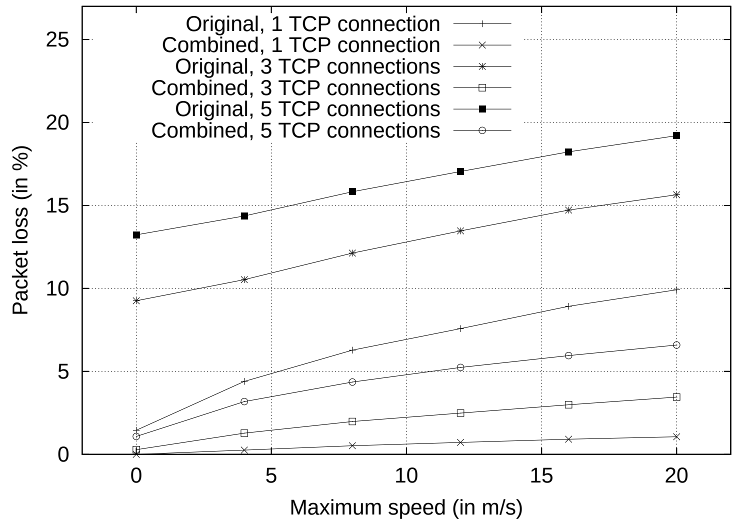

Fig. 7. Comparison of packet losses with various number of TCP connections

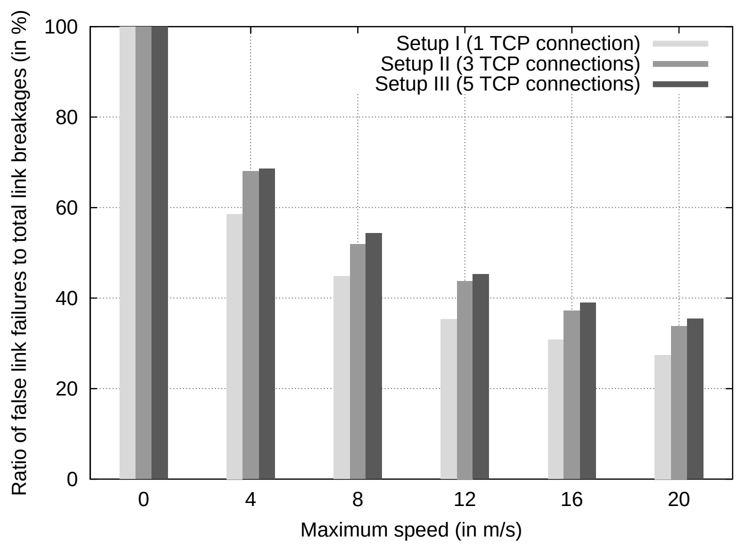

Fig. 8. Fraction of False Link Failures

Reactive LM schemes decrease with an increase in traffic load. Furthermore, when the network is heavily loaded (setup III) and node mobility is low (the maximum moving speed is less than 10 m/s), as seen in Figure 11, the *Reactive LM* scheme in fact degrades TCP goodput by about 0-3% (as analyzed in Section IV-B.2). As compared to the *Proactive LM* and *Reactive LM* schemes, *Persistent MAC* always improves TCP goodput. This is a direct consequence of the fact that congestion exists (even with a single TCP connection) with any level of mobility.

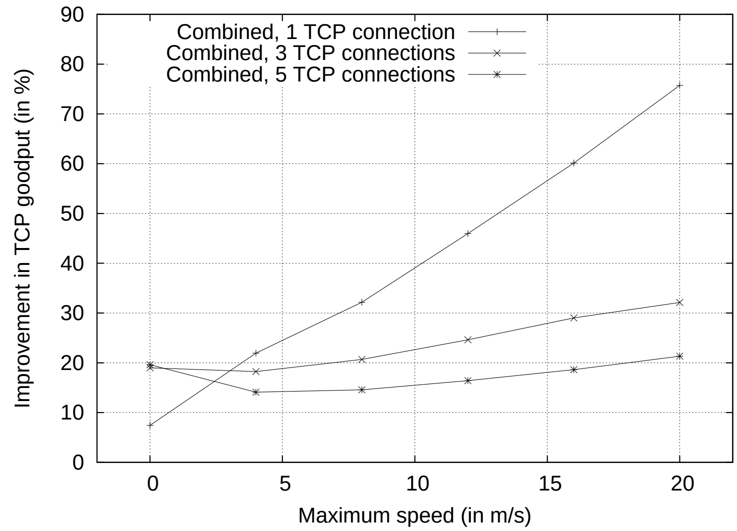

Fig. 9. Improvement in TCP goodput versus maximum speed with various number of TCP connections

