Alleviating Self-Interference in MANETs

Alex Varshavsky and Eyal de Lara
Department of Computer Science
University of Toronto
{*walex,delara*}*@cs.toronto.edu*

Abstract

The interference range in multi-hop ad hoc networks (MANETs) is typically twice as large as the transmission range. This phenomena causes packets of a multi-hop flow to interference with each other as they are relayed over the multi-hop route. This interference, an instance of the notorious hidden terminal problem, is caused by simultaneous transmissions by down-stream nodes unaware of ongoing transmissions by up-stream nodes.

DMAC is a novel MAC protocol that alleviates the hidden terminal problem by deferring further transmissions until the previously transmitted packets travel far enough to avoid interference with the newly transmitted packets. For simple chain topologies, DMAC improves the throughput of CBR and TCP flows by up to 100% and 60%, respectively. For random mobile topologies with up to 40 simultaneous flows, DMAC improves the throughput of TCP flows by up to 30%.

1. Introduction

A multi-hop mobile ad hoc network (MANET) is a group of mobile wireless-enabled nodes that communicate by relaying packets through intermediate nodes. MANETs do not require the presence of preexisting infrastructure, forming instead a cooperative impromptu network. Due to their ease of deployment, MANETs have been proposed for emergency relief and military communications.

Typically, nodes in MANET communicate over the same wireless channel. As a result, closely located nodes cannot transmit simultaneously. Instead, nodes contend for the wireless channel following a media access protocol (MAC), which runs on every node. The most common distributed MAC protocol used in MANETs today is the IEEE 802.11b Distributed Coordination Function (DCF) [9].

Traditionally, IEEE 802.11b DCF was designed for infrastructure based networks[1], and thus it is not surprising that they exhibits deficiencies when applied in multi-hop scenarios. Typically, the interference range of a node in MANETs is twice as large as the transmission range. As a result, packets of the same flow may interfere with each other as they are retransmitted along a multi-hop path. This form of interference between packets of the same flow, or *self-interference*, has been shown to cause severe degradations in network throughput [12]. This problem is an instance of the notorious *hidden terminal problem* because nodes down the route are not aware of concurrent transmission by upstream nodes and are thus being *hidden*.

This paper presents deferrable MAC (DMAC), a novel MAC protocol that alleviates the hidden terminal problem by deferring packet transmissions until previously transmitted packets have traveled far enough along the multi-hop path to avoid self-interference. In DMAC, the deferral interval is correlated with the route length to the destination. If the packet is targeted for an immediate neighboring node, no retransmission will be necessary and therefore the next packet will not be deferred. Conversely, if the packet needs to be retransmitted by multiple intermediate nodes to reach the destination, a node will defer its next packet transmission to allow the packet to advance far enough to avoid collisions with the newly transmitted packets. DMAC obtains the route length to the destination either from the packet itself (if the route length is embedded into the packet) or from the routing layer via a cross-layer optimization.

We evaluated DMAC in both static and mobile environments. Experimental results show that in chain topologies, DMAC doubles the achievable throughput of the standard 802.11b MAC for Constant Bit Rate (CBR) flows, and achieves an improvement of up to 60% for TCP flows. Furthermore, for random mobile networks with up to 40 simultaneous TCP flows, DMAC improves network throughput by up to 30%. The fact that even in mobile networks with multiple interfering flows DMAC achieves performance gains over the traditional MAC attests to the significance of the self-interference problem in MANETs.

