TCP-Friendly Medium Access Control for Ad-Hoc Wireless Networks: Alleviating Self-Contention

Dan Berger[†], Zhenqiang Ye[†], Prasun Sinha[‡],
Srikanth Krishnamurthy[†], Michalis Faloutsos[†], Satish K. Tripathi[*]

[†]Dept. of Computer Science & Engineering
University of California, Riverside, CA 92521
{zye,dberger,krish,michalis}@cs.ucr.edu

[‡]Dept. of Computer Science & Engineering
Ohio State University, Columbus, OH 43210
prasun@cis.ohio-state.edu

[*]Dept. of Computer Science & Engineering
University at Buffalo, SUNY Buffalo, NY 14260
tripathi@buffalo.edu

Abstract

In this paper we focus on self-contention – contention between packets of the same transport layer connection along the path from source to destination. We observe that self-contention plays an important role in degrading TCP performance in multi-hop wireless networks and that the use of the popular IEEE 802.11 MAC protocol exacerbates self-contention.

We propose and study two MAC-layer approaches to alleviate self-contention. The first approach, called ***quick-exchange*** *(QE), is designed with the intent of reducing the effects of inter-flow self-contention (e.g. between packets of the same connection traveling in opposite directions). The design of our second mechanism, called* ***fast-forward*** *(FF), is geared towards decreasing intra-flow self-contention (e.g. between packets of the same connection traveling in the same direction).*

We simulate and study our proposed schemes and observe that quick-exchange consistently improves network aggregate goodput (by as much as 20% in string topologies, 15% in random static scenarios, and 10% in random mobile scenarios). In contrast to our expectations, fast-forward causes sporadic and often negative effects on goodput for TCP connections. Upon investigation we find that while the MAC is, in some respect, operating more efficiently, as demonstrated by improved UDP throughput; interactions with TCPs congestion control mechanism cause the goodput to degrade. We analyze various effects that cause the respective behaviors with QE and FF in detail.

This work was supported by DARPA FTN Grant #: F30602-01-2-0535 and NSF CAREER Grant #: ANI-0237920.

1 Introduction

Contention is a fundamental problem in multi-hop wireless ad hoc networks and one of the main factors contributing to poor TCP performance in such networks. In ad-hoc wireless networks, adjacent nodes impede each-other as they contend for access to the shared wireless channel. Furthermore, packets in ad-hoc networks are typically forwarded through multiple hops en route from source to destination; resulting in interference patterns not seen in centrally managed wireless networks such as wireless LANs or cellular networks. *Self-contention*, a key form of contention, has received little attention to date. This is among the first works to focus on self-contention, document it's effects, and propose ways to alleviate its impact.

We formally define self-contention to be the contention between packets of the *same* transport layer connection attempting to access the shared channel. We loosely define a **"flow"** as a stream of packets between a source node to a sink node belonging to a single transport layer connection. Applying this definition to a single TCP connection identifies two flows; the first flow consisting of the TCP-DATA packets traveling from source to sink, and the second flow consisting of the TCP-ACK packets of the same connection traveling from the sink back toward the source. Self-contention occurs both between packets traveling in opposite directions (TCP-DATA and TCP-ACK packets) – which we term *inter-flow* self-contention – as well as between packets traveling in the same direction (multiple TCP-DATA or TCP-ACK packets buffered at nodes that are within the interference range[1] of each-other) – which we

[1]The neighbors of a node a that can decode a's transmission are said to be in the transmission range of a. Other neighbors that can detect interference from a's transmission are said to be within a's interference, or sensing, range.

term *intra-flow* self-contention.

Is worth noting that the issue of self-contention is not addressed in the IEEE 802.11 standard, originally designed for wireless LANs (WLANs). In WLANs, intra-flow contention is not an issue, as it is a single-hop rather than a multi-hop network. Inter-flow contention *does* exist in WLANs as the communication channel between a user and the base-station is shared [18].

We assert that self-contention should be addressed with MAC layer mechanisms for three reasons. First, self-contention arises due to distributed access to the shared medium and is, therefore, *inherently a MAC layer phenomenon*. Second, a MAC layer solution *leaves widely deployed upper layer protocols*, such as UDP and TCP, *unchanged*. Third, the MAC protocol commonly used for multi-hop networks, *IEEE 802.11, is a rapidly evolving standard*, and is more amenable to enhancements than entrenched transport layer protocols (such as UDP and TCP).

In this paper, we propose and examine two MAC layer mechanisms to address self-contention. The first, called *quick-exchange*, targets inter-flow contention. The second, called *fast-forward*, targets intra-flow contention.

Quick-exchange allows two nodes which successfully acquire the channel to *exchange* packets, one in each direction, rather than forcing them to (re)acquire the channel for each packet. This permits the contention caused by the reverse flow of the transport connection to be subsumed. In addition, quick-exchange reduces the average number of control packets required per data packet transmission and the average amount of time nodes spend backing off to avoid packet collisions.

Our *fast-forward* mechanism allows a node to aggressively forward a received (or suitable queued) packet immediately after successful data packet reception. The intent of this behavior is to quickly spread packets in the same flow across the nodes along the path of the connection, thus reducing intra-flow contention.

Simulation results show that both schemes offer improvements in constructed static topologies. Fast-forward, however, has undesirable interactions with TCP congestion control logic and, despite providing measurable gains in UDP scenarios, fails to consistently improve aggregate network goodput in random scenarios with multiple TCP connections. Quick exchange consistently outperforms the standard MAC - providing goodput increases up to 20% in static and 10% in random mobile scenarios.

The rest of the paper is organized as follows. Section 2 presents related work on improving transport layer throughput in multi-hop networks and briefly discusses the relevant portions of the IEEE 802.11 protocol and the Distributed Coordination Function (DCF) used in the ad-hoc mode. Section 3 presents the quick exchange and fast-forward enhancements. The performance of our approach is studied using extensive simulations and the results are discussed in Section 4. Section 5 discusses issues that need further investigation. In Section 6, we conclude our paper.

2 Related Work & Background

Various mechanisms have been proposed to improve the transport layer performance of ad-hoc networks at the MAC, link, routing, and transport layers.

MAC/Link Layer Proposals:

In [11], Gerla et al. present a study of the interactions between the MAC and TCP layers. Their study, based on FAMA [10] and CSMA protocols, shows that the interaction of TCP-DATA and TCP-ACK packets can be eliminated by choosing a TCP window size of one. However, the conclusions were based on studies using only three different static topologies.

TULIP [18] proposes improvements to the link and MAC layers for Wireless LANs and more recently [24] proposes a protocol called DCF+ in their effort to improve TCP performance over 802.11. Though some of the proposed mechanisms in TULIP and the DCF+ mechanism bear resemblance to our quick-exchange approach, the study is limited to single-hop Wireless LANs. Furthermore, the TULIP studies do not include the effects of hidden terminals and interference from other transmissions. In [3], Acharya et. all propose a mechanism similar to fast forward which they call DCMA (Data-driven Cut-through Multiple Access). Though the core concept is similar, their study is restricted to UDP flows across string and grid topologies and does not consider limiting channel reservations.

In [9], Fu et. all propose two mechanisms, Link-RED and adaptive-pacing for improving TCP throughput. The Link-RED mechanism marks/drops packets when the number of MAC layer retries exceeds a certain threshold (this is taken as indicating congestion). In addition when there is congestion, the adaptive-pacing approach introduces an additional MAC layer back-off at the sensing node equal to one packet transmission time to allow traffic to be spread among intermediate nodes along the path. Thus, adaptive-pacing is targeted for reducing intra-flow self-contention.

Routing Layer Proposals:

Some routing layer mechanisms have been proposed specifically to improve TCP performance. The COPAS [6] protocol routes the TCP DATA packets (in the forward di-

rection) and TCP-ACK packets (in the reverse direction) along node-disjoint paths to eliminate interactions between TCP-DATA and TCP-ACK packets of the same connection. [15] concludes that multi-path routing can be used to improve the performance of TCP only if the routes are selected carefully. [4] proposes physical layer and routing layer mechanisms for reducing the effect of link failures on TCP throughput and, in contrast to COPAS, proposes 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.

Transport Layer Proposals:

Various approaches have been proposed at the transport layer for improving TCP performance. The explicit link failure notification can be used to freeze the TCP state and improve TCP goodput [5, 12, 16]. In the absence of network feedback, a scheme for improving TCP performance has been proposed in [23]. It uses out-of-order packet arrivals at the sender or the receiver as an indication of a route change, and temporarily freezes the TCP state. Dyer and Boppana, in [7], propose a TCP layer enhancement called fixed-RTO, where the RTO is doubled on a timeout, but after that, it is kept fixed until the route is re-established. Finally Sundaresan et. al propose a replacement for TCP to be used in ad hoc networks, in [21], that uses end-to-end rate control with selective acknowledgments.

Though MAC layer self-contention has been discussed by some researchers [6, 9, 21], only one of the approaches [9] addresses the problem by enhancing the MAC layer, whereas the other approaches reside above the MAC layer. Our proposed MAC layer mechanisms can be used along with other routing and transport layer solutions discussed above for further improving TCP goodput.

2.1 IEEE 802.11 MAC Protocol in Brief

This section briefly describes portions of the Distributed Coordination Function (DCF) mode of IEEE 802.11 pertinent to our study. For more details on IEEE 802.11, please refer to [2].

IEEE 802.11 is fundamentally a CSMA/CA protocol on shared wireless media. To initiate transmission, if the medium appears idle, a node sends a MAC layer control packet, called an RTS (request-to-send), to the receiver. If the receiver receives the RTS, and physical and virtual carrier sense that indicate the medium is idle, it responds with a CTS (clear-to-send). This is followed by data transmission by the sender, and a subsequent acknowledgment from the receiver. These transmissions are separated by short durations called SIFS (Short Inter-Frame Space).

If the packet transmission fails (i.e., no acknowledgment is received), the sender backs off before attempting retransmission. The number of retransmission attempts per packet is limited. If all retry attempts are exhausted, the packet is dropped and the link layer is informed.

While the 802.11 standard supports multiple data rates (the so-called extended rate set), it mandates that control packets, including RTS, CTS, and ACKs, be sent using the base rate. 802.11b, for example, allows data transmission at up to 11Mbps, but defines it's base rate to be 1Mbps[2].