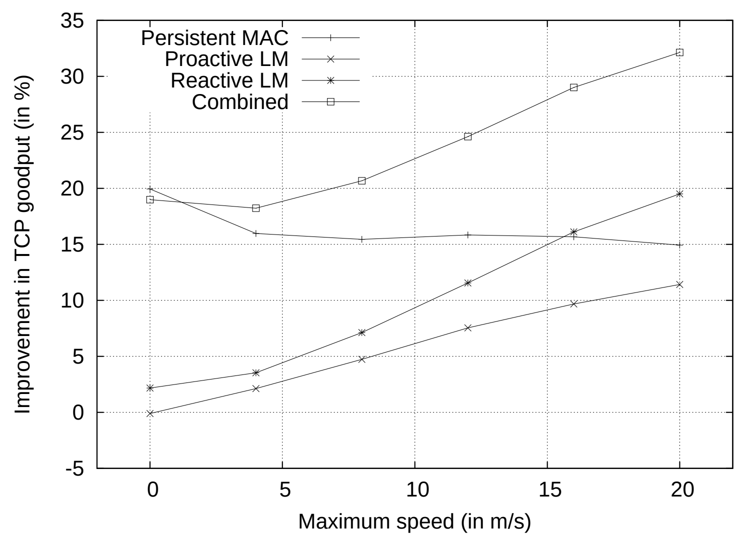

Fig. 10. Improvement in TCP goodput versus maximum speed for three TCP connections

B. Effects of Node Mobility

Since *Proactive LM* and *Reactive LM* schemes are used to stem packet losses due to mobility, the benefits that they provide are significant in scenarios of high mobility. Furthermore, the benefits due to these two schemes increase with node mobility. In order to demonstrate the benefits of these schemes in terms of helping TCP cope with mobility induced failures, we perform simulations with highly mobile

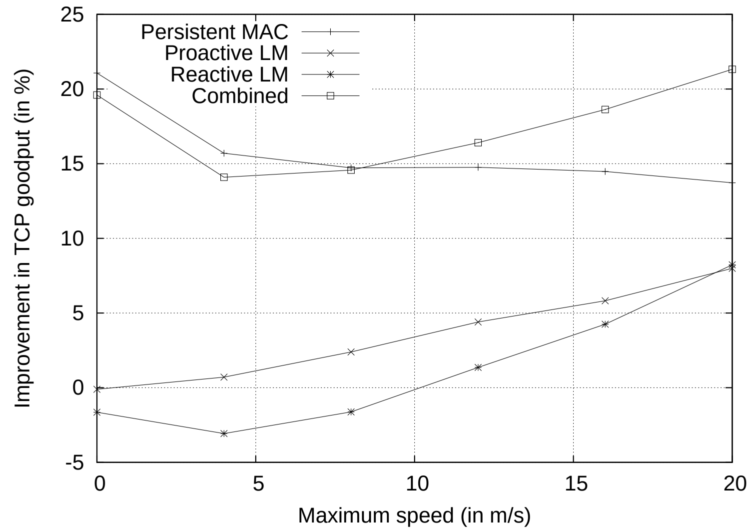

Fig. 11. Improvement in TCP goodput versus maximum speed for five TCP connections

scenarios, in which the minimum moving speed of a node is set to 50% of the maximum speed. Figure 12 shows the improvement ratio in TCP goodput in this scenario, when the network is heavily loaded (i.e., there are 5 TCP sessions in the network). We see that when node mobility is low, the benefits come mainly from *Persistent MAC*. The benefits provided by *Proactive LM* and *Reactive LM* schemes increase steadily with node mobility. In extremely high mobility, the *combined* scheme performs much better than the *Persistent MAC* or any scheme considered in isolation. In comparison, the benefits provided by *Persistent MAC* decreases with the increase in node mobility. This is a direct consequence of this scheme failing to cope with mobility induced failures. Thus, while the *reactive/proactive LM* schemes help predominantly in coping with mobility induced link failures, the *persistent MAC* primarily helps in coping with congestion induced false link failures. Together, these schemes provides a unified framework for coping with both kinds of link failures.