1 Infrastructure based networks are networks where nodes communicate directly with the base station or an access point.

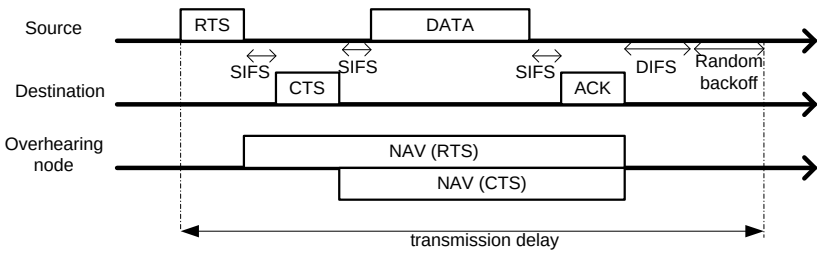


Figure 1: Operation of IEEE 802.11b DCF

The rest of this paper is organized as follows. In Section 2, we describe the IEEE 802.11b DCF and the hidden terminal problem. Section 3 presents the DMAC protocol, and Section 4 discusses experimental results. We describe related works in Section 5. Finally, Section 6 concludes the paper and presents avenues for future research.

2. Background

In this section, we first describe the IEEE 802.11b Distributed Coordination Function (DCF) [9]. We then describe the hidden terminal problem.

2.1. Distributed Coordination Function (DCF)

The IEEE 802.11b DCF reduces the likelihood of packet collisions by making the neighboring nodes of both the transmitter and receiver aware that a data transmission is about to begin. The operation of IEEE 802.11b DCF is summarized in Figure 1. In DCF, before transmitting the actual data packet, a node transmits a short Request-To-Send (RTS) packet. On receiving the RTS, the destination node senses the channel for a Short InterFrame Space (SIFS) interval, and if the channel is idle replies with a Clear-To-Send (CTS) packet. Both RTS and CTS packets contain the expected duration of a time for which the channel will be in use. Whenever a host overhears a RTS or CTS, it defers its transmission for the duration specified in the packet. The duration for which the node has to defer its transmission is stored in the variable called the Network Allocation Vector (NAV). On receiving the CTS packet and sensing the channel for SIFS interval, the source node starts the data transmission. Once the DATA packet is successfully received, the destination node senses the channel for SIFS interval, and sends an acknowledgment (ACK) to the source node. After successfully receiving the ACK, the station senses the channel for Distributed InterFrame Space (DIFS) interval and then defers its next transmission for a randomly chosen interval. DCF drops a data packet if it fails to perform the RTS-CTS exchange 7 consecutive times or when it fails receive an ACK after 4 consecutive data packet retransmissions.

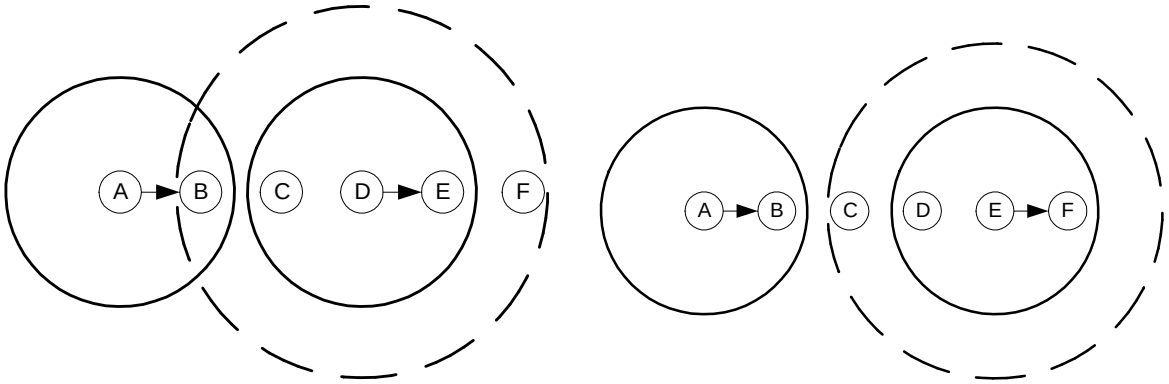
DCF, however, does not eliminate all collisions. We next discuss an instance of the hidden terminal problem that is not addressed by DCF and which as we will show in Section 4 can dramatically reduce MANET performance.

2.2. The Hidden Terminal Problem

A hidden terminal is a node that is unaware of a transmission in its vicinity and whose attempt to transmit data will corrupt another ongoing transmission. We refer to the existence of hidden terminals in a network as the *hidden terminal problem*. We further refer to the range at which nodes can successfully receive packets as the *transmission range*, and to the range at which a transmission will corrupt other ongoing transmissions in other parts of the networks as the *interference range*. The interference range is usually more than twice the transmission range of a node, which makes hidden terminals a serious problem in MANETs.

The 802.11b DCF works reasonably well on wireless LANs where all communication takes place through an access point and packets are only retransmitted from the access point to the wireless receivers. The RTS-CTS exchange, however, fails to eliminate the hidden terminal problem when packets are retransmitted over multiple hops. Consider a chain of nodes depicted in Figure 2(a) where packets are being sent from A to F. Assume that node A is sending packets to node B. The solid circles represent the maximal transmission range of nodes A and D and the dashed circle represents the interference range of node D. Now, assume that node D decides to transmit a RTS packet to node E. Note that node D is unaware of the ongoing transmission because it received neither A's RTS nor B's CTS packets. A transmission of a packet by node D will therefore corrupt packets received by node B.

In contrast, Figure 2(b) shows an example of a successful spatial channel reuse. Nodes A and E are far enough from each other, and therefore can transmit packets simultaneously without interfering with each other.



(a) Hidden Terminal Problem

(b) Successful transmission

Figure 2: Collision and Successful transmissions among chain of nodes. Nodes A and D cannot transmit simultaneously, but A end E can.

3. Deferrable MAC

DMAC leverages the observation that for flows that span multiple hops, it is possible to determine how far a packet needs to advance over the multi-hop route before it is possible to transmit a new packet, such that subsequent transmissions of the new and old packets are likely not to interfere with each other. We further refer to the time a node needs to defer its transmission to avoid collisions with the previously transmitted packet as *deferral interval*.

The deferral interval is dependent on the route length to the destination. Consider a chain of nodes depicted in Figure 2(a). If node A transmits packets to node C, A needs to defer for an interval long enough to allow successful retransmission of the packet by node B. Moreover, if a destination is more than 3 hops away from the source, at least 3 deferrals are necessarily. This is depicted by chain of nodes in Figure 2(b), where nodes A and E are far enough from each other to allow simultaneous transmissions.

Figure 3 shows the pseudo-coded algorithm for computing the interval a node needs to defer after successfully transmitting a packet. The algorithm accepts a route length to the destination and the packet size as parameters. The delay for one packet transmission is calculated according to the description in Figure 1. Note that DMAC does not maintain additional state since only one deferral interval needs to be maintained at any time. The algorithm takes transmitted packet size into account, and will work correctly even if a stream of packets with multiple packet sizes is transmitted.

Although the minimum amount of time a node needs to defer after transmitting a packet is equal to 3 transmissions, in practice, a node may need to defer more if its neighbor nodes are grouped densely together. We have experimented with various deferral intervals, and discovered that 3 works best in practice. A more accurate way to measure the deferral interval would be for a node to listen to broadcasted packets in its surrounding, and estimate the number of nodes of the same flow it may interfere with. We plan to explore this option in the future.

Since route lengths are maintained by the routing layer, they are not available to the MAC layer directly. For routing protocols that include route to the destination in the packet (e.g., Dynamic Source Routing (DSR) [10]), DMAC can simply extract the route length directly from the packet. If route is not present in the packet, DMAC has to obtain the route length from the routing layer. Currently, we have implemented DMAC under DSR routing protocol.

In networks with multiple competing flows, transmissions by nodes of different flows are not correlated, and therefore may collide or interfere with each other, rendering our scheme suboptimal. However, as we will see in the next section, DMAC outperforms the traditional MAC even in highly congested networks.

4. Evaluation

We used simulations to compare DMAC to the traditional 802.11b MAC. We ran our experiments on the `ns-2` simulator [6] with CMU wireless extension [13]. We chose physical radio characteristics that approximate the Lucent WaveLAN direct sequence spread spectrum radio with a 250m nominal transmission range, 550m interference range and a raw capacity of 2Mb/s.

In the rest of this section, we first quantify the advantages of DMAC for static chain topologies using both Constant

```
function calculate_defer_interval(int route_length, int packet_size)

// RTS_SIZE, CTS_SIZE and ACK_SIZE are constants
double one_defer_interval =
          propagation_delay(RTS_SIZE)
        + SIFS
        + propagation_delay(CTS_SIZE)
        + SIFS
        + propagation_delay(packet_size)
        + SIFS
        + propagation_delay(ACK_SIZE)
        + DIFS
        + RANDOM_AVERAGE_BACKOFF;

// assuming that route_length > 0, node should ignore self transmission
transmissions_to_defer = route_length - 1;

// defer for 3 transmissions
if (transmissions_to_defer > 3)
        transmissions_to_defer = 3;

return transmissions_to_defer*one_defer_interval;
```

Figure 3: Algorithm that calculates the defer interval of a station after a packet transmission.

Bit Rate (CBR) and TCP traffic sources. Then, we evaluate the advantages of DMAC for random static and mobile topologies with multiple flows. We further refer to the version of TCP that runs over the DMAC as DTCP.

4.1. Static Chain Topologies

In this section, we quantify the benefits of DMAC in terms of throughput in chain topologies using static routing. Nodes are positioned 200 meters apart. The first node in the chain transmits 1000 byte packets toward the last node in the chain for 100 seconds.

CBR Figure 4(a) depicts the measured average throughput as a function of the sending rate for CBR flow. Different lines represent chains of different length (higher lines represent chains of shorter length). Every point in the graphs has been averaged over 50 runs.

For chains of 2,3 and 4 nodes the throughput is gradually increasing up to a certain point and then stays flat, reaching the network capacity. This is an expected result since with up to 4 nodes, no hidden terminals are present in the network. Nodes overhear one another's transmissions and no packet drops occur. The extra packets produced by higher sending rates overflow the sender's buffer and are being discarded. However, for the chains of more than 4 nodes, DCF fails to optimally schedule packet transmissions, causing contention between nodes that have packets to transmit. As a result, a decrease of up to 55% of the optimal throughput may be experienced!

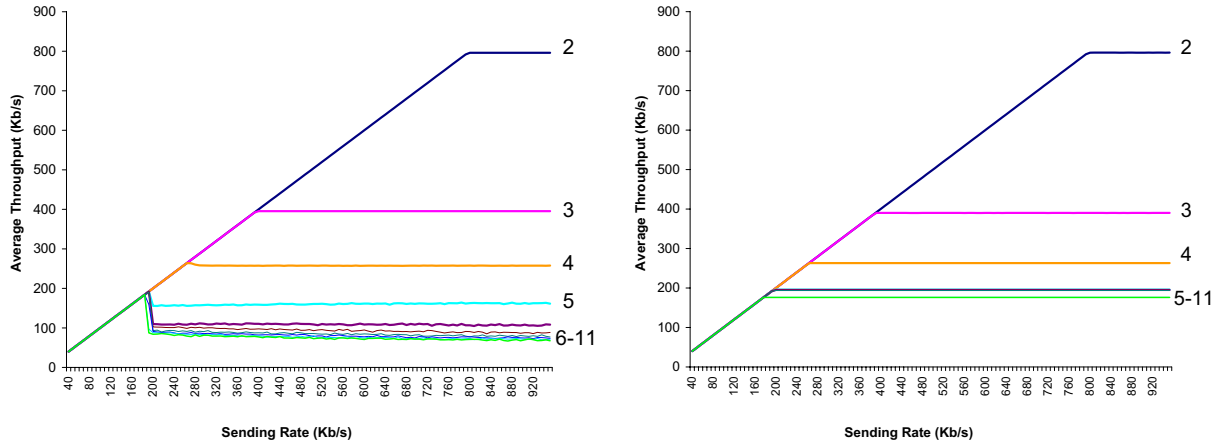
Figure 4(b) shows the results for the same experiment run over the proposed DMAC protocol. In contrast to 802.11b MAC, the throughput over DMAC does not decrease as the sending rate increases for all chains. Therefore, applications running over DMAC do not have to carefully tune their sending rate [2] to achieve the optimal throughput. Note that the optimal sending rate is dependent on both the packet size and the route length. Therefore, tuning the sending rate for applications that transmit packets with different packet sizes may be even more complicated, whereas DMAC handles various packet sizes automatically.

TCP Figure 5(a) compares the measured average throughput of TCP flows on chains of various length. Clearly, by deferring packet transmissions, DTCP incurs less contention and consequentially beats TCP by up to 60% (for 5-node chain).

The comparison between packet drops as a result of MAC layer contention is depicted in Figure 5(b). There are no drops for chains of 2 and 3 nodes because nodes overhear one another's transmissions and consequentially no hidden terminals are present in the network.

For chains of four and more nodes, both DMAC and MAC experience packet drops, which are the consequence of interference between data packets flowing toward the destination and TCP acknowledgments flowing back to the source node. Currently, DMAC does not optimize streams

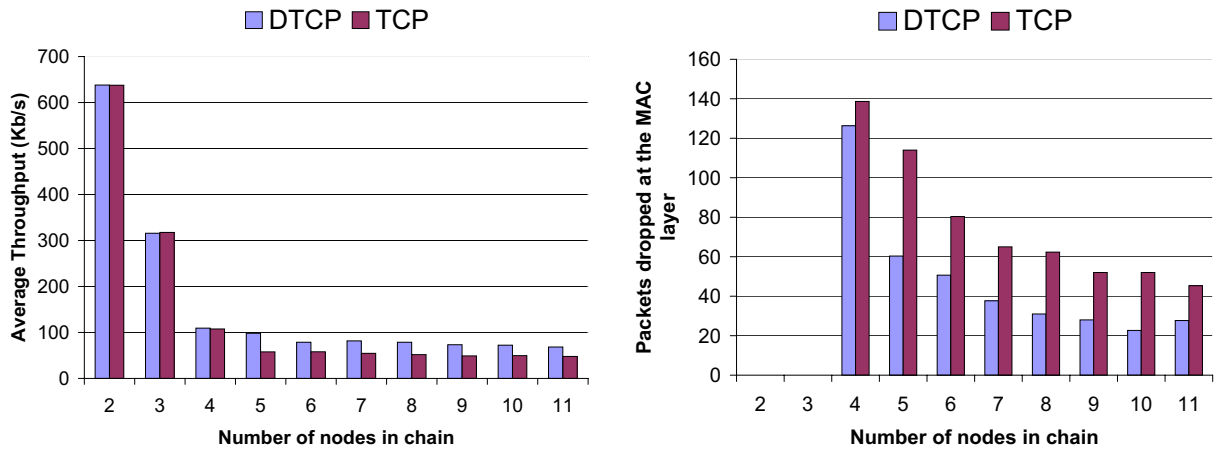
2 Tuning the sending rate may not be feasible for the application layer programs since the optimal sending rate is dependent on the constantly changing route length in multi-hop mobile ad hoc networks.



(a) 802.11b DCF

(b) Deferrable 802.11b DCF

Figure 4: CBR throughput as a function of sending rate. Lines represent chains of different lengths (the numbers beside the lines represent chain length)



(a) Average Throughput

(b) Dropped Packets

Figure 5: Comparison of achievable throughput between TCP runs over 802.11b MAC and DMAC protocols for chains of different lengths.

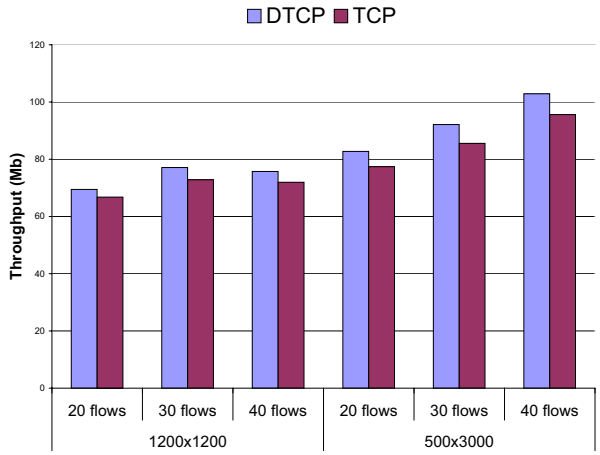


Figure 6: **[Stationary Network]** Comparison of TCP and DTCP for 20,30 and 40 flows.

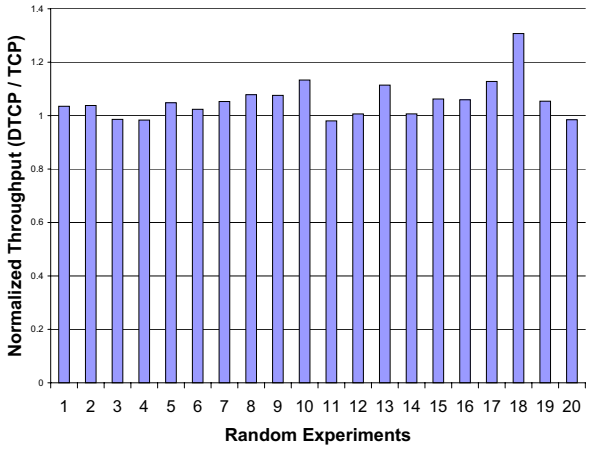


Figure 7: **[Static Network, 500m x 3000m, 30 flows]** Normalized throughput (DTCP throughput / TCP throughput) of 20 random runs.

of data flowing in opposite directions, but instead handles them similarly to IEEE 802.11b MAC.

4.2. Random Topologies and Multiple Flows

In this section, we evaluate the performance of TCP and DTCP in terms of throughput in both stationary and mobile networks. The results are for networks of 100 nodes randomly placed on both rectangular 500m x 3000m and square 1200m x 1200m flat spaces. In both scenarios, every node has about 13 neighbors on average, enough to preserve connectivity even in mobile networks. Similarly to related research [3, 5], the rectangular shape was chosen to force the use of longer routes between communication pairs. Both protocols send data packets of 1000 bytes. All experiments, use DSR as the underlying routing protocol.

We have experimented with 20, 30 and 40 simultaneous flows. Each flow is defined by a pair of randomly chosen source and destination nodes. In each experiment, backlogged TCP sources stream packets toward the destination nodes for 100 seconds.

Stationary Networks Figure 6 compares the overall throughput for 20, 30 and 40 flows. The results for both rectangular (1200m x 1200m) and square (500m x 3000m) areas are presented. Each point in the graph is averaged over 20 random topologies. The bars represent the achieved throughput in terms of Megabits received. Interestingly, the overall throughput results for the rectangular area are more impressive than for the square area. This is because in the rectangular area less contention and interference occurs, allowing for more simultaneous transmissions.

DTCP consistently outperforms TCP for both square and rectangular topologies.

The performance advantage of DTCP over TCP is more significant for the rectangular scenarios. This is because the average route length for the square scenario is 4, and consequentially, as was shown in Figure 5(a), TCP and DTCP tend to achieve similar results. In contrast, the average route length for rectangular topologies is 8.5, which results in better DTCP behavior.

Figure 7 shows the normalized throughput of the 20 experiments with 30 simultaneous flows over different random static topologies. DTCP outperforms TCP in 16 out of 20 runs and achieves performance improvements of up to 30%. The average performance improvement of DTCP over all runs was 10%. TCP tends to send packets at bursts, while DTCP paces the transmissions. Therefore, in situation where many closely positioned nodes interfere with each other, TCP may occasionally outperform DTCP.

