Low Delay Marking for TCP in Wireless Ad Hoc Networks

Choong-Soo Lee, Mingzhe Li, Emmanuel Agu, Mark Claypool, Robert Kinicki
{clee01,lmz,emmanuel,claypool,rek}@cs.wpi.edu
Computer Science Department at Worcester Polytechnic Institute
Worcester, MA, 01609, USA

Abstract

End-hosts on wireless ad hoc networks typically use TCP as their transport layer protocol. Being designed for wired networks, TCP can perform poorly over wireless networks. Research that has proposed ways to improve TCP performance over wireless networks has concentrated primarily on improving TCP throughput. However, emerging applications, such as interactive multimedia and network games, require reduced delay at least as much as increased throughput. This paper presents LDM[1], an IP layer queue marking mechanism that uses estimates of the number of hops and flows at each wireless node to approximate the optimal marking probability. Analysis of NS-2 simulation results indicates that LDM greatly reduces the round-trip time of TCP connections while improving throughput under many configurations.

1 Introduction

Wireless ad hoc networks currently carry traffic using the Transmission Control Protocol (TCP). However, TCP was designed for wired networks and thus can perform poorly in ad hoc wireless environments including IEEE 802.11 networks [1].

The Media Access Control (MAC) layer of IEEE 802.11 wireless ad hoc networks uses the Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) with a Request-to-Send/Clear-to-Send (RTS/CTS) mechanism to avoid data packet collisions. The RTS/CTS pre-exchange helps mitigate the hidden terminal effect that arises because wireless nodes have the transmission range less than the interference range. A transmission can interfere with another transmission because the latter is outside of its transmission range but within its interference range. The RTS/CTS pre-exchange greatly reduces data packet collisions due to the hidden terminal problem but also causes some side effects when the MAC layer becomes over-saturated.

[1]LDM stands Low Delay Marking

The primary reasons for TCP performance degradation are the contention delays and contention drops that the RTS/CTS mechanism causes, which have been identified as RTS/CTS jamming [2] and RTS/CTS-induced congestion [3].

Previous research on the improvement of TCP performance over wireless ad hoc networks includes the investigation of link breakage and routing failure related problems [4, 5, 6], link layer solutions [7, 8], MAC layer solutions [9], and TCP protocol modifications [10]. A few recent papers present techniques to improve TCP throughput by controlling the total number of packets in flight. Fu *et al.* [8] present a link layer approach, Link-RED (LRED), that limits the TCP sending window to reduce MAC layer collisions, and Adaptive Pacing (AP), which adds a random delay when sending packets to reduce the probability of MAC layer collisions. Chen *et al.* [9] attempt a similar improvement by directly limiting TCP's window size.

Most proposed improvements to TCP are link layer optimizations which are difficult to deploy since they are tied to network card-specific device drivers rather than the more general operating system. Furthermore, throughput has been the most common measure of improvement. However, emerging applications such as streaming multimedia and network games, demand lower round-trip times. Moreover, with the steady increase in maximum wireless network bandwidth (currently up to 54 Mbps for the 802.11g standard), end-to-end delays will become increasingly important relative to throughput.

This paper presents Low Delay Marking (LDM) a technique to modify the IP layer packet queue manager. The goal is to improve round-trip times loss rates and collisions for wireless ad hoc networks, with minimal impact on throughput. LDM is intended to facilitate easy deployment since operating system upgrades can be done independently of hardware changes in the wireless network devices.

The rest of this paper is organized as follows: Section 2 reviews background literature on the hidden terminal problem, LRED and AP; Section 3 focuses on the LDM mechanisms; Section 4 describes the simulation setup and analyzes the simulation results; and Section 5 summarizes our findings and mentions some possible future work.

2 Background

This section briefly introduces background relevant to this investigation, including TCP with Explicit Congestion Notification (ECN) and the Link RED and Adaptive Pacing algorithms for dealing with wireless MAC layer retransmissions.

2.1 Explicit Congestion Notification

Explicit Congestion Notification (ECN) [11] allows routers to mark packets instead of dropping to indicate congestion. The key advantage of marking is that the TCP source receives the explicit congestion indicator much sooner than when packets are dropped. The critical point for this research is that for ad hoc networks with a small diameter (about 15 hops or fewer), the window size of a TCP flow needs to be small for optimal performance [10, 8]. With small window sizes, an IP router that drops a packet from a TCP flow will force a timeout since the sender can not get three duplicate acknowledgments. With these same window sizes, an IP router that marks a packet from a TCP flow allows the TCP source to continue transmitting at a reduced rate since three duplicate acknowledgments are not required. We assume that all future TCP sources will be ECN enabled.