C. Distribution of the number of RTS attempts

As described earlier, with *Persistent MAC*, a node increases the number of RTS retransmission attempts to a neighbor, if, based on its signal strength table, it concludes that the neighbor is within range. One might hypothesize that the failure of seven RTS attempts suggests that it is futile to attempt further RTS

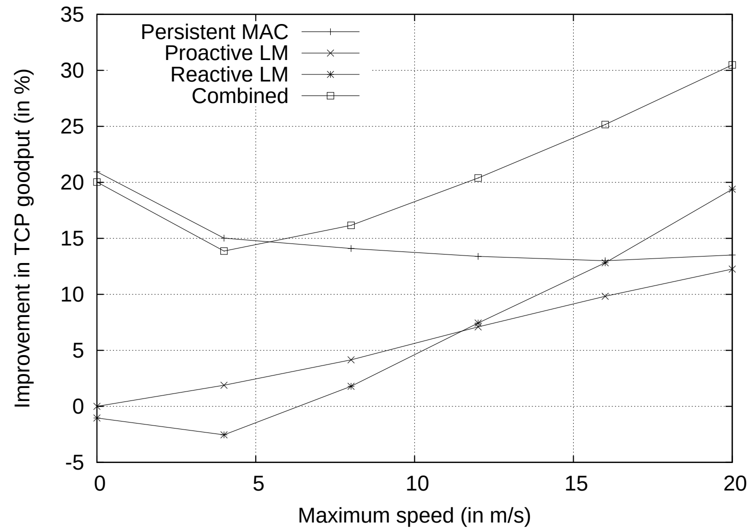

Fig. 12. Improvement in TCP goodput versus maximum speed for five TCP connections in high mobility pattern (min_speed = 50% of the max_speed).

transmissions. In order to study if the hypothesis is true, we did simulations using the scenario shown in Figure 3 with one TCP connection (which is from *Source 1* to *Sink 1*). All the nodes are static during the simulation and therefore all packet losses are due to false link failures. The TCP section lasts for 6000 seconds. All the other simulation parameters are the same as those in previous simulations. In particular, we investigate the distribution of the number of RTS attempts at *Source 1* [7]. In Figure 13, the ordinate for a certain abscissa value x, represents the fraction of data packets (that are either successfully transmitted or eventually dropped) that are preceded by x RTS attempts. Of the 96598 data packets transmitted by *source 1*, about 76% of the data packets are successfully transmitted with a single RTS attempt. The percentage of the data packets that need 2 or 3 RTS attempts are less than the percentage that need 4 RTS attempts. This may be attributed to the back-off times chosen with the IEEE 802.11 protocol being very short during the first couple of RTS attempts. The back-off time increases exponentially with the increase in RTS attempts. As per this observation, if we increase the limit of the number of RTS attempts from 7 to 14, we can expect to salvage a significant portion of the packets that will be dropped after 7 RTS

[7] We also looked at the statistics for other nodes. The distribution of the number of RTS attempts and the conditional probabilities $P(i, j)$ (as we will discuss later) were consistent with those reported for *Source 1*.

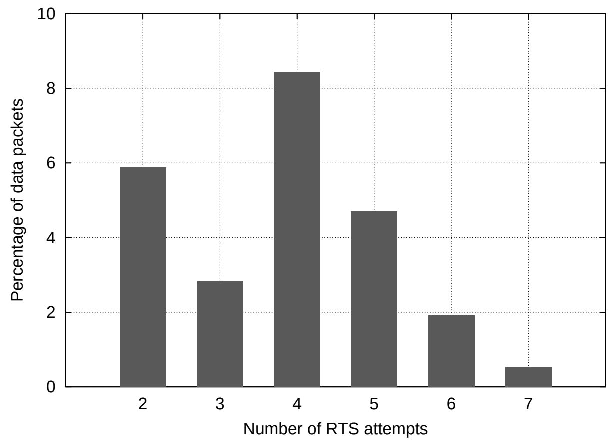

Fig. 13. Percentage of the data packets that experience various number of RTS attempts

failures since these extra RTS attempts are interleaved by long back-off times. This results in a significant chance that by the instance of an extra RTS attempt the congestion that was seen during previous RTS attempt would have subsided.