3 Quick-Exchange and Fast-Forward

In this section we discus our *quick-exchange* and *fast-forward* enhancements to the MAC layer. Though the presentation is in reference to the IEEE 802.11 MAC protocol, the concept can be applied to other CSMA/CA MAC protocols. We present motivation for, and an overview of, the protocol modifications, followed by a discussion of the specific changes needed to the packet formats. We conclude this section by deliberating why we expect benefits from these two enhancements.

3.1 Quick-Exchange

Motivation & Overview:

Consider an adjacent pair of nodes in which the first node has a packet queued for the second node, and the second node has a packet queued for the first node. The first observation is that the IEEE 802.11 MAC protocol requires at least 6 control packets (RTS, CTS, and ACK for each packet) to enable transmission of these two data packets. The second observation is that prior to each packet transmission the DCF requires the sender to backoff for a random period for collision avoidance. When using a reliable transport protocol such as TCP, RTCP [20], XTP [1], or SCTP [8] which require a reverse flow (from the destination to the source) carrying an acknowledgment or feedback information it is reasonable to expect this situation to occur frequently over the lifetime of the transport connection.

The quick-exchange extension allows the two packets to be exchanged between adjacent nodes in a single dialog (RTS-CTS exchange) as shown in Figure 1. The RTS-CTS-DATA-ACK dialog is extended by an additional data packet transmission (DATA2) from the RTS-receiver with the piggybacked acknowledgment (ACK1) for the first data packet (DATA1). Finally, if DATA2 is received correctly, the RTS-sender sends a corresponding acknowledgment (ACK2).

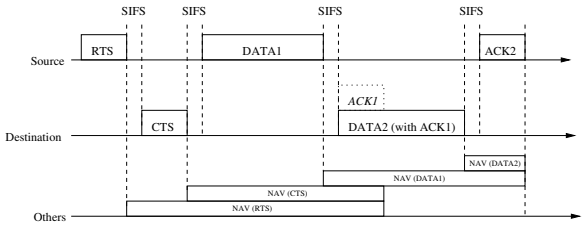

Figure 1. 802.11 with Quick-Exchange

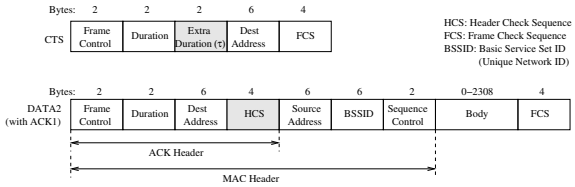

Figure 2. Modified packet formats required by quick-exchange enhancement (new fields in gray)

After receiving an RTS, the RTS-receiver searches its MAC layer packet buffer and interface queue for a suitable packet awaiting transmission to the RTS-sender[2]. If the RTS-receiver does not have such a packet, the standard 802.11 MAC behavior continues (RTS-CTS-DATA-ACK). If the RTS-receiver finds a suitable packet, the packet is moved to the head of the interface queue. The RTS-receiver then indicates the additional time, τ, needed for the quick-exchange in a separate field of the CTS packet.

When the RTS-sender receives the CTS, it identifies that an extra duration of τ is needed for the dialog. If the RTS-sender wishes to honor the quick-exchange request, the contents of the duration field in the DATA1 packet is modified to include the extra time τ and the channel is reserved for the transmission of DATA2. Note that in the case where the RTS-sender chooses not to honor the request for quick-exchange, the RTS-recipient can detect this by observing that the duration field in the transmitted DATA packet has not been extended by the requested amount.

On transmission of the DATA packet, the neighbors of the RTS-sender update their NAVs (Network Allocation Vectors) accordingly and defer transmissions until the end of the quick-exchange. Since the duration field in the piggybacked ACK1 packet also includes the time needed for quick-exchange, the neighbors of the RTS-receiver are also notified and defer transmissions to allow the quick-exchange to succeed.

The actions taken on transmission failure are similar to the standard DCF. If the RTS-sender does not receive a CTS, it backs off and again contends for the channel by retransmitting the RTS at a later time. If DATA1 or DATA2 are transmitted but are not acknowledged, the respective sender initiates a standard retransmission after a backoff. The normal retry rules with regards to packet abandonment are honored.

Though we started our discussion with a single TCP connection, we note that exchanged DATA packets need not be from the same transport connection. Furthermore, we note that as this approach *stretches* the duration of the channel reservation held by the two nodes. We advocate its use for packet exchanges where at least one of the two packets is a small packet (such as a TCP-ACK) to minimize potential unfairness[3]. In our study we impose a limit on the total number of DATA bytes allowed to be transmitted in a single RTS-CTS dialog with quick-exchange.

Packet Modifications:

Our modifications to the standard CTS and DATA packet formats are shown in gray in Figure 2. We now describe the modifications to the packet formats in detail.

- *CTS packet modifications:* The CTS packet now contains a new field that indicates the additional duration (τ) required for allowing the successful transmission of the DATA2 packet. This field is used by the RTS-sender to modify the content of its duration field in the subsequent transmission of the DATA1 packet. Thus, *the duration field of the DATA1 and ACK1 packets advertises the transmission of DATA2 to the neighbors*.

  This transmission of DATA2 can also be advertised by increasing the value in the duration field (by τ) in the CTS packet. However, this optimistic reservation may lead to an increased wastage of channel capacity if the RTS-sender is unable or unwilling to accept the quick-exchange request.

- *DATA packet modifications:* The modified DATA packet format is used only for sending DATA2, as DATA2 contains the piggybacked acknowledgment for DATA1. We define a new packet type (as designated by the frame control field of the MAC header) for DATA2 (see Figure 2).

  The ACK1 and DATA2 packets are sent in the same physical frame, thereby saving the overhead of one physical layer header and one MAC layer header. A separate checksum for the ACK header (HCS - header

[2]To speed up the search for a suitable packet, other queuing techniques such as per-neighbor queues may be used

[3]The potential unfairness is a consequence of the recipient of a data packet getting an advantage in channel access as compared with its neighbors.

check sequence) allows the RTS-sender to receive and process the ACK portion independent of the packet DATA payload.

Benefits of Quick-Exchange :

The expected benefits of our quick-exchange enhancement are as follows:

1. *Reduced inter-flow self contention:* By allowing flows sharing a logical link to cooperate in moving packets across the network, reliable transport connections may see increased usable channel capacity.

2. *Reduced control overhead:* The transmission of DATA2 does not require explicit RTS or CTS packets. Reducing control overhead can markedly increase usable channel capacity. The 802.11 specification requires that control packets be sent at the base rate, which is typically a much lower bit rate than is used for data packets - so while each control packet is small, they carry a high cost in terms of usable channel capacity.

3. *Fewer backoffs:* Transmission of the second data packet of the quick-exchange process is not preceded by a random backoff period. This reduces the average time that a node spends backing off before transmission and thereby, is expected to increase channel utilization.

3.2 Fast-Forward

Motivation and Overview:

Examining the interactions between the packet transmissions of nodes along the path of a single TCP connection, we observe that a significant portion (20%) of transmission attempts fail to acquire the channel on their first attempt. Adjacent nodes compete with each other for media access while attempting to service packets of the same flow.

This strongly suggests that senders are attempting to introduce packets into the network faster than the network is able to service those packets. In this case, the sender is contending with those packets still within the senders interference range. Fast-forward is intended to alleviate this bottleneck by permitting a node, upon receipt of a packet, to aggressively forward the packet toward its ultimate destination and out of the interference range of the sender, before the sender re-acquires the medium for subsequent transmission attempts.

The fast-forward modification introduces a new packet type (ACK-RTS) which serves both as a MAC-layer DATA

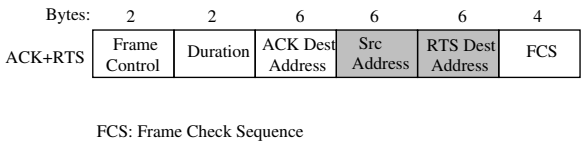

Figure 3. Packet formats for fast-forward enhancement (new fields in gray)

acknowledgment as well as a new RTS message, as shown in Figure 3. Upon receipt of a data packet header, the receiving node determines, via a request to the routing layer, the next hop for the incoming packet. If the next hop can be determined *without the need to generate route discovery or address resolution traffic* (i.e. there are hits in both the ARP and routing caches), the receiving node sends an ACK-RTS; the ACK-RTS simultaneously informs the sender of successful receipt and announces the intent to forward this packet.

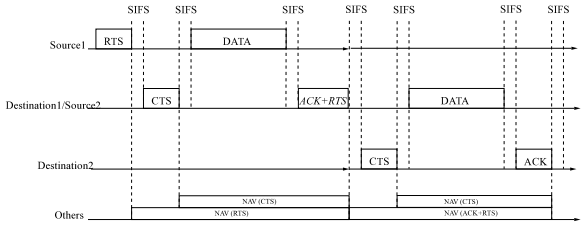

Figure 4. 802.11 with Fast-Forward

The node targeted by the ACK-RTS packet responds per the normal 802.11 RTS/CTS mechanism, and all nodes that receive the ACK-RTS update their NAVs to reflect the specified additional duration. The modified packet dialog is shown in Figure 4. In the event that no CTS is received in response to the ACK-RTS, or if the transmission fails, the data packet is delivered to the higher layers and normal DCF defined behavior is resumed.

By design fast forward can cause a connection to monopolize the channel, which presents a potential for unfairness. We attempt to limit the potential unfairness by restricting fast-forward in two ways; we focus on a probabilistic approach in which each node, upon receipt of a packet suitable for fast-forwarding, uses a Bernoulli process to determine if the packet should be fast-forwarded. So a node chooses to fast-forward a received packet with probability p (a system parameter).

Since the intuition that motivates fast-forward is to help a packet move out of the sending node's interference range, we have also studied limiting the number of consecutive fast-forward events a given packet can

undergo. One obvious choice for this limit would be $\lceil interference\ range\ /\ transmission\ range \rceil$ where $\lceil x \rceil$ denotes the ceiling of x. These results are omitted due to space constraints, but are similar to presented results.

As in quick exchange, where the intuition was motivated by a single TCP connection and then generalized to the multi-flow case, we observe that restricting fast-forward to packets intended for the same end-point as the incoming packet is not necessary. In our study we examine three possible policies; ***any-forward***, ***link-forward***, and ***flow-forward***. Any-forward is the most generic method and will cause a node to fast-forward the packet currently at the head of the interface queue - regardless of destination. Link-forward restricts the candidate packets to those waiting to be sent to the same next hop as the incoming packet, and flow-forward restricts candidates to those packets intended for the same transport layer end-point.

Packet Modifications:

Fast-forward requires a new packet type, an ACK-RTS, which can be thought of as an augmented ACK packet. The packet format is shown in Figure 3. Twelve bytes are added to an ACK packet - 6 bytes to represent the target of the RTS portion of the message, and 6 bytes to denote the sender of the packet.

Benefits of Fast-Forward:

The expected benefits of our Fast-Forward enhancement are as follows:

1. *Reduced intra-flow self contention:* By allowing intermediate nodes to fast forward data packets, the sender is prevented from injecting subsequent packets into the network until the previous packet has been forwarded beyond its interference range. This, in turn, is expected to reduce contention induced back-offs and false link failures in the network.

2. *Reduced control overhead:* When fast-forwarding a packet, the ACK-RTS packet acts as both an acknowledgement to previous-hop sender and an RTS request to the next-hop node. Thus, one RTS control packet is suppressed with successful fast-forward.

3. *Reduced backoff delay*: Since a packet is fast forwarded to the next hop node without requiring a random backoff period prior to transmission, the backoff delay at intermediate nodes will be reduced.

4 Simulation and Analysis

Our simulations are conducted using version 2.26 of the *ns-2* simulator [13]. Our proposed modifications are implemented as extensions to the existing 802.11 MAC implementation. In the course of our work we also made modifications and corrections to the core 802.11 code which have been submitted to the ns-2 maintainers for inclusion in future releases[4].

In all simulated scenarios, nodes have a common nominal transmission range of 250m and interference range of 550m. We focus our study to use of the Ad-hoc On-demand Distance Vector (AODV) routing protocol [19]. We have conducted experiments with dynamic source routing (DSR) [14] to confirm that our modifications are routing layer independent; results were similar and are omitted due to the space constraints. Our reliable transport protocol of choice is TCP-NewReno with delayed acknowledgment[5]. In all cases, data packets are 1000 bytes and all transport layer connections, once initiated, always have data to transmit.

In our study we examine static string as well as randomly generated topologies with varying levels of load and mobility.

While we examined several parameterizations of our proposed schemes we limit the results presented in the interest of brevity and clarity.

It is important to note that when using the combination of quick exchange and fast forward, quick exchange has "priority" over fast forward. This is because the decision to quick exchange is made by the receiver between reception of the RTS and transmission of the CTS, whereas the decision to fast-forward is made by the receiver in the time between receipt of a DATA frame and transmission of an ACK/ACK-RTS. Simply put, a node which has a packet suitable for quick exchange can either initiate quick exchange or decide to wait in the hope that the impending DATA packet will be suitable for fast-forward. As the fast-forward policy gets more restrictive (from any-forward, to link-forward, to flow-forward) the chances of the incoming packet being suitable decrease, making deference of quick-exchange in favor of potential fast-forward a less attractive strategy.

[4] Of particular significance to our work were the addition of accurate physical carrier sense and corrected EIFS back-off behavior. These patches to ns-2.26 are available from the authors upon request.

[5] A limited number of experiments were conducted with other TCP variants. The observed results were similar in flavor to those reported.

4.1 String Topology

We begin our study by examining the behavior of our proposed schemes in static string topologies. We examine strings that vary between 2 to 19 hops in length, with the inter-node spacing set to between 120 and 240 meters. A single TCP connection is established from one end of the string to the other. Reported results are averages over 20 simulation runs of 300 second duration.

Performance and Analysis:

Fast forward and quick exchange both outperform the standard MAC protocol in simple string topologies. This is easily explained since both mechanisms are able to achieve their objectives in the absence of competition between transport connections. The combination of the two enhancements typically produces improvements that are slightly better than either scheme in isolation.

Figure 5 plots the results as a percentage change versus the standard MAC protocol for various string lengths. We observe that while improvements due to quick exchange seem unrelated to the path length, gains from fast forward generally increase with path length (and can be as much as 45% for an 18 hop string). We also observe that in some cases, while either scheme by itself provides only small gains, significant improvements can be had by using them in combination. We attribute this to the opportunistic nature of the schemes; if a node cannot quick-exchange a given packet, it may be able to fast-forward it thereby improving channel utilization and efficiency.

Reducing the inter-node spacing does not substantively affect the results as only one node within 250 meters of the sender will forward the packet and all others are active only during route discovery.

The fast-forward results presented here use the probabilistic limit described in section 3.2. We explored values of p from 0.1 (infrequent fast-forward) to 1.0 (always fast-forward) and observed that improvement increased as the probability increased and plateaued at $p = 0.75$ Above $p = 0.75$ a packet placed on the string was fast-forwarded directly to the other end but the effects of inter-flow contention (between the TCP-DATA packets and TCP-ACK packets) increased and prevented further improvement.

4.2 Random Static Scenarios

We continue our investigation by deploying our proposed enhancements in randomly generated topologies. In static scenarios we randomly place 100 nodes in a 2.5 kilometer by 1 kilometer region. We vary the number of TCP flows

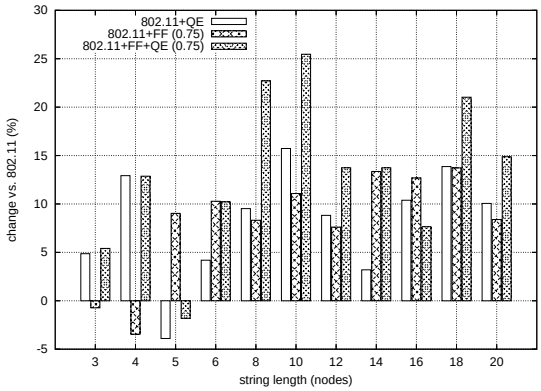

Figure 5. Percentage Goodput Change vs. String Length. 802.11. QE, Probabilistic FF (p=.75), and QE+FF (p=.75). 240m inter-node spacing.

in the network from 4 to 25 to study the performance of our changes with increased network load. Reported results are averages over 20 runs, each simulating 300 seconds of activity.

Performance and Analysis:

Figure 6 plots the change in aggregate goodput with QE and FF versus the standard 802.11 MAC protocol against the number of active connections in the network. While quick exchange produces consistent gains over the standard MAC protocol, fast forward under a variety of limits only occasionally outperforms the standard MAC protocol, and more often significantly under-performs as compared to standard 802.11. This was in contrary to our expectations of achieving goodput gains with our fast-forward scheme. We first discuss the behavior of quick-exchange in these scenarios and then return to the analysis of fast-forward.

Quick exchange is able to produce consistent benefits which generally increase with the connection count. As the number of connections increases, so does the likelihood that neighboring nodes will have packets suitable for quick-exchange destined for one-another. Since the only requirement is that the combined size of the packets be under a threshold (1400 bytes in our experiments), DATA packets from one TCP connection are frequently quick exchanged for ACK packets of an unrelated TCP connection. Similarly, multiple TCP-ACK packets for different transport flows are often exchanged, reducing the round-trip delay for both flows.

Subsequent investigation into the behavior of fast-forward revealed that two primary factors contributed to the counter-intuitive results. Fast-forward in many scenarios,

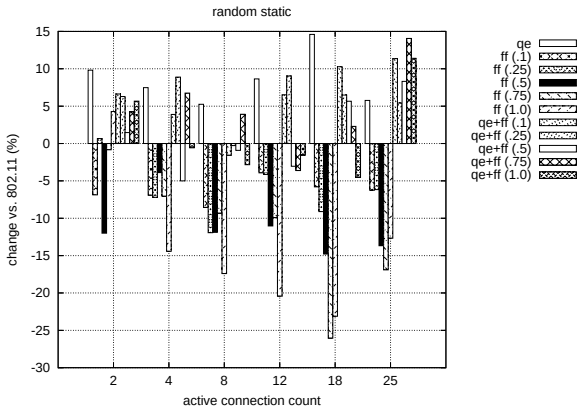

Figure 6. Percentage Change vs. 802.11. QE, Probabilistic FF (p), FF+QE. Random Static Topologies, Aggregate goodput

allowed a longer (in terms of hop count) connection to more successfully utilize the channel - producing gains of up to 300% for connections of 10 hops. Shorter flows (2-4 hops in length) in the neighborhood of this longer flow were unable to *dominate* the channel as completely to the large extent that they would otherwise, resulting in reductions in goodput in excess of 50%. When viewed in the absolute, the 300% gain experienced by the longer flow contribute much less to the aggregate goodput than the 50% loss seen by the shorter flow.

By examining a set of representative scenarios we observed that the second factor contributing to fast-forward's unpredictable behavior resulted from the interplay between fast forward and TCP's round trip time (*RTT*) estimate and back-off mechanism. To understand the problem, we consider the round-trip time observed by a TCP sender when a given DATA packet is successfully fast-forwarded by many of the intermediate nodes. We then compare this against the case when a packet is fast-forwarded by very few intermediate nodes (e.g. for a different packet transmission of the same connection). These two scenarios will produce significantly different round trip times, increasing the round trip time variance measured by the TCP sender. In the presence of multiple flows the effect is multiplied - short *RTT* observations for one flow directly correlate to long *RTT* observations for other flows in the same interference region. This increased variance, and corresponding increase in the smoothed *RTT* estimate, cause TCP to react more slowly to actual packet drops, caused in this case, by false link failures along the path. Figure 7 illustrates the TCP source *RTT* variance and smoothed *RTT* for a representative flow in a multi-flow random scenario. We observe that quick exchange, in contrast, tends to reduce *RTT* variance and produce a lower smoothed *RTT* estimate.

The MAC with both fast-forward and quick-exchange was occasionally able to outperform the standard MAC. In these cases, quick-exchange was offsetting some of the losses incurred by fast-forward - helping mitigate the adverse effects fast-forward has on TCP round-trip estimation.

To validate our intuition that fast-forward could produce benefits in a multi-hop wireless network were it not for the fluctuations in TCP *RTT*, we replaced the TCP flows in our random-static experiments with UDP constant bitrate traffic. By removing TCP's congestion control and back-off mechanism we confirmed that fast-forward can produce throughput improvements, even in heavily loaded (25 active connections) scenarios. **This suggests that our fast-forward modification does improve MAC efficiency at the cost of higher round-trip-time variance, and that TCP is unable to take advantage of the MAC-layer improvement.** Table 1 summarizes these findings.