2.2 Link RED and Adaptive Pacing

Random Early Detection (RED) [12] is an Active Queue Management (AQM) scheme that uses the average queue length to determine the dropping or marking probability of packets in the queue. LRED [8] is a data link layer strategy based on RED that keys on the average number of 802.11 retries instead of queue length. Analogous to RED, LRED uses parameters such as min_{th}, max_{th} and max_p to compute the drop probability. LRED can achieve the optimal window size desired by TCP flows on wireless LANs for some configurations, but it shares RED's tuning weaknesses, noted in [13, 14, 15]. Moreover, the fact that LRED drops packets makes it difficult to configure when TCP windows are small and marking at the IP layer based on MAC layer data poses possible network layer violations.

Along with LRED, [8] presents Adaptive Pacing (AP) which is activated by LRED when the average number of retries is less than min_{th} and deactivated when the average number of retries exceeds min_{th}. AP increases MAC layer backoff intervals by the retransmission time of one data frame every time an ACK frame is received. Our analysis in [16] indicates that most of the throughput improvements from LRED coupled with AP are due to AP and not LRED. Unfortunately, the downside of AP is that the additional backoff time between transmissions increases the round-trip times.

3 LDM Mechanism

This section presents the Low Delay Marking (LDM) algorithm which is run at each node on a multihop ad hoc wireless network as illustrated in Figure 2. Each node counts the number of flows traveling through it, as explained in Section 3.3, and maintains per-flow state information on the number of hops per flow, as described in Section 3.2. For each arriving packet, the node computes the optimal window size for the flow, as described in Section 3.1, and marks the packet with the marking probability required to meet this window size, as described in Section 3.4. Figure 1 summarizes the LDM algorithm. In the algorithm, f_i is the i-th flow; h_i is the number of wireless hops f_i makes in going from source to destination; p_{mark} is the marking probability calculated by the IP packet queue management; n is the total number of flows going through the node; w_{opt} is the optimal window size for f_i; and p is the packet that arrived at the node.

```
at each node, on receiving packet p
    identify flow f_i to which p belongs
    estimate h_i for f_i
    estimate n
    calculate w_opt
    calculate p_mark
    mark p with probability p_mark
```

Figure 1: The LDM Algorithm

3.1 Optimal Window Size of a TCP Flow

[10] and [8] derive expressions for the optimal TCP window size as a function of the number of hops between the source and destination nodes in a multihop wireless network. Summarizing these results, a TCP flow achieves maximum throughput when its window size is about one-fourth of the number of hops in a wireless network chain. This restricted window size limits the number of packets in the network, thereby reducing MAC layer congestion (RTS/CTS collisions). However, in determining this optimal TCP window size, neither [10] nor [8] take into account the number of flows. Intuitively, the *aggregate* window size among all flows should be one-fourth of the number of hops

(h). Thus, each flow should have a window size of one-fourth of the number of hops divided by the number of flows (n):

$$w_{opt} = \frac{\frac{h}{4}}{n} \tag{1}$$

3.2 Number of Hops for a Flow

To estimate the number of hops from the source to a destination for a flow, each node keeps per-flow state information, where a flow is identified by an IP source-destination pair. For each active flow, a node records the average time-to-live (TTL) values in the data packets it routes. It also observes destination-source acknowledgment packets for the same flow and records their average TTL value. Since the default TTL values set by modern operating system are typically 128 or 256, each node can compute the number of hops from the node to the source and the number of hops from the node to the destination, thus determining the total number of hops for each flow from source to destination. For example, if a node observes a data packet with a TTL value of 250 and then a corresponding acknowledgment packet with a TTL value of 251, it can compute the number of hops for that flow (h_i) as $(256 - 250) + (256 - 251) = 11$.

3.3 Number of Flows at a Node

Based on Morris' calculations[17], the number of flows at a node can be counted using a fixed-length bit vector v. When a packet arrives, it is hashed based on source-destination address and port number and the corresponding bit in v is set. The count of bits in v is an approximation of the number of active flows. The bits in v are cleared at a rate so as to reset every bit in v every few seconds. When a bit is cleared, the corresponding per-flow state information kept (for example, number of hops for the flow) is also cleared. This method of tracking flows is very accurate when the number of bits in v is significantly larger than the number of flows and does not require any explicit modification of TCP.