We also notice that the congestion in ad hoc networks is transient and it does not persist for a long time. More specifically, we found that the number of RTS attempts for a particular packet at a node has little correlation with that for the previous packet at the same node. Expression (4) shows the conditional probabilities $P(i, j)$ that the current packet experiences i RTS attempts given that the previous packet experiences j RTS attempts at *Source 1*. We observe that most of the packets are transmitted successfully with a single RTS attempt, no matter how many RTS attempts were made for the previous packet. A high number of RTS attempts for the previous packet does not means that the current packet will experience a high number of RTS attempts. These experimental results in turn, suggest that increasing the number of RTS attempts as with the *Persistent MAC* is in fact a reasonable way of decreasing the possibility of false link failures.

$$P(i,j) = \begin{pmatrix} 0.7276 & 0.0841 & 0.0485 & 0.0664 & 0.0442 & 0.0232 & 0.0060 \\ 0.7199 & 0.0793 & 0.0458 & 0.0775 & 0.0502 & 0.0215 & 0.0058 \\ 0.7921 & 0.0463 & 0.0271 & 0.0725 & 0.0420 & 0.0163 & 0.0037 \\ 0.7841 & 0.0538 & 0.0223 & 0.0761 & 0.0465 & 0.0132 & 0.0040 \\ 0.7870 & 0.0622 & 0.0200 & 0.0703 & 0.0357 & 0.0200 & 0.0049 \\ 0.7577 & 0.0558 & 0.0231 & 0.0731 & 0.0615 & 0.0231 & 0.0058 \end{pmatrix} \quad (4)$$

D. Summary

In summary, the three mechanisms that are proposed, independently and more significantly in combination, decrease packet losses in ad hoc networks. The reduction of packet loss results in fewer TCP retransmission time-outs and therefore higher TCP goodput. The higher the packet losses due to mobility, the greater is the improvement by the combination of the proposed mechanisms. The simulation results show that, in high mobility, the combined scheme can improve the TCP goodput by up to 75% when the network is lightly loaded and 14%-30% when the network is heavily loaded. In these simulations the TCP connections are approximately eight hops long. With shorter connections, there are fewer link failures and consequently fewer packet losses; therefore, the improvement in TCP goodput is less significant.

VI. Conclusions and Future Work

In this paper our objective is to reduce the packet losses due to mobility in ad hoc networks and thereby improve the performance of TCP. Towards this, we propose a link management framework that helps in salvaging TCP packets in transit upon the incidence of link failure. The framework consists of three individual components. First, we induce a temporary increase in the transmit power level when a node moves out of range to temporarily re-establish the failed link. This would enable the TCP packets that are already in flight to traverse the link.

The use of the IEEE 802.11 MAC protocol causes *false link failures* due to congestion. We propose a mechanism that allows us to distinguish between true link failures due to mobility and false link failures.

This mechanism is based on the measurement of signal strength at the physical layer and is used to determine if a node is still within range. We then increase power levels to temporarily reestablish a failed link only if it is determined to be mobility induced. We include a proactive scheme, in which weak links are identified based on these signal strength measurements and routes are proactively found prior to failure. This scheme in turn helps in switching to the new route even before the failure occurs and thus can stem packet losses. The proactive and reactive signal strength based schemes are unified with another simple MAC layer extension. With our extension, the MAC layer, upon perceiving false link failures, simply increases the number of RTS attempts in order to salvage transit TCP packets.

The simulation results with *ns-2* show that these mechanisms together can considerably reduce the number of packet losses. Consequently, the number of TCP retransmission time-outs is reduced and the TCP sources send more packets. The simulation results show that in high mobility, our framework can improve the performance of a TCP session by as much as 75% when the network is lightly loaded. When the network is heavy loaded, the proposed approaches can improve TCP goodput, on average, by about 14% - 30%.

We recognize that additional mechanisms are necessary to correctly determine the levels of congestion of the network. These mechanisms can help us to decide whether the *reactive LM* approach should be incorporated to salvage packets in transit since during heavy congestion and low mobility, temporary increases in transmission power can lead to some adverse effects. The design of such congestion estimation mechanisms is beyond the scope of this paper and is a topic for future study.