Number of Connections	Improvement vs. 802.11 (%)
1	21.5
4	8.8
8	4.5
12	3.3
18	5.9
25	6.0

Table 1. UDP CBR Traffic, 802.11+FF, 1000 byte packets, 200 packets/second/source. Averaged over 50 runs.

Experiments with fast-forward in mobile scenarios exaggerated these issues (mobility and link failures caused further degradation). We omit these results from this paper since they do not lend any additional insight.

4.3 Random Mobile Scenarios

We examine the behavior of quick exchange in randomly generated topologies with varying levels of mobility. Mobile scenarios are generated using the random way-point mobility model with a pause time of 10 seconds. We modified the *ns-2* random way-point generation code to set $speed_{min}$ to 10% of $speed_{max}$, rather than the default of 0, to avoid some of the potential pitfalls of the random way-point model [26].

In all scenarios we randomly place 100 nodes in 1 kilometer by 1 kilometer region. We vary the number of TCP

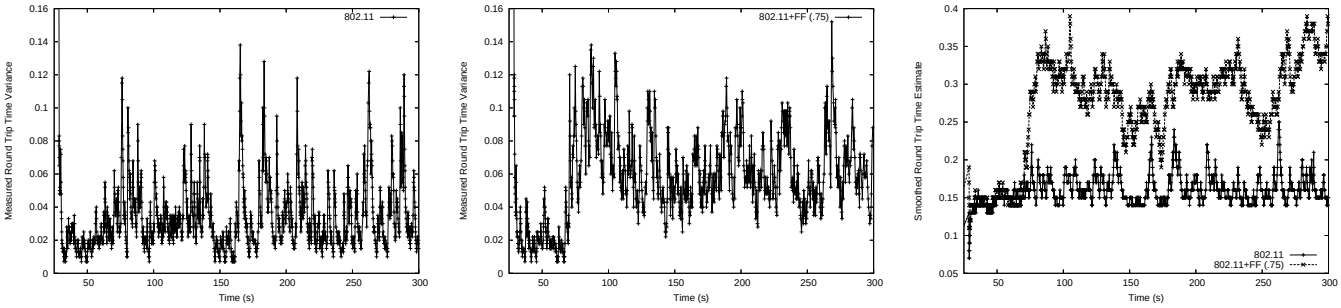

Figure 7. Round Trip Variance, Smoothed Round Trip Estimate vs. Time; 802.11, 802.11+FF

flows in the network from 4 to 25, and vary the maximum speed of motion from 1 to 20 meters per second – representing the range from pedestrian to vehicular speeds. Reported results are averages over 20 runs of 300 second duration.

Performance and Analysis:

In all studied scenarios, quick exchange delivers moderate but measurable improvements over the standard 802.11 MAC protocol. These results are summarized in Table 2. As suggested by our intuition, false link failures are reduced on average by 20% as compared to the standard MAC protocol. Further, we observe that even in the face of high mobility (20m/s), less than 1% of the initiated quick exchanges fail.

Number of Connections	Speed (m/s)			
	1	5	10	20
4	2.4	10.2	5.5	5.8
8	7.1	6.3	5.3	6.0
12	6.6	7.5	2.9	7.6
18	9.8	5.1	6.9	5.6
25	6.7	8.6	5.1	9.2

Table 2. Random Mobile, 802.11+QE Percentage Improvement vs. 802.11.

5 Discussion and Future Work

In this section we revisit some issues that are not completely addressed in the paper, and are part of future work.

- *MAC layer fairness:* Though our main goal is to increase the end-to-end goodput of transport connections, our approach may lead to local MAC layer unfairness. MAC layer fairness has been studied by some researchers [17, 22], but there is no clear consensus as to what fairness metric is useful for multi-hop flows.
- *TCP layer fairness:* Since flows with fewer hops consume less spectrum, giving higher channel share to shorter flows leads to improved aggregate goodput. For scenarios with multiple flows we have studied the aggregate as well as individual good-puts. However, equalizing the goodput of all the TCP connections may lead to lower utilization. Mechanisms to achieve TCP fairness in ad-hoc networks has been studied only in a few cases with simplifying assumptions [9, 25]. TCP layer fairness is a relatively unchartered territory that is not explored in our work.

Our study with fast-forward and UDP traffic suggest that fast-forward does improve MAC efficiency; however the benifits are offset by the unfairness that fast-forward causes between multiple TCP connections (i.e. fast-forward increases the *RTT* variation). The effects of the increased *RTT* variation may be alleviated by previously proposed techniques, such as freezing TCP state upon packet loss [5, 12, 16], fixed RTO [7], or rate based control [21]. The design of a reliable transport layer solution that exploits the reduced MAC layer contention provided by our proposed enhancements needs further investigation.

6 Conclusions

In this work, we study the fundamental problem of self-contention - contention between packets of the same flow unique to multi-hop wireless networks - which has received relatively little attention to date. We propose and study two MAC layer enhancements, called *quick-exchange* and *fast-forward*, that attempt to address the problem of self-contention in different ways. Quick exchange facilitates efficient exchange of packets between communicating neighbors, reducing inter-flow self contention. Fast-forward attempts to move packets out of a senders interference range

to reduce intra-flow self contention.