Mobile Networks Figures 8 compares the throughput results for both rectangular and square topologies in mobile networks for 20, 30, and 40 simultaneous flows. Nodes move following the random waypoint model [3] with no pause time to stress test our evaluation. In this model, a node chooses a random point within the space and starts moving toward that point at a speed randomly chosen from an interval 0-V_{max} (in our experiments V_{max} is equal to 10m/s). Upon reaching its destination, the node selects another destination and speed, repeating the process.

Similarly to static topologies, DTCP consistently outperforms TCP. For example n for 40 simultaneous flows DTCP achieve an performance improvement of 9%.

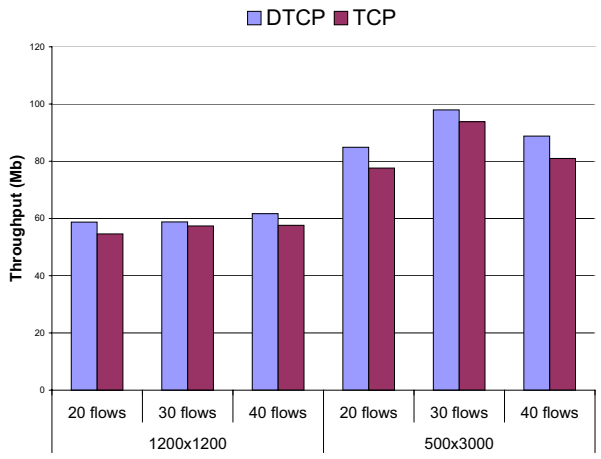


Figure 8: **[Mobile Network]** Comparison of TCP and DTCP for 20,30 and 40 flows.

4.3. Summary and Discussion

In summary, DMAC proved effective in both stationary and mobile topologies. For CBR flows, DMAC achieves double the throughput for chains of 5 and more nodes. TCP flows over DMAC consistently outperform TCP flows over the regular 802.11b MAC protocol, and achieve up to 30% improvement in throughput. We believe that such a simple optimization as deferring transmission according to the route length to the destination is worth the effort to modify the MAC layer.

5. Related Work

The hidden terminal problem was reported [7] to degrade the performance in carrier sense multiple access (CSMA) protocols [14] substantially. The busy tone multiple access (BTMA) protocol was proposed to tackle the hidden terminal problem in CSMA based networks [15]. The idea behind BTMA is that either a base station or a receiver of a packet transmits a busy-tone signal over a separate channel. The nodes that overhear the busy tone refrain from transmitting. Ye *et al.* [16] have proposed a jamming-based MAC, where a source node traffic is being transmitted on a separate channel from a destination node traffic. Bertossi and Bonuccelli [2] investigate the problem of assigning different channels for simultaneous transmissions to avoid the hidden terminal problem. The main disadvantage of this schemes is that a separate busy-tone channel needs to be allocated for the busy-tone transmissions. In contrast, our work concentrates on alleviating the hidden terminal problem in single-channel MANETs.

Cesana *et al.* [4] suggest inserting information about interference and receiver power levels into CTS packets. In this way, each node can estimate the interference it generates on other ongoing transmissions, and refrain from starting a new transmission if it is likely to corrupt other ongoing transmissions. In contrast, DMAC does not modify RTS/CTS packets, but deffers transmissions long enough to avoid the hidden terminal problem.

Numerous efforts were reported for improving TCP behavior over wireless networks [1, 8, 11]. Those techniques are, however, complementary to our work and can be applied to TCP protocols over DMAC as well.

6. Conclusions and Future Work

In this paper, we first showed an instance of the hidden terminal problem, where packets of the same flow interfere with each other while they traverse multiple hops from a source to a destination. We then described a novel MAC protocol that alleviates this problem by deferring further transmissions until the previously transmitted packets advance far enough to avoid interference with newly transmitted packets. Finally, we presented an algorithm that calculates the deferral interval needed to avoid the problem.