3.4 Marking Probability

TCP performance models under congestion marking come from work in [18] and [19], with more detailed performance models in [20] and [21]. Based on results from pilot studies (see [16] for full details), we use the relationship between marking rate (p) and window size (w) derived in [17]:

$$p = \frac{0.76}{w^2} \tag{2}$$

From algorithms described in the previous sections and the state information kept on each active TCP flow, an LDM node calculates the optimal window size for each TCP flow and, using Equation 2, the appropriate marking probability to achieve that window size:

$$p_{mark} = \frac{0.76}{\left(\frac{h}{4 \times n}\right)^2} = \frac{12.16 \times n^2}{h^2} \tag{3}$$

However, a w_{opt} of 1 results in a marking marking probability of 0.76 which, even with packet marking, causes timeouts. Therefore, if w_{opt} is calculated to be 1 or less, an optimal window size of 2 is used for w_{opt}.

Equation 3 represents the overall marking probability that needs to be applied to each flow. We propose that each ad hoc node contributes to this total equally, although alternate policies where the first node in a route applies the full marking probability are also possible. Since a packet has to go through $h - 1$ nodes from source to destination, LDM distributes the probability evenly over $h-1$ nodes. Other distributions are possible. Let p_{node} be the per-node marking probability. We can relate p_{node} to p_{mark} by:

$$p_{mark} = 1 - (1 - p_{node})^{(h-1)}$$

$$p_{node} = \left(1 - \frac{12.16 \times n^2}{h^2}\right)^{\frac{1}{h-1}} \tag{4}$$

Thus, the overall marking probability, p_{mark} is the same as the probability of the packet not being marked through all $h-1$ nodes with probability of p_{node}. Using Equation 4, each node calculates the per-node marking probability for all incoming packets.

For evaluation purposes, the mechanisms described in Section 3.2 and Section 3.3 have been hard-coded into the simulation code used to evaluate LDM, with implementation and evaluation of the per-flow record keeping being future work.

4 Evaluation

This section discusses the simulation setup and analyzes the experimental results. Experiments presented include default TCP performance, TCP performance with window restrictions, TCP performance with adaptive pacing, and TCP performance with the LDM algorithm.

4.1 Simulation Setup

To evaluate the effectiveness of LDM, we enhanced the NS-2 simulator [22] to include code for the LDM algorithm as described in Section 3. Due to the unavailability of Adaptive Pacing code from [8], we also had to implement Adaptive Pacing in NS-2 so as to be able to compare it with LDM. The simulated wireless

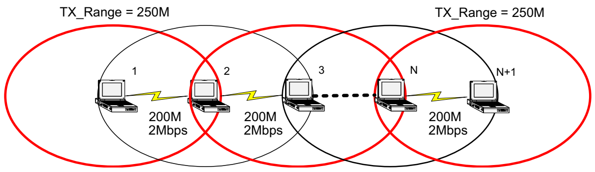


Figure 2: Simulation Topology

ad-hoc network topology used in this investigation is shown in Figure 2. In general, there are $h+1$ wireless nodes, N_0 through N_h, connected over an IEEE 802.11 chain topology. Default IEEE 802.11 layer settings are used with a wireless capacity of 2 Mbps and AODV routing. All flows use TCP NewReno with a maximum window size of 32, except in the window constrained case.

The experiments reported in this paper include: regular TCP, which represents the current practice in ad hoc network performance; TCP which represents the optimal performance by manually constraining the window size of each TCP flow with full network knowledge; Adaptive Pacing where all MAC frames are delayed by an additional amount, as described in Section 2.2, and LDM. Each of these cases was simulated with 7 hop, 15 hop and 24 hop ad hoc chain topologies where all nodes are immobile. Each simulation was run five times, with the graphs depicting the averages and minimum and maximum values shown with error bars. While the graphs report performance in absolute terms for round-trip time, loss fraction, and RTS collision fraction, throughput is normalized to that of regular TCP to help clarify the performance differences.

Two sets of simulations were run: one with a single TCP flow and the other with three TCP flows. Due to space contraints, only the detailed results from the three flow experiment are presented. However, the summary of both sets of experiