A Bidirectional Multi-channel MAC Protocol for Improving TCP Performance on Multihop Wireless Ad Hoc Networks

Tianbo Kuang Carey Williamson
Department of Computer Science
University of Calgary
Calgary, AB, Canada T2N 1N4
email: {kuang, carey}@cpsc.ucalgary.ca

Abstract

The TCP protocol often suffers from performance problems in conventional single-channel multihop wireless ad hoc networks. The problems arise from hidden node and exposed node problems, which can lead to channel contention in the forward direction between TCP DATA packets that are part of the same TCP flow control window, as well as contention between TCP DATA and TCP ACK packets flowing in opposite directions. In this paper, we propose and evaluate a novel bidirectional multi-channel MAC protocol designed to improve TCP performance over a multihop wireless network. The protocol uses multiple transmission channels at the physical layer to reduce TCP DATA-DATA contention, and bidirectional RTS/CTS channel reservations to reduce TCP DATA-ACK collisions. Simulation results for static multihop networks show TCP throughput gains of 50% to 185%, compared to a conventional IEEE 802.11 MAC protocol. Fairness is also improved with our protocol, since contention is confined to a short handshake period on the control channel.

Keywords: Simulation, ns-2, TCP Performance, Multihop Wireless Ad Hoc Networks

1 Introduction

Multihop wireless ad hoc networks offer communications capability to mobile hosts without requiring a fixed infrastructure. In a multihop wireless network, the limited radio transmission range of each node often requires packets to traverse multiple intermediate wireless nodes en route from the source to the destination. That is, in addition to roles as a traffic source and a traffic sink, each node acts as a store-and-forward router for packet traffic originated from other sources. The easy-to-deploy nature of multihop wireless ad hoc networks leads to their growing use in sensor network, disaster recovery, and military communications applications.

An example of a multihop wireless network is shown in Figure 1. This example shows nodes (labeled A, B, C, D, X) laid out in a simple linear topology referred to as a “chain” network.

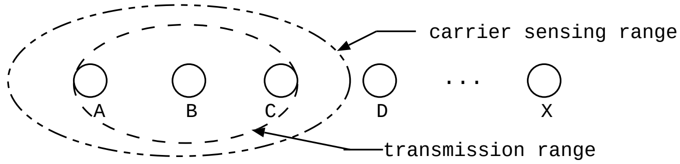


Figure 1: A Multihop Wireless Network with a Chain Topology

The topology is called multihop because (for example) a packet destined from A to D must use B and C as intermediate routers. Each of these intermediate forwarding steps is called a hop.

Figure 1 also shows two important physical properties of a multihop wireless ad hoc network. The circle labeled *transmission range* is the range over which other nodes can receive frames from a given node (B) without excessive errors. This range can be controlled by the transmission power. A wireless node also has a *carrier sensing range* that is at least as large as the transmission range. This range determines which other nodes a given node (e.g., node B) can "hear" when competing for access to the shared wireless channel.

A well-known problem related to the limited carrier sensing range is the *hidden node* problem. Nodes beyond the carrier sensing range of each other may try to send a frame to the same destination at the same time, causing collisions at the receiver. For example, simultaneous transmissions from B to C and from D to C in Figure 1 will collide at C, because D is a hidden node for B (and vice versa). Another problem is the *exposed node* problem, where a node near an active sender is ineligible to send or receive. For example, if C is sending to D, then the exposed node B cannot simultaneously receive data that A wants to send to B.

The overall performance achieved within a multihop wireless network depends on the Medium Access Control (MAC) protocol and transport protocol used. If the nodes are mobile, performance also depends on the mobility patterns and the ad hoc routing protocols used.

A popular MAC protocol for multihop wireless networks is the IEEE 802.11 MAC [3]. This protocol, which requires each node to sense the channel before sending a frame, is called CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance). To address the hidden node problem, the 802.11 MAC uses a Request-To-Send (RTS) and Clear-To-Send (CTS) handshake. A node sends an RTS control frame containing the MAC address of the intended receiver to indicate that it has a frame to send. Upon receiving the RTS, the intended receiver returns a CTS control frame if it is okay to receive the data frame. With this procedure, the other nodes within the carrier sensing range of the sender or the receiver must refrain from transmitting or receiving until the frame is finished. Nodes that are three hops away are unaffected, if the distance between nodes (and the carrier sensing range of each node) is the same as the transmission range. We call this distance (3 hops) the *contention distance*.

The determination of a transport-layer protocol suitable for future multihop wireless networks is not yet finalized. Since the Transmission Control Protocol (TCP) is used by many applications in the current Internet, it seems natural to consider this protocol for deployment and interoperability.

Unfortunately, previous studies show that TCP performance over multihop wireless networks is often disappointing. Even in a *static* multihop wireless environment, ignoring wireless channel errors and mobility-related link breakages, TCP shows performance problems [9, 18, 21]. The first problem is that TCP throughput often decreases dramatically with the number of hops traversed by a flow, regardless of the MAC protocol used [9, 18]. This trend is particularly evident when the TCP flow control window size is large, and many packets are "in flight" at a time. The reason is link-layer packet delay/drops caused by contention between data packets traveling in the same direction (i.e., the exposed node problem), and collisions between data packets and TCP acknowledgment (ACK) packets traveling in opposite directions (i.e., the hidden node problem). The second performance problem is unfairness that can occur between different TCP flows [18, 21]. The unfairness is due to unfortunate interactions between TCP and MAC-layer timers, causing a "rich get richer" and "poor get poorer" phenomenon (See Section 6.2 for a more detailed explanation for this).

Several methods have been proposed to remedy these problems. For example, Fu *et al.* [9] proposed a link-layer version of the Random Early Detection (RED) [7] algorithm to signal the TCP sender about impending congestion, and an *adaptive pacing* algorithm to distribute TCP data packets evenly across a multihop chain. Combined, these algorithms provide a throughput improvement of 5%-30%. As another example, Cordeiro *et al.* [6] proposed disjoint routes for forward TCP packets and backward TCP ACKs so that contention is reduced. They reported an average throughput improvement of 90% for 50 nodes randomly placed in a 1000m x 1000m rectangular area. However, the throughput improvement is smaller when the node density increases. We estimate the maximum improvement for 100 nodes in the same area as 47%.

The foregoing methods all assume a *single* transmission channel in the physical layer. Nevertheless, *multiple* channels can be used. For example, in the widely-deployed IEEE 802.11b standard, three non-overlapping channels can be used simultaneously. Some engineers advocate four slightly-overlapping simultaneous channels [13]. In Code Division Multiple Access (CDMA) [19] systems, more channels can be obtained since each chip code can be viewed as a different channel.

Several *multi-channel MAC* (MCMAC) protocols have been proposed to improve the overall ad hoc network capacity [10, 12, 17, 20, 23]. However, these studies typically measure the overall network capacity by employing a metric called aggregate *one-hop* throughput, or the total successfully delivered data per unit of time in the whole network, including those delivered by the intermediate nodes[1]. Few published studies have focused on TCP performance over a multi-channel MAC, and none of the proposed multi-channel MAC protocols specifically address the TCP traffic dynamics. While the overall capacity of an ad hoc network is interesting in its own right, we believe that the one-hop throughput does not reflect the DATA-DATA and DATA-ACK contention behavior *within* a single flow, which affects the throughput of that flow. Moreover, the one-hop throughput does not reflect the interactions between the TCP layer and the MAC mechanism.

In this paper, we propose a novel bidirectional multi-channel MAC (Bi-MCMAC) protocol to improve TCP performance in multihop wireless networks. The scheme is designed with a special emphasis on bidirectional TCP traffic issues. It reduces the link-layer contention using two key ideas:

- *The protocol uses multiple transmission channels at the physical layer.* One of the K channels ($K > 1$) is a control channel, while the other $K - 1$ are data channels. By

[1]More specifically, the one-hop throughput is usually measured as follows: in a grid or random network, a number of nodes are randomly chosen to be the traffic sources. Then a number of destinations, each being one hop away from its source, are randomly chosen. The successfully delivered data (usually unidirectional UDP-like data) is summed up and divided by the total time to form the aggregate one-hop throughput.

using multiple channels, the *contention distance* of the protocol is decreased, resulting in improved TCP throughput in multihop networks.

- *The protocol extends the RTS/CTS handshake to do bidirectional channel reservations.* This optimization is particularly apropos in the common case where TCP packets (e.g., data and ACKs) are flowing in opposite directions. By scheduling a bidirectional data transfer with a single RTS/CTS handshake, the contention between TCP data and ACK packets is reduced.

We evaluate the throughput, fairness, and delay performance of the proposed MAC protocol in a *static* multihop environment using the `ns-2` network simulator [22]. Simulation results show that the new MAC protocols improve TCP performance significantly for all three metrics. Furthermore, the bidirectional multi-channel MAC protocol requires only one transceiver/antenna. Therefore it can be implemented on current network hardware platforms.

The rest of the paper is organized as follows. Section 2 provides a brief description of related work. Section 3 introduces the proposed MAC protocol. Section 4 describes the performance evaluation methodology for the experiments. The specific experimental setups and results for throughput, fairness, and delay performance evaluation are described in Section 5, Section 6, and Section 7, respectively. Section 8 discusses implementation-related issues and possible extensions for future work. Finally, Section 8 concludes the paper.

2 Related Work

Fu *et al.* [9] analyzed the TCP throughput over a static multihop wireless network. Their analysis showed that in an h-hop chain network topology, the optimal TCP window size is $h/4$. A smaller value leads to bandwidth underutilization, while a larger value leads to excessive link-layer contention. These authors also proposed a link-level version of RED, and an adaptive pacing algorithm to improve TCP throughput. Cordeiro *et al.* [6] proposed a routing protocol to separate forward TCP packets from backward TCP ACKs. The intent is to reduce the DATA-ACK contention in a static multihop environment. Our work is similar to theirs in that we also study the TCP performance in a static multihop wireless network. The difference is that we focus on a multi-channel MAC protocol, while they used a single-channel IEEE 802.11 MAC.

Other techniques to improve the capacity of a single-channel multihop wireless network include MAC with parallel transmission by Acharya [2], and power-controlled MAC protocols by Monks *et al.* [15]. These protocols attempt to increase the number of concurrent frame transmissions by exploiting spatial channel reuse.

Several previous papers study multi-channel MAC protocols [10, 12, 17, 20, 23]. These all focus on the overall capacity of the network; none concentrate specifically on improving TCP performance at the flow level. The algorithm in [20] is a receiver-initiated collision avoidance scheme for frequency-hopping networks. In contrast, our algorithm is sender-initiated, and is intended for direct-sequence CDMA systems. The proposed algorithms in [10, 12] require multiple transceivers/antennas, while our algorithm requires only one transceiver/antenna pair. The works in [17, 23] are closest in spirit to our own in that they require only one transceiver/antenna. However, they do not consider bidirectional traffic issues in conjunction with the RTS/CTS handshake.

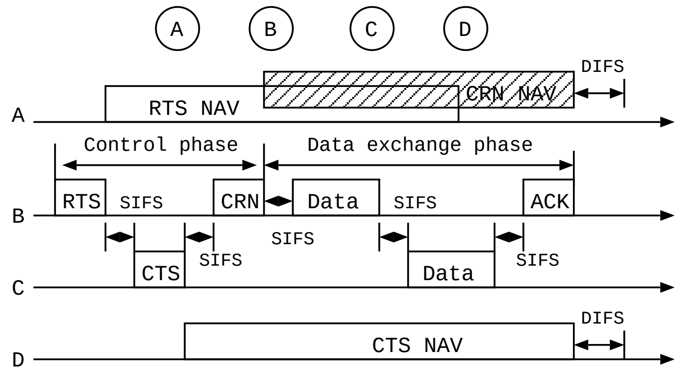


Figure 2: Proposed Bidirectional Multi-channel MAC Protocol

3 Proposed Bi-MCMAC Protocol

This section describes the proposed bidirectional multi-channel MAC protocol.

3.1 Protocol Overview

In general, two things have to be done in a multi-channel MAC protocol before a frame transmission can occur. First, a channel *negotiation* procedure must determine which (data) channel to use for a transmission between two nodes. Second, a channel *reservation* procedure must notify other nodes how long the chosen channel is reserved for this transmission episode.

The above procedures differ from the actual data frame transmission in that they exchange *control* information. A single *control channel* can be allocated to carry this information. The proposed Bi-MCMAC protocol adopts this approach, with a single control channel and several data channels. For each data transfer, it divides the process into two phases: the *control phase*, and the *data exchange phase*. In the control phase, control frames are exchanged between the communicating nodes on the control channel to negotiate the data channel to be used, as well as to indicate the channel reservation time (which can include the transmission time for one or more frames from each direction). Other nodes hearing the control frames employ *virtual carrier sensing* to keep the data channel reservation information. That is, they mark the data channel as reserved and set a timer for the indicated channel reservation time. To prevent a pair of nodes from occupying the channel indefinitely, a single handshake only allows one data packet exchange. In order to ensure fairness, each node must compete to access the control channel.

After the control phase, both the sender and receiver switch to the chosen data channel, and the data exchange phase begins. In the data exchange phase, the sender sends a frame to the receiver. If the frame is correctly received, the receiver prepares to send a MAC-layer acknowledgment back to sender. This acknowledgment is piggybacked in a data frame if the receiver has outbound data (e.g., a TCP ACK packet) for the sender, or is sent alone otherwise. If the original sender receives only a MAC-layer ACK, the data exchange episode ends. Otherwise, if it receives a data frame, it returns a MAC-layer ACK to the receiver, and the data exchange episode ends after that. Both nodes switch back to the control channel after the data exchange phase, to start another round if desired. The two phases can be seen in Figure 2.

Our proposed Bi-MCMAC protocol is an extension to the IEEE 802.11 MAC protocol. The

Bi-MCMAC uses the same carrier sensing and backoff procedure as that of the 802.11 MAC to compete for the control channel. When it obtains the control channel successfully, the Bi-MCMAC protocol extends the RTS/CTS exchange in the IEEE 802.11 MAC protocol to support the channel negotiation and reservation. The algorithms are described in the following two subsections.

3.2 The Channel Negotiation Procedure

Both the sender and receiver must maintain a list of free (data) channels. When the sender wishes to send a frame, it sends an RTS frame containing its own free channel list. The RTS sending procedure is the same as that of the IEEE 802.11 MAC (i.e., it has to compete for use of the shared control channel). When the receiver receives the RTS frame, it checks its own local free channels and compares them against the proposed list in the RTS frame. If there is at least one channel that is in both lists, it selects one such channel and sends this channel number back in a CTS frame. Otherwise, the receiver simply does not reply with a CTS. When the sender receives the CTS, it extracts the channel number to use. If no CTS arrives, it will timeout and retransmit the RTS (up to some maximum retry limit).

The algorithm used to select a channel is a separate topic that is beyond the scope of this current paper. In this paper, we use a simple "hysteresis" heuristic, wherein the receiver always selects the channel used for the last successful transmission, unless that channel is unavailable. In that case, it selects the lowest-numbered free channel. Studying channel selection algorithms is part of our planned future work.

3.3 The Channel Reservation Procedure

The channel reservation procedure needs to announce two pieces of information: which channel is reserved, and for how long. It is important that this information reaches nodes close to the sender *and* nodes close to the receiver. The nodes close to the receiver obtain this information from the CTS. The nodes near the sender (but out of the range of the receiver) *cannot* get this information because they see only the RTS, not the CTS, and there is no reserved channel information in the RTS frame. (Nor do these nodes know the time duration of the reservation, which was possibly modified by the receiver.) For this reason, we add a new control frame called the *Channel Reservation Notification* (CRN) to the RTS/CTS handshake. The CRN is sent after the sender receives the CTS, containing the channel and reservation duration information.