In simple chain topologies, DMAC improves the throughput of CBR and TCP flows by up to 100% and 60% respectively. Moreover, in both static and mobile random topologies with up to 40 simultaneous flows, DMAC achieves up to 30% improvement over the traditional MAC in terms of the overall network throughput. While DMAC employs only a simple localized optimization, it is interesting that it works in practice even with multiple flows.

In the future, we plan to investigate the effects of our scheme on the latency of the flows. We also plan to investigate how DMAC affects other schemes that improve TCP performance in MANETs. Finally, we plan to take energy of the received signals into consideration when deciding whether two signals interfere with each other at the receiver.

References

[1] H. Balakrishnan, V. Padmanabhan, S. Seshan, and R. H. Katz. A comparison of mechanisms for improving tcp performance over wireless links. *IEEE/ACM Transactions on Networking*, December 1997.

[2] Alan A. Bertossi and Maurizio A. Bonuccelli. Code assignment for hidden terminal interference avoidance in multihop packet radio networks. *IEEE/ACM Transactions on Networking*, August 1995.

[3] Josh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu, and Jorjeta Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *Proc. of MobiCom*, 1998.

[4] Matteo Cesana, Daniela Maniezzo, Pierpaolo Bergarno, and Mario Gerla. Interference aware (ia) macl an enchancement to ieee802.11b dcf. In *Proc. of IEEE VTC*, 2003.

[5] Samir Ranjan Das, Charles E. Perkins, and Elizabeth E. Royer. Performance comparison of two on-demand routing protocols for ad hoc networks. In *Proc. of INFOCOM*, pages 3–12, 2000.

[6] Kevin Fall and Kannan Varadhan. ns notes and documentation. The VINT Project, UCBerkeley, LBNL, USC/ISI, and Xerox PARC, November 1997. Available at http://citeseer.nj.nec.com/fall00ns.html.

[7] Chane L. Fullmer and J.J. Garcia-Luna-Aceves. Solutions to hidden terminal problems in wireless networks. In *Proc. of SIGCOM*, 1997.

[8] Gavin Holland and Nitin Vaidya. Analysis of TCP performance over mobile ad hoc networks. In *Proc. of MobiCom*, 1999.

[9] IEEE. "IEEE std 802.11 - wireless LAN medium access control (MAC) and physical layer (PHY) specifications", 1997.

[10] David B Johnson and David A Maltz. Dynamic source routing in ad hoc wireless networks. In Tomasz Imielinski and Hank Korth, editors, *Mobile Computing*, volume 353, chapter 5, pages 153–181. Kluwer Academic Publishers, 1996.

[11] S. Kopparty, S.V. Krishnamurthy, M. Faloutsos, and S.K. Tripathi. Split tcp for mobile ad hoc networks. In *Proc. of GLOBECOM*, 2002.

[12] Jinyang Li, Charles Blake, Douglas S. J. De Couto, Hu Imm Lee, and Robert Morris. Capacity of ad hoc wireless networks. In *Proc. of MobiCom*, pages 61–69, 2001.

[13] The CMU Monarch project. wireless and mobility extensions to ns-2, October 1999.

[14] F. A. Tobagi and L. Kleinrock. Packet switching in radio channels: Part 1 - carrier sense multiple-access modes and their throughput-delay characteristics. *IEEE Transactions on Communication*, December 1975.

[15] C. Wu and V. O. K. Li. Receiver-initiated busy-tone multiple access in packet radio networks. In *Proc. of SIGCOM Workshop: Frontiers in Computer Communications Technology*, 1987.

[16] Shiang-Rung Ye, You-Chiun Wang, and Yu-Chee Tseng. A jamming-based mac protocol to improve the performance of wireless multihop ad hoc networks. In *Proc. of IEEE VTC*, 2003.