We evaluate the performance of our approaches using simulations in *ns-2* with our modifications to the IEEE 802.11 as the MAC protocol. Our quick-exchange enhancement provides improvements in TCP goodput by up to 20% in string topologies, 15% in random static scenarios and 10% in random mobile scenarios. Our fast-forward enhancement in conjunction with TCP fails to outperform the standard MAC protocol in randomly generated topologies. We attribute this degradation to the interaction between FF and TCP, and show that FF is improving MAC efficiency – as evidenced by improved throughput among multiple competing UDP flows – but that unmodified TCP is unable to benefit from the improvement. It is possible that TCP may be modified to better exploit the efficiency provided by fast-forward, and we intend to consider this in future work.

References

[1] Xpress Transfer Protocol. XTP Forum, Santa Barbara, CA, 1992.

[2] I. S. . 1999. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications , 1999.

[3] A. Acharya, A. Misra, and S. Bansal. A label-switching packet forwarding architecture for multi-hop wireless LANs. In *Proc. ACM International Workshop on Wireless Mobile Multimedia (WoWMoM)*, September 2002.

[4] V. Anantharaman and R. Sivakumar. A Microscopic Analysis of TCP Performance over Wireless Ad-hoc Networks. In *Proc. ACM SIGMETRICS (poster paper)*, June 2002.

[5] K. Chandran, S. V. S. Raghunathan, and R. Prakash. A Feedback Based Scheme for Improving TCP Performance in Ad-Hoc Wireless Networks. In *Proc. IEEE ICDCS*, pages 472–479, May 1998.

[6] C. Cordeiro, S. R. Das, and D. P. Agrawal. COPAS: Dynamic Contention-Balancing to Enhance the Performance of TCP over Multi-hop Wireless Networks. In *Proc. 10th Intl. Conf. on Computer Comm. and Networks (IC3N)*, pages 382–387, Oct. 2002.

[7] T. D. Dyer and R. V. Boppana. A Comparison of TCP Performance over Three Routing Protocols for Mobile Ad Hoc Networks. In *Proc. ACM MOBIHOC*, Oct. 2001.

[8] R. S. et. al. SCTP: Stream Control Transmission Protocol. RFC 2960, Network Working Group, 2000.

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

[10] C. Fullmer and J. J. Garcia-Luna-Aceves. Floor Acquisition multiple access (FAMA) for packet-radio networks. In *Proc. ACM SIGCOMM*, 1995.

[11] M. Gerla, , R. Bagrodia, L. Zhang, and L. Wang. TCP over Wireless Multi-hop Protocols. In *Proc. IEEE ICC*, June 1999.

[12] G. Holland and N. H. Vaidya. Analysis of TCP Performance over Mobile Ad-hoc Networks. In *Proc. ACM MOBICOM*, Aug. 1999.

[13] I. S. Institute. ns-2: Network Simultor. http://www.isi.edu/nsnam/ns/.

[14] D. B. Johnson and D. A. Maltz. *Dynamic Source Routing in Ad Hoc Wireless Networks*. Kluwer Academic Publishers, 1996.

[15] H. Lim, K. Xu, and M. Gerla. TCP Performance over Multipath Routing in Mobile Ad-hoc Networks. In *Proc. IEEE ICC*, 2003.

[16] J. Liu and S. Singh. ATCP: TCP for Mobile Ad-hoc Networks. *IEEE JSAC*, 19(7):1300–1315, July 2001.

[17] T. Nandagopal, T.-E. Kim, X. Gao, and V. Bharghavan. Achieving MAC Layer Fairness in Wireless Packet Networks. In *Proc. ACM MOBICOM*, Aug. 2000.

[18] C. Parsa and J. J. Garcia-Luna-Aceves. Improving TCP Performance over Wireless Networks at the Link Layer. *ACM Mobile Networks and Applications Journal*, 5(1):57–71, 2000.

[19] C. E. Perkins and E. M. Royer. Ad-hoc on-demand distance vector routing. In *Proc. 2nd IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–100, February 1999.

[20] H. Schulzrinne, S. Casner, R. Frederick, and S. McCane. RTP: A Transport Protocol for Real Time Applications. RFC 1889, Network Working Group, 1994.

[21] K. Sundaresan, V. Anantharaman, H.-Y. Hsieh, and R. Sivakumar. ATP: A Reliable Transport Protocol for Ad-hoc Networks. In *Proc. ACM MOBIHOC*, June 2003.

[22] N. H. Vaidya, P. Bahl, and S. Gupta. Distributed Fair Scheduling in a Wireless LAN. In *Proc. ACM MOBICOM*, Aug. 2000.

[23] F. Wang and Y. Zhang. Improving TCP Performance over Mobile Ad-hoc Networks with Out-of-order Detection and Response. In *Proc. ACM MOBIHOC*, 2002.

[24] H. Wu, Y. Peng, K. Long, S. Cheng, and J. Ma. Performance of Reliable Transport Protocol over IEEE 802.11 Wireless LAN: Analysis And Enhancement. In *Proc. of IEEE INFOCOM*, 2002.

[25] K. Xu, M. Gerla, L. Qi, and Y. Shu. Enhancing tcp fairness in ad hoc wireless networks using neighborhood red. In *Proc. ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, September 2003.

[26] J. Yoon, M. Liu, and B. Noble. Random Waypoint Considered Harmful. In *Proc. of IEEE INFOCOM*, 2003.