The channel reservation can be easily implemented using the RTS/CTS/CRN procedure. More specifically, when the sender sends an RTS, it sets a duration value in the RTS frame based on the transmission time required for its data frame. When the receiver receives the RTS, if it has data to send back, it increases the duration value in the CTS to include the time to transmit its own frame. After receiving the CTS, the sender announces the reservation by copying the duration value in the CTS to the corresponding field in the outgoing CRN frame.

Other nodes that overhear the CTS frame set a Network Allocation Vector (NAV) timer on the reserved data channel for a duration announced by the CTS frame. For those nodes that are not the destination of the RTS frame but are able to hear the RTS, they need to set a timer on the control channel based on the value announced by the RTS. This is to ensure that they do not compete for the control channel when the RTS sender and its intended receiver is using the control channel. This constraint is relaxed, however, when the nodes overhear the CRN frame, at which time the nodes cancel the control channel timer and set a timer on the reserved data channel. This means that the nodes are free to compete for the control channel after the control phase, as long as they do not want to transmit data to a node that has an on-going data transmission (this can be determined from the data channel timer.). In contrast, in the conventional 802.11 MAC protocol, nodes overhearing the RTS/CTS frames have to refrain from accessing the channel for the *whole* duration of the data transmission procedure.

With this design, it is also safe when a node overhears the RTS frame, but not the CRN frame. Such a node will refrain from contending for the control channel during the whole data transmission period.

Other than the above difference, the Bi-MCMAC protocol behaves the same as the 802.11 MAC protocol in that it employs collision avoidance, exponential backoff, and MAC layer retransmission. Our bidirectional multi-channel MAC protocol is summarized succinctly in Figure 2. This example shows the exchange of frames between node B and node C in a four-node multihop network. Note how node A is notified of the channel reservation time using the CRN frame.

4 Evaluation Methodology

We have conducted extensive simulations using the `ns-2` network simulator [22] to evaluate the TCP performance over the proposed Bi-MCMAC protocol. To fulfill this purpose, we have modified the ns-2 source code to model the Bi-MCMAC protocol. In addition, we also implemented a pure multi-channel MAC (MCMAC) protocol to compare the performance, which is essentially the proposed Bi-MCMAC without the bidirectional data exchange optimization.

Our performance evaluation focused on three aspects: throughput, fairness, and delay. In order to to this, we set up different ad hoc networks and TCP sessions in the simulator. In this section, we describe the setups that are common to all experiments. The specific configurations for different performance metrics, i.e. throughput, fairness, and delay, are deferred to the corresponding sections.

Table 1 summarizes the parameter settings used in the ns-2 network simulations. The transmission range and carrier sensing range were each set to 250 meters. The number of channels was $K = 4$, with channel 1 being the control channel and the other 3 channels serving as data channels. For all experiments, we consider only static multihop wireless networks. Therefore, specific ad hoc routing protocols do not have a big impact on the results.

The TCP protocol configuration is also indicated in Table 1. TCP NewReno [8] was used in the study since it is capable of handling multiple packet losses in a window of data (not an unlikely event in a multihop wireless network). Since TCP ACKs create contention between forward and backward traffic, TCP's Delayed-ACK option may have an impact on the performance. We did a preliminary simulation test with the IEEE 802.11 MAC protocol to test this hypothesis. The results indicate that enabling Delayed-ACKs improves throughput by about 10%. Detailed analysis also shows a decrease in the number of MAC-layer collisions. For this reason, all remaining experiments use the TCP Delayed-ACK option.

For all experiments, the errors were assumed to be caused by collisions. That is, no wireless channel errors were considered. The simulation time for each experiment was 300 seconds. All simulation experiments used 50 replications, each with a different seed for the random number

Table 1: Parameter Settings for ns-2 Network Simulations

Parameter	Value
SIFS	10 μs
DIFS	50 μs
Speed	1 Mbps
Number of Channels	4
Center Frequencies	2.412 MHz, 2.427 MHz, 2.447 MHz, 2.462 MHz
Channel Model	Two-ray ground
Transmission Range	250 meters
Carrier Sensing Range	250 meters
Ad Hoc Routing Protocol	AODV
Transport Protocol	TCP NewReno
TCP Delayed ACK	Enabled

generator (either for the MAC layer random backoff or the random topology generator). The results reported in the graphs and tables represent the mean, standard deviation, and 99% confidence interval calculated from the 50 runs.

5 Throughput Performance

5.1 Throughput Experiments

For the throughput experiments, three topologies were used, namely the chain topology, the grid topology, and the random topology. The chain topology was introduced in Figure 1. In our simulation, the distance between nodes is 250 meters. The number of nodes was varied from 2 to 18, in steps of 2. A single TCP connection was set up from the first node to the last node, with an infinite amount of data transferred from the application layer.

In a chain topology, a node has at most two neighboring nodes. Therefore, the local contention is low. To evaluate the effects of increased local contention, we simulated a grid topology, where a node could have up to four local neighbors (competitors), and several random topologies, where a node could have more than four local neighbors (competitors). The grid topology is shown in Figure 3, where 100 nodes were placed in rows and columns that are 250 meters apart. Two kinds of random topologies were tested. In the first random topology, 100 nodes were randomly (uniform) placed within a flat area of 500m x 500m. The second random topology is the same as the first one except that the flat area is reduced to 250m x 250m. Note that since the wireless transmission range is 250 meters, in the 250m x 250m topology, most nodes are within the transmission range of each other. In other words, they almost form a single cell. The three test scenarios can be viewed as increasing node density, with the grid topology having the lowest density, the 250m x 250m random topology having the highest density, and the 500m x 500m random topology in between.

For both grid and random topologies, a number of TCP connections, varying from 2 to 12 in steps of 2, were established between randomly chosen source and destination nodes. These TCP

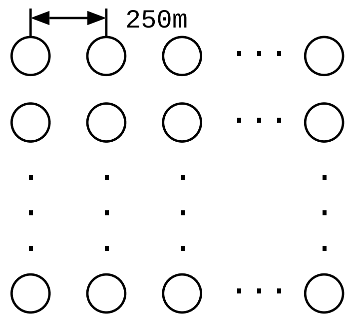


Figure 3: The Grid Topology

connections were randomly started within 0 and 1000 ms to avoid periodic congestion effects. In all experiments, the application was assumed to have an infinite amount of data to send.

Each experiment was simulated for 300 seconds. When the simulation finishes, the TCP "goodput" (the total number of bytes reaching the destination divided by the total simulation time) was calculated. The experiment was then repeated for 50 times. For the chain topology, the seed for the random number generator of the MAC layer backoff timer was changed for each repetition. For the grid topology, the seed for the random source/destination generator was different for each repetition. For the random topology, the seeds for both the random topology generator and the random source/destination generator were different for each repetition. Accordingly, for the grid topology, a total of 300 scenarios were tested; for the random topology experiments, for each number of TCP flow, 50 topologies were tested, yielding a total of $6 \times 50 = 300$ topologies. Note that the seeds of the random number generators for the MAC backoff timers were not changed for the grid and random topologies. In other words, both 802.11 MAC and Bi-MCMAC use the same MAC backoff sequence. This ensures that the ad hoc routing protocol finds the same route for the two protocols so that a fair comparison can be made.

The following subsections report the results for the chain topology, the grid topology, and the random topology, respectively.

5.2 Chain Topology

5.2.1 Ideal Throughput of a Chain

According to Li *et al.* [14], the maximum throughput of a single-channel chain with sufficient length is $\frac{C}{3}$, where C is the transmission rate of the wireless nodes in bits per second (bps). When the chain is long enough, nodes that are 3 hops apart can be transmitting at the same time without disturbing each other, if there is an ideal scheduler.