REFERENCES

[1] K. Chandran, S. Raghunathan, S. Venkatesan, R. Prakash, “A Feedback based Scheme for Improving TCP Performance in Ad-hoc Wireless Networks”, *Proceedings of ICDCS*, May 1998.

[2] D. Kim, C.-K. Toh and Y. Choi, “TCP-BuS: Improving TCP Performance in Wireless Ad Hoc Networks”, *Journal Of Communications And Networks*, June 2001.

[3] G. Holland and N. Vaidya, “Analysis of TCP Performance over Mobile Ad Hoc Networks”, *Proceedings of ACM MOBICOM*, 1999.

[4] J. Liu and S. Singh, “ATCP: TCP for Mobile Ad-hoc Networks”, *IEEE Journal on Selected Areas in Communications*, Vol.19, No.7, July 2001.

[5] T. Saadawi and S. Xu, “Performance Evaluation of TCP Algorithms in Multi-hop Wireless Packet Networks”, *Journal of Wireless Communications and Mobile Computing 2002*.

[6] T. D. Dyer and R. V. Boppana, “A Comparison of TCP Performance over Three Routing Protocols for Mobile Ad Hoc Networks”, *Proceedings of ACM MOBIHOC*, 2001.

[7] K. Sundaresan, V. Anantharaman, H.-Y. Hsieh and R. Sivakumar, “ATP: A Reliable Transport Protocol for Ad-hoc Networks”, *Proceedings of ACM MOBIHOC*, 2003.

[8] C. Cordeiro, S. R. Das and D. P. Agrawal, “COPAS: Dynamic Contention-Balancing to Enhance the Performance of TCP over Multi-hop Wireless Networks”, *Proceedings of the 10th International Conference on Computer Communications and Networks (IC3N)*, 2002.

[9] S. Kopparty, S.V. Krishnamurthy, M. Faloutsos and S.K. Tripathi, “Split-TCP for Mobile Ad Hoc Networks”, *Proceedings of IEEE GLOBECOM*, 2002.

[10] V. Anantharaman and R. Sivakumar, “A Microscopic Analysis of TCP Performance over Wireless Ad-hoc Networks”, *Proceedings of the ACM SIGMETRICS (poster paper)*, 2002.

[11] H. Lim, K. Xu and M. Gerla, “TCP Performance over Multipath Routing in Mobile Ad-hoc Networks”, *Proceedings of IEEE ICC*, 2003.

[12] T. Goff, D.S. Phatak and N.B. Abu-Ghazaleh, “Analysis of TCP Performance on Ad Hoc Networks Using Preemptive Maintenance Routing ”, *Proceedings of International Conference on Parallel Processing (ICPP)*,, 2001.

[13] Z. Fu, P. Zerfos, H. Luo, S. Lu, L. Zhang and M. Gerla, “The Impact of Multihop Wireless Channel on TCP Throughput and Loss”, *Proceedings of IEEE INFOCOM*, 2003.

[14] C.E. Perkins and E. M. Royer, “Ad-hoc On-Demand Distance Vector Routing”, *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications*, 1999.

[15] IEEE Standards Department, “Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications”, *IEEE standard 802.11*, 1999.

[16] N. Poojary, S.V. Krishnamurthy and S. Dao, “Medium Access Control in a Network of Ad Hoc Mobile Nodes with Heterogeneous Power Capabilities”, *Proceedings of IEEE ICC*, 2001.

[17] S. Xu and T. Saadawi, “Does the IEEE 802.11 MAC Protocol Work Well in Multi-hop Wireless Ad Hoc Networks?”, *IEEE Communications Magazine*, Vol.39, Issue 6, June 2001.

[18] L.S. Brakmo and L. Peterson, “TCP Vegas: End to End Congestion Avoidance on a Global Internet”, *IEEE Journal On Selected Areas In Communications*, Vol. 13, No. 8, October 1995.

[19] K. Fall and K. Varadhan, *The ns Manual*, 2001.

[20] *The Network Simulator - ns-2*, http://www.isi.edu/nsnam/ns/.

[21] J. Yoon, M. Liu and B. Noble, “Random Waypoint Considered Harmful”, *Proceedings of IEEE INFOCOM*, 2003.