Following the analysis of Li *et al.*, the ideal throughput for a chain employing multichannel is $\frac{C}{2}$, as the contention distance is reduced from 3 to 2. Comparing these two throughput limits shows a definite advantage for the multi-channel protocol. Note that both bounds are independent of the number of hops traversed.

It is clear that this analysis ignores all adverse effects such as RTS/CTS/ACK overhead, collisions, and errors. Therefore it is only a loose upper bound for the achievable throughput. In Appendix 8 we estimate the efficiency of the three MAC protocols for TCP traffic when two nodes are involved by considering the control handshake overhead. It is found that the efficiency of the three protocols for transporting TCP traffic between two nodes is estimated to be 77.8% (802.11), 73.2% (MCMAC), and 78.4% (Bi-MCMAC) in steady state, if the packet size is 1024 bytes. Ignoring the contention issues, in a 1 Mbps 802.11 MAC, the ideal TCP throughput of a chain is therefore $(1/3) * 77.8\% = 259$ kbps. This value is $(1/2) * 78.4\% = 392$ kbps for the Bi-MCMAC protocol. In the next section, we show that this estimate is fairly accurate. We also see how the MAC protocols perform compared to these ideal results.

5.2.2 Simulation Results

Figure 4 (a) shows the average throughput for TCP NewReno as a function of the network size. In general, the throughput is good on the 2-node topology (as expected), but drops sharply as the next few nodes are added to the chain topology. Beyond 8 nodes, the dropoff in throughput is gradual. Note that the 99% confidence intervals are too small to discern in the graph.

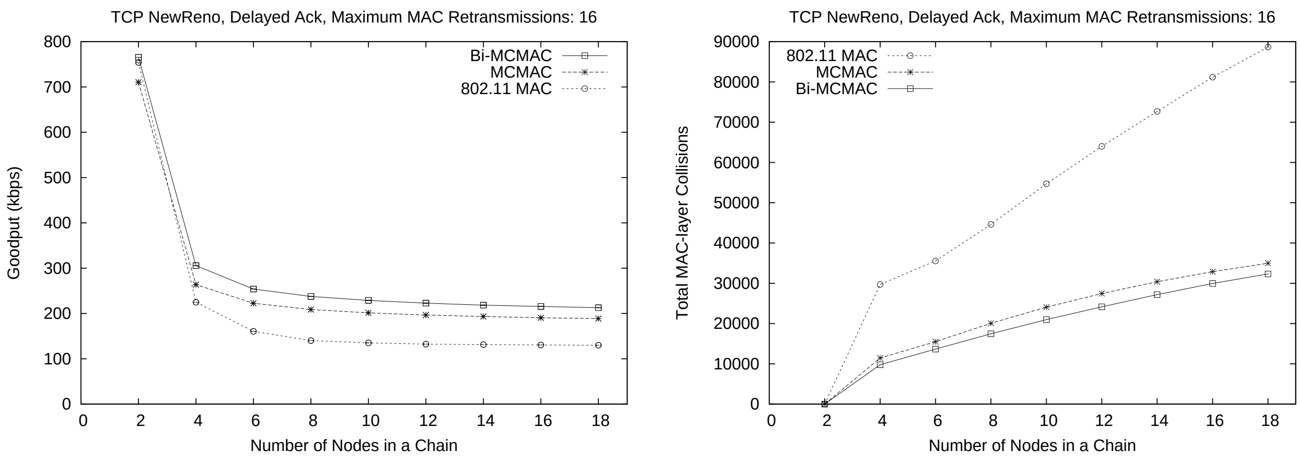
For a 2-node network, there are few differences among the three protocols, since the throughput results are dominated by the MAC overhead. The simulation results confirm the foregoing analysis. That is, in a two node network, the Bi-MCMAC provides the best performance, followed by the 802.11 MAC, followed by the MCMAC. In fact, the average TCP throughputs for the three protocols are 765.4 kbps, 754.0 kbps, and 710.5 kbps respectively, quite consistent with the estimated results. The slight discrepancy is because of contention that was not included in the estimation.

When the number of nodes in a chain increases, the DATA-DATA contention arises. This effect, coupled with DATA-ACK contention, causes the throughput degradation. Among the three MAC protocols evaluated, the Bi-MCMAC protocol mitigates both effects, therefore providing the best throughput. The performance of the pure MCMAC protocol follows that of the Bi-MCMAC protocol as it mitigates the DATA-DATA contention but not DATA-ACK contention. In contrast, IEEE 802.11 MAC protocol has the worst performance as it suffers from both effects. The simulation results show that, compared to the IEEE 802.11 MAC protocol, the average throughput advantage of the Bi-MCMAC protocol on large chain networks (more than 6 nodes) is about 67.1%. For a 4-node network, the average improvement is 35.9%, and for 6 nodes the advantage is 58.2%. These improvements are lower because the length of the chain is insufficient to exploit concurrent transmissions fully. The pure MCMAC protocol provides about 47.5% throughput advantage over IEEE 802.11 MAC for large chain networks. This result indicates that the bidirectional reservation optimization is an important feature of the Bi-MCMAC protocol.

Figure 4 (b) provides further results to explain the performance advantages of the Bi-MCMAC protocol. This graph shows the total number of MAC-layer collisions for the three MAC protocols, as a function of network topology size. Both multi-channel protocols have a much lower collision count, about one-quarter of that for the IEEE 802.11 MAC protocol, despite the fact that they sent *more* TCP packets within the simulated time.

In all the experiments, the multi-channel protocols rarely drop any packets due to excessive MAC-layer collisions and retransmissions. In contrast, the average contention drops of the IEEE 802 MAC protocol vary from 0 to 18 packets with different network sizes.

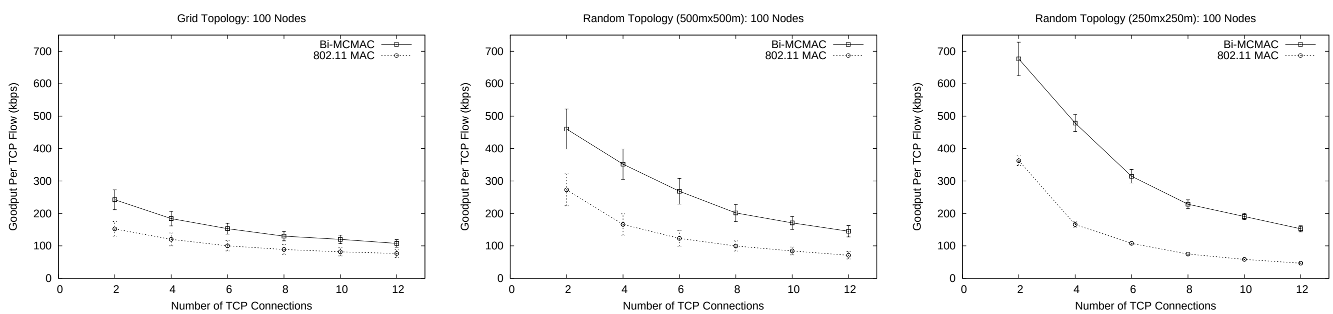
Finally we compare the simulation results to the analytical results mentioned previously. The average throughput of the Bi-MCMAC protocol for a 12-node chain is 222.9 kbps, which is 56.9% of the upper bound (392 kbps). The average throughput of the IEEE 802.11 MAC protocol in



(a) Throughput

(b) Collisions

Figure 4: Throughput Simulation Results in a Chain Topology Multihop Network



(a) Grid Topology

(b) Random Topology (500m x 500m)

(c) Random Topology (250m x 250m)

Figure 5: Per-flow Throughput for the Grid and Random Topology

the same network is only 132.4 kbps, which is 51.0% of its upper bound (259 kbps). This means that neither protocol achieves the ideal throughput. Further improvement is still possible.

5.3 Grid Topology

With the grid topology, we only compare the performance of the 802.11 MAC protocol and that of the Bi-MCMAC protocol. In order to do a fair comparison, in each experiment the seeds of the random number generators are set to be the same for both protocols so that the sources and destinations are the same. In other words, for both protocols, the TCP connections traverse the same nodes and the same number of hops.

Figure 5 (a) shows the per-flow throughput versus the number of TCP connections. It can be seen that the per-flow TCP throughput for Bi-MCMAC is still better than that of the 802.11 MAC, in spite of the increased local contention. The average improvement is 50%. The graph also shows that the 99% confidence intervals decrease as the number of flows increase. This is likely caused by the spreading of traffic among the nodes. More specifically, when the number of flows increases, more and more nodes in the network are likely to have traffic to send, causing more uniformly distributed contention. Accordingly, the variation is smaller. In contrast, when the number of flows is low, the traffic is less uniformly distributed among the nodes, causing uneven local contention. The result is more variation.

5.4 Random Topology

Similar to the grid topology, for each experiment in the random network, the same topology and the same set of source and destination nodes were used for both MAC protocols to ensure a fair comparison.

Figure 5 (b) and (c) show the simulation results. The Bi-MCMAC protocol again shows better performance, with an average improvement of 100% for the the 500m x 500m topology and 185% for the high density 250m x 250m topology.

Comparing the results, it can be seen that when the node density increases, the Bi-MCMAC protocol has greater advantages. This is because local contention increases. The Bi-MCMAC protocol resolves this contention by allowing more concurrent transmissions, therefore increasing the gain.

The throughput gains also show that the *reservation missing* problem (explained below) of the proposed protocol is not a severe disadvantage. Reservation missing occurs when a node misses the control handshake initiated by nearby nodes because its receiver is on the *data channel* (i.e., it is transmitting or receiving a data frame). Then when the same node has new data to send after finishing a data transfer cycle, it is possible that: (1) the node initiates a control handshake to a node in the data exchange phase; or (2) the node selects the same data channel as some nearby on-going transfers. The former case causes the initiating node to backoff as it does not receive any response from the receiving node. The latter case is more severe as it causes a collision on the data channel. The reservation missing problem is expected to be worse when the node density increases. However, from the throughput results, it seems that the gain outstrips the potential performance loss.

Finally, it is worth mentioning that although the Bi-MCMAC protocol is designed for bi-directional TCP traffic, it is even more useful if there is data traffic traveling in both directions of the same route. We explicitly did some tests on a chain network with two TCP flows in opposite directions. The results show that the average improvement (over 802.11 MAC) is 112.1% for large chains (≥ 6 hops). This is a better improvement than one TCP flow in a chain network.

6 Fairness Performance

6.1 Fairness Experiments

While the notion of "fairness" in the generic sense is quite obvious, there is no universally accepted definition of fairness in the networking research community. This is especially true in shared channel wireless networks where different flows have location-dependent contention [4, 16]. One of the consequences of this uneven contention was shown in Section 5 — the achieved TCP

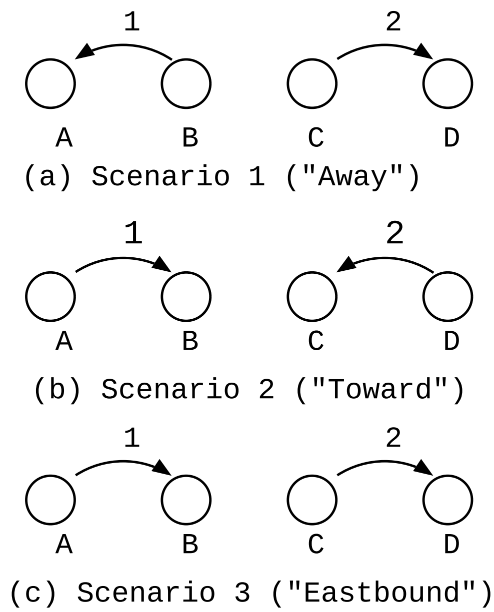


Figure 6: Scenarios to Test Fairness in Multihop Networks

throughput in a multihop wireless network decreases greatly when the number of hops increases from 2 to 8.

With the above discussion in mind, we study fairness in two ways. The first is in the strict sense: when two TCP connections traverse similar network paths and face similar competition, they should achieve similar throughput. The second study treats fairness in a more general sense: different flows should have similar throughput when they share the same physical channels, regardless of their local contention. We are not advocating which notion of fairness is better. Rather, we are trying to show the performance of the proposed protocol under both definitions.

Accordingly, for the first study, we focused on three test scenarios that are well documented in the literature [5] to have unfairness problems, and show that the proposed protocol solves the problems. These scenarios, shown in Figure 6, are tested on a chain network. The distance between nodes is 250 meters, which is the same as the wireless transmission range and carrier sensing range. Scenario 1 ("Away") has two adjacent TCP sources each sending traffic to neighbors in opposite directions (i.e., away from each other). Scenario 2 ("Toward") has two well-separated TCP sources each sending traffic to distinct neighbors. However, the two receivers happen to be adjacent, so that the two TCP flows moving toward each other may interfere. Scenario 3 ("Eastbound") has two separated TCP sources each sending traffic to its neighbor on the right.

For the second study, we tested fairness using the same experiments as in Section 5. That is, random TCP flows were created on a grid topology and on a random topology of two different sizes. The number of nodes that a flow traverses is different for each flow in all the experiments. However, these flows share a common set of channels. In a general sense of fairness, they should be allocated similar throughput.

As in Section 5, in all experiments, the application is assumed to have an infinite amount of data to send. Each experiment was executed for 300 seconds, with 50 repetitions. Jain's Fairness

Table 2: Simulated Throughput for Fairness Scenario 1 “Away” (kbps)

	802.11		MCMAC		Bi-MCMAC	
	Exp 1	Exp 2	Avg	Std	Avg	Std
Connection 1	0.20	753.79	686.93	1.07	762.59	0.42
Connection 2	753.42	0.07	686.92	1.09	762.53	0.37
Total Throughput	753.62	753.86	1373.85	-	1525.12	-

Index (FI) [11] is used to quantify the fairness. It is defined as:

$$FI = \frac{\left(\sum_{i=1}^{N} T_i\right)^2}{N \sum_{i=1}^{N} {T_i}^2} \tag{1}$$

where T_i is the throughput of TCP connection i and N is the total number of connections. Absolute fairness is achieved when $FI = 1$. The worst case unfairness is achieved when $FI = 1/N$.

6.2 Results for Strict Sense Fairness

6.2.1 Fairness Scenario 1: Away

Table 2 shows the simulation results for Scenario 1 where the two TCP flows are destined away from each other. For the MCMAC and Bi-MCMAC protocols, the average throughput and its standard deviation over 50 repetitions are shown. However for the IEEE 802.11 MAC protocol, only the results for 2 of the 50 runs are displayed. This is because the IEEE 802.11 MAC protocol is extremely unfair, with *either* Connection 1 or Connection 2 seizing almost all of the bandwidth, leaving the other connection on the verge of starvation. This unfairness occurs repeatedly in all the replications, though only two are shown here. This results in an average fairness index of 0.59, indicating extreme unfairness for the two flows.

For the multi-channel protocols, the two connections achieve roughly the same throughput. In contrast to the IEEE 802.11 MAC protocol, the average FIs are about 1.00 for both protocols. Furthermore, the throughput allocations are quite consistent: the Bi-MCMAC has a standard deviation of 0.42 for Connection 1 and 0.37 for Connection 2, calculated from the 50 replications.

These results are easily explained by the differences in the MAC protocols. In the IEEE 802.11 MAC, there is only one channel. When node B succeeds in competing for the channel, it starts sending a TCP packet to node A. When node D has a frame (e.g., a TCP ACK) to send to node C, it is difficult to acquire the channel because node C (an exposed node) will refrain from returning a CTS. The only chance for node D to succeed is if it happens to send out its RTS in the brief interval after node B finishes one data transmission and before node B sends its next RTS. Given the randomized nature of the MAC-layer backoff algorithm, this is rare. Eventually, the TCP ACK is dropped after a maximum number of RTS retransmissions, and a routing failure is reported to the routing module in node D. The routing module attempts to discover a new route to node C. Unfortunately, like the RTS frame, the routing packet is unlikely to reach node C. Lacking the TCP ACK, the TCP sender (node C) will timeout again and again, resending the TCP data packet in vain. On the other hand, if C succeeds in its TCP data transmission

Table 3: Simulated Throughput for Fairness Scenario 2 "Toward" (kbps)

	802.11		MCMAC		Bi-MCMAC	
	Exp 1	Exp 2	Avg	Std	Avg	Std
Connection 1	100.18	620.26	703.02	0.33	764.30	0.18
Connection 2	635.63	113.44	703.02	0.32	764.32	0.22
Total Throughput	735.81	733.70	1406.04	-	1528.62	-

first, then node A will lose out, leading to Connection 1 being starved. This type of repeatable behavior results in the observed unfairness.

Multi-channel MAC protocols avoid this "poor get poorer" starvation problem because contention arises only when both connections are in the control phase. Since the control phase is short compared to the data exchange phase, the contention is greatly reduced. The IEEE 802.11 MAC protocol has contention throughout the control and data exchange phases.

In addition to improving fairness, the multi-channel MAC protocols significantly increase the total throughput in Scenario 1. For example, the Bi-MCMAC protocol doubles the aggregate throughput compared to that of the IEEE 802.11 MAC. In other words, in a static multihop environment, the multi-channel MAC protocols can improve the overall network capacity.

6.2.2 Fairness Scenario 2: Toward

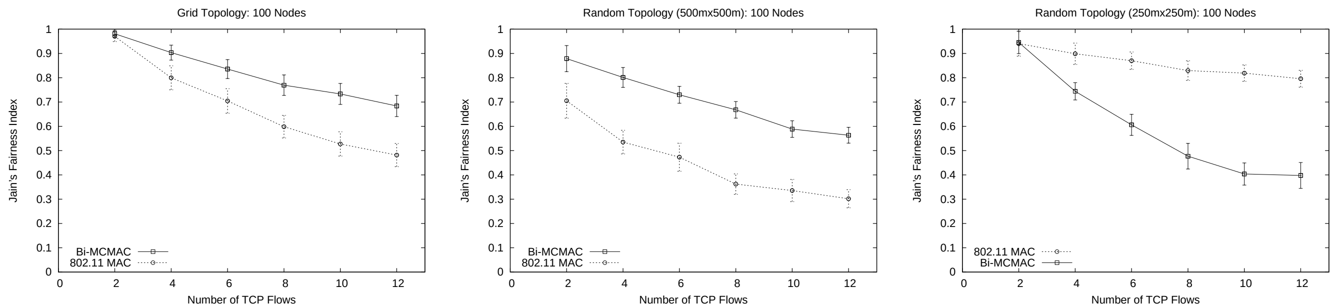
Table 3 summarizes the simulation results for fairness Scenario 2, where the two TCP connections flow toward each other. The unfairness here is not as severe as in Scenario 2, but there is still an unfairness problem for the IEEE 802.11 MAC. (In some runs, starvation occurs, but it does not occur repeatably). The unfairness arises when (for example) a successful transmission of a TCP data packet occurs from A to B. In this case, the exposed node (e.g., node C) must refrain from its CTS replies to node D, as indicated earlier. However, the time duration involved is only the transmission time of a TCP ACK packet, rather than the transmission time of a TCP data packet. Moreover, ACK packets are less prevalent than data packets, since TCP Delayed-ACK is employed. Therefore, Connection 2 has a good (but not equal) chance to obtain some network bandwidth. Again, the multi-channel protocols solve the unfairness problem and increase the aggregate throughput. The multi-channel protocols improve the average FI from 0.80 (for the IEEE 802.11 MAC) to values close to 1.00. The variation in throughput allocation is also low.

6.2.3 Fairness Scenario 3: Eastbound

Scenario 3 with "eastbound" traffic is the harshest fairness test case, because node B is an exposed node for every packet sent by node C. With the IEEE 802.11 MAC protocol, Connection 2 obtains all of the network capacity, and Connection 1 always starves, as shown in Table 4. The average throughput of Connection 1 is zero, repeatably. The two multi-channel protocols improve the fairness (and the aggregate throughput), although Connection 2 always has a slight advantage (slightly higher throughput, much lower standard deviation). As a result, the average FIs of the multi-channel protocols are very close to 1.00, while the average FI of the IEEE 802.11 MAC protocol is 0.50.

Table 4: Simulated Throughput for Fairness Scenario 3 “Eastbound” (kbps)

	802.11		MCMAC		Bi-MCMAC	
	Avg	Std	Avg	Std	Avg	Std
Connection 1	0.00	0.00	691.91	14.36	757.40	5.29
Connection 2	754.00	0.12	693.27	1.05	763.19	0.37
Total Throughput	754.00	-	1385.18	-	1520.59	-



(a) Grid Topology

(b) Random Topology (500m x 500m)

(c) Random Topology (250m x 250m)

Figure 7: Fairness for the Grid and Random Topology

6.3 Results for General Sense Fairness

Figure 7 shows simulation results for general sense fairness. From the graph, it can be seen that for low density topologies (the grid and the 500m x 500m random topology) the Bi-MCMAC protocol has better fairness than that of the 802.11 MAC protocol. When the number of flows is small in a low density multihop network, different flows are not likely to compete with each other. When the number of flows increases, the contention between flows increases, leading to unfairness.

However, in the 250m x 250m topology (shown in Figure 7 (c)), the Bi-MCMAC protocol has *worse* fairness than that of the 802.11 MAC protocol. The reason for this is explained as follows. Since the Bi-MCMAC protocol has better throughput performance, it also has more RTS/CTS/CRN handshakes. Since the 250m x 250m topology is almost a single-cell topology, the control channel could become congested (i.e., it has a higher collision rate). The higher collision rate causes ad hoc routing messages to be lost[2]. The net effect is that some flows have difficulty establishing a route and are always in a disadvantaged state. Detailed analysis of the simulation results shows that most disadvantaged flows have zero throughput, confirming this reasoning.

While the fairness performance is worse in the 250m x 250m random topology, it still has better throughput performance, as shown in Section 5. The Bi-MCMAC protocol was designed to improve TCP performance over multihop ad hoc networks. Its worse fairness performance in

[2]Note that routing messages are more vulnerable to collision and loss since they are broadcast packets, which do not undergo MAC-layer retransmission.

the single-cell case is a minor drawback.

7 Delay Performance

7.1 Delay Experiments

Unlike the throughput experiments that simulate FTP-like data transfers, the delay experiments simulate Web-like transfers. A chain topology with 10 nodes was chosen in this study. Two sets of experiments were conducted. In the first set of experiments, one TCP connection was established between the first node and the last node to simulate the transmission of one Web object. The number of packets in a Web object was varied from 2 to 142, each having 512 bytes. The transfer time for each Web object was recorded.

The first set of experiments can be viewed as transfering Web pages containing only one Web object. In real Web applications, a Web page typically contains several Web objects. In other situations, multiple Web object transfers from different Web sessions may share the same route. The second set of experiments simulate these situations. More specifically, in these experiements, the Web object size was fixed as 10 packets, while the number of Web objects was varied from 2 to 20 (i.e. the number of TCP connections between the first node and the last node varied from 2 to 20). We started the transfers uniformly at random between 0 and 1000 ms. The transfer time for each Web object was recorded and the average transfer time per object was then calculated.

All of the experiments were repeated 50 times, with different seeds for the MAC backoff timer in each repetition. The mean and 99% confidence intervals were calculated.

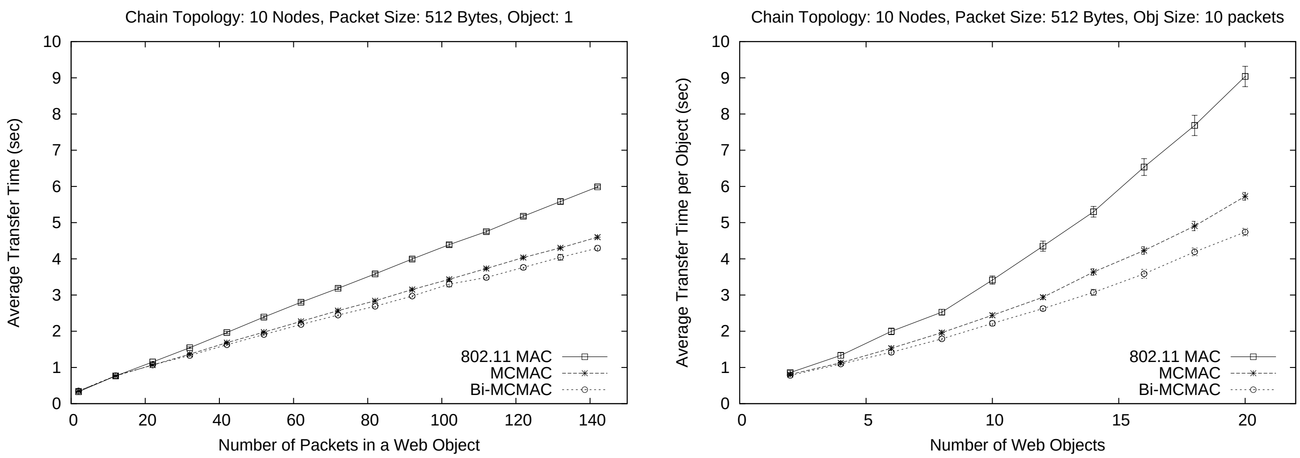
7.2 Simulation Results

Figure 8 (a) shows the average transfer time for a single Web object in a 10-node chain multihop wireless network. The transfer time is smaller for Web objects larger than 12 packets, when multichannel protocols are used. When the Web objects are small, the number of packets in flight is low, due to the TCP congestion control restriction. Few packets in flight leads to low contention along the route. The multichannel protocols have no obvious advantage, since the CRN overhead causes larger delay.

When the Web objects get larger, the multichannel protocols perform better than the 802.11 MAC protocol. As the TCP congestion window size becomes larger, more packets are in flight. As a result, the contention along the route is more severe. The 802.11 MAC protocol performance degrades as shown in Section 5.

When the Web objects are larger than 32 packets, the Bi-MCMAC protocol outperforms the MCMAC protocol in transfer time. Again, this is due to the large number of packets in flight. With a large number of packets in flight, there are also many TCP ACKs returning in the opposite direction. The Bi-MCMAC protocol has the advantage of transferring a data packet and a TCP ACK in a single handshake, therefore reducing the total transfer time.

Figure 8 (b) shows the simulation results for multiple Web objects sharing a single route. The Bi-MCMAC protocol has the best performance when there are many Web objects. Although the number of packets from each individual Web object is small, packets from other objects are traversing the same route, providing enough TCP ACKs in the opposite direction for the Bi-



(a) Single Web Object

(b) Multiple Web Objects

Figure 8: Simulated Web Transfer Time in a Chain Network

MCMAC protocol to exploit. These results show the benefits of Bi-MCMAC protocol for both intra-flow and inter-flow contention.

8 Conclusion

In this paper, we proposed and evaluated a novel bidirectional multi-channel MAC-layer protocol. The protocol is explicitly designed to improve TCP performance over a static multihop wireless ad hoc network.

The proposed protocol is a logical extension of the conventional IEEE 802.11 MAC protocol. The basic 802.11 RTS/CTS handshake is augmented with channel negotiation and reservation mechanisms. Moreover, after a successful RTS/CTS handshake, bidirectional data exchange is allowed, with one frame in each direction. Although the protocol design is explicitly targeted at TCP, which induces loosely-synchronized forward and backward traffic (data and ACKs), we expect that our Bi-MCMAC protocol will improve the overall performance for general bidirectional traffic (from different connections) in a multihop wireless network.

The ns-2 network simulator was used to evaluate TCP performance with the proposed protocol, in terms of throughput, fairness, and delay. Simulation results show significant throughput gains in static chain topologies, grid topologies, and random topologies, with an improvement of 50% to 185%, depending on the node density. For a simple 2-node network, the TCP throughput of the Bi-MCMAC protocol is slightly higher than that of the IEEE 802.11 MAC protocol, in spite of the fact that Bi-MCMAC has more control overhead.

Our simulation results also show excellent strict sense fairness results for the three fairness scenarios tested. This is true for both multi-channel protocols. On the contrary, the 802.11 MAC protocol shows unfair TCP performance. In the worst case, the IEEE 802.11 MAC protocol completely starves one of the two competing TCP connections. For general fairness, our protocol shows better performance for all topoplogies except in the single cell case where the node density is very high. However, since the protocol is intended to use in a multihop environment, this drawback is not a big problem.

We also simulated the transfer time for Web-like traffic in a chain topology. Results show that Bi-MCMAC has lower transfer time when the number of packets in a Web object is larger than 12 packets (512 bytes each). When multiple Web objects share the same chain network, the Bi-MCMAC protocol shows equally significant performance gains.

Ongoing work is evaluating the performance impacts of node mobility and ad hoc routing. The possible extensions mentioned in Appendix B are also on our research agenda.

Appendix A: Estimation of the Protocol Efficiency

To estimate the RTS/CTS/ACK overhead, we consider a two-node topology where one node is a TCP sender and the other is a TCP receiver. Referring to Figure 2, the total time needed to transmit a data packet over the 802.11 MAC is: $DIFS + RTS + SIFS + CTS + SIFS + data + SIFS + ACK$. If the TCP packet size is assumed to be 1024 bytes, the MAC layer data frame size will be 1076 bytes, since the 802.11 MAC layer adds a 52-byte header to each packet. Assuming a MAC transmission rate of 1 Mbps, the time required to transmit a data frame is therefore 8608 μs. For the 802.11 MAC, the RTS frame size is 44 bytes, while both the CTS and ACK frame sizes are 38 bytes. Accordingly, the time required to transmit an RTS frame and a CTS/ACK frame is 352 μs and 304 μs respectively (1 Mbps transmission rate). In a Direct Sequence Spread Spectrum physical layer, the DIFS and SIFS values are 50 μs and 10 μs respectively [3]. Therefore, the total time needed to transmit a 1024-byte packet over a 802.11 MAC layer is $8608 + 352 + 304 * 2 + 50 + 30 = 9648$ μs. For a TCP packet, it takes two RTS/CTS/ACK cycles to complete the data transfer since a cycle is needed to send the TCP ACK back to the sender. Similarly, the TCP ACK cycle time is $736 + 352 + 304 * 2 + 50 + 30 = 1776$ μs, assuming the TCP ACK size is 40 bytes. The efficiency is therefore $1024 * 8/(9648 + 1776) = 71.7\%$. When considering TCP delayed ACK, in steady state, two TCP packets are sent within three RTS/CTS/ACK cycles. The efficiency is similarly estimated to be 77.8%.

For the MCMAC and Bi-MCMAC protocols, the RTS, CTS, and CRN frame sizes are increased since they need to convey channel reservation information. In our simulation, they are all 45 bytes. The ACK frame and the MAC layer header are the same as those of 802.11 MAC protocol. Using similar method, the efficiency of TCP (with delayed ACK option) over the MC-MAC is 73.2%, which is smaller than 77.8% because of the CRN overhead. For the Bi-MCMAC protocol, it only takes two cycles to transmit two packets, because in the second cycle the TCP data and ACK can be transmitted within the same handshake. With this in mind, the efficiency is estimated to be 78.4%.

Note that in the above estimation, the propagation delays are ignored. If taken into account, they should have similar effect on all MAC protocols. Collisions are also ignored to estimate the ideal throughput.

Appendix B: Discussion

Appendix B.1: Implementation-Related Issues

The proposed Bi-MCMAC can be implemented with current network hardware platforms. The bidirectional MAC requires the RTS receiver to check if it has data waiting to be sent to the RTS sender. A potential head-of-line (HOL) blocking could happen if the first packet in the buffer is not destined to the sender. A per-neighbor queue solves this problem. The node receiving the RTS must check the queue corresponding to the source address of the RTS frame to determine if there is data to send back.

Another issue is the implementation of the Virtual Carrier Sense (VCS). Unlike the IEEE 802.11 MAC protocol, we need a VCS for each channel. Furthermore, a VCS must be associated with some MAC address. After the control phase, a node is free to send an RTS to any nodes except those in a data exchange phase. That is, when a node has data to send, it first checks the VCS status for the intended destination address. The sender may refrain from transmitting an RTS to that node, depending on the indication.

One way to do this is to add a flag in the routing table. When the VCS of the channel for the desired destination address is detected, the flag is set. When the VCS clears, the flag is also cleared. However, this approach would violate the layering principle of the protocol stack, since the MAC layer accesses network-layer information.

A better method is to implement VCS with the forwarding scheme proposed by Acharya [1]. In this approach, a label switching table is maintained in the MAC layer to enable "cut-through" forwarding of frames. The VCS flag can be maintained in the label switching table.

Appendix B.2: Extensions

This paper presents initial results for our bidirectional multi-channel MAC protocol. Several possible extensions of this work are immediately apparent. We comment on a few of these here.

In the proposed protocol, after a successful RTS/CTS/CRN handshake, there is only one MAC frame exchanged in each direction. This idea can easily be extended to a "batch mode" that allows multiple data frames to be exchanged following each handshake.

Intuitively, batch mode offers two advantages. First, multiple data frames per handshake amortize the handshake overhead. Second, reserving multiple frame transmissions at a time can reduce contention. However, transmitting more data means that the two nodes spend more time in the data exchange phase, which makes the reservation missing problem more severe. We expect that there exists an optimal number of frames exchanged per handshake. We leave this for further study.

The second extension would be to marry batch mode with TCP, so that an entire window of TCP packets is exchanged following a single handshake. Coupled with the ACK-clocking nature of TCP, this protocol in steady-state would operate like Stop-and-Wait. That is, the sender sends W packets, and waits for the acknowledgments. It then sends another W packets, and so on. In this way, no contention occurs. We do not favor this approach, however, for reasons explained below.

Assuming there are h hops in a chain and the packet size is a, the time for a burst of W TCP packets to reach the last node in a chain network is $\frac{8hWa}{C}$ seconds. If the ACK size is b

bytes, then a burst of W TCP ACKs reaches the origin node after $\frac{8hWb}{C}$ seconds. The total time to exchange W packets is thus $\frac{8hW(a+b)}{C}$ seconds, with throughput $\frac{C}{h(a+b)}$ bps. In this expression, the maximum throughput is inversely related to the number of hops in a chain, which is not very scalable. In contrast, the upperbound results shown in Section 5.2 do not depend on the number of nodes in a chain.

A final research direction worth pursuing is the packet scheduling algorithm. In a complex topology, HOL blocking may be a problem in a conventional First-In-First-Out packet queue if the destination desired by the front packet is busy. With a multi-channel MAC protocol, packets destined to other neighbors may be eligible for transmission. Using a per-neighbor queue (as discussed in Section 8), an algorithm can check other queues if the current packet is blocked. This may improve the overall system performance. We are currently exploring these ideas.

References

[1] A. Acharya, A. Misra, and S. Bansal "A Label-switching Packet Forwarding Architecture for Multi-Hop Wireless LANs", *Proceedings of WoWMoM'02*, Atlanta, USA, September 2002.

[2] A. Acharya, A. Misra, and S. Bansal, "MACA-P: A MAC for Concurrent Transmissions in Multi-hop Wireless Networks", *Proceedings of IEEE Pervasive Computing and Communications Conference*, Dallas, USA, March 2003.

[3] ANSI/IEEE Standard 802.11b, "Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Higher-Speed Physical Layer Extension in the 2.4 GHz band", 1999.

[4] C. Barrett and D. Engelhart, "Analyzing the Short-Term Fairness of IEEE 802.11 in Wireless Multi-hop Radio Networks", *Proceedings of MASCOTS'02*, Fort Worth, TX, USA, October 2002.

[5] V. Bharghavan, "MACAW: A Media Access Protocol for Wireless LAN's", *Proceedings of ACM SIGCOMM'95*, Cambridge, USA, August 1995.

[6] C. Cordeiro, S. Das, and D. Agrawal, "COPAS: Dynamic Contention-Balancing to Enhance the Performance of TCP over Multi-hop Wireless Networks", *Proceedings of IC3N'02*, Miami, USA, October 2002.

[7] S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance", *IEEE/ACM Transactions on Networking*, Vol. 1, No. 4, August 1993.

[8] S. Floyd and T. Henderson, "RFC 2582: The NewReno Modification of TCP's Fast Recovery Algorithm", April 1999.

[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", *Proceedings of IEEE INFOCOM'03*, San Francisco, USA, April 2003.

[10] W. Hung, K. Law, and A. Leon-Garcia, "A Dynamic Multi-Channel MAC for Ad Hoc LAN", *Proceedings of 21st Biennial Symposium on Communications*, Kingston, Canada, June 2002.

[11] R. Jain, *The Art of Computer Systems Performance Analysis*, John Wiley & Sons, 1991.

[12] N. Jain, S. Das, and A. Nasipuri, "A Multichannel CSMA MAC Protocol with Receiver-Based Channel Selection for Multihop Wireless Networks", *Proceedings of the IEEE IC-CCN'01*, Phoenix, USA, October 2001.

[13] J. Louderback, "4 Simultaneous Channels Okay For 802.11b", ExtremeTech Article, available at: http://www.extremetech.com/article2/0,3973,709067,00.asp

[14] J. Li, C. Blake, D. Couto, H. Lee, and R. Morris, "Capacity of Ad Hoc Wireless Networks", *Proceedings of ACM MobiCom'01*, Rome, Italy, July 2001.

[15] J. Monks, V. Bharghavan, and W. Hwu, "A Power Controlled Multiple Access Protocol for Wireless Packets Networks", *Proceedings of IEEE INFOCOM'01*, Anchorage, USA, April 2001.

[16] T. Nandagopal, T. Kim, X. Gao, and V. Bharghavan, "Achieving MAC Layer Fairness in Wireless Packet Networks", *Proceedings of MOBICOM'00*, Boston, MA, USA, August 2000.

[17] J. So and N. Vaidya, "A Multi-channel MAC Protocol for Ad Hoc Wireless Networks", Technical Report, Dept. of Computer Science, University of Illinois at Urbana-Champaign, January 2001.

[18] K. Tan and M. Gerla, "Fair Sharing of MAC under TCP in Wireless Ad Hoc Networks", *Proceedings of IEEE MMT'99*, Venice, Italy, October 1999.

[19] R. Theodore, *Wireless Communications: Principles and Practice*, Second Edition, Prentice Hall PTR, 2002.

[20] A. Tzamaloukas and J. Garcia-Luna-Aceves, "A Receiver-Initiated Collision-Avoidance Protocol for Multi-Channel Networks", *Proceedings of IEEE INFOCOM'01*, Anchorage, USA, April 2001.

[21] S. Xu and T. Saddawi, "Does the IEEE 802.11 MAC Protocol Work Well in Multihop Wireless Ad Hoc Networks?", IEEE Communications Magazine, June 2001.

[22] VINT Group, "Network Simulator ns-2 (version 2.26)", available at: http://www.isi.edu/nsnam/ns.

[23] S. Wu, Y. Tseng, C. Liu, and J. Sheu, "A Multi-Channel MAC Protocol with Power Control for Multi-Hop Mobile Ad Hoc Networks", The Computer Journal, 45(1): 101-110, 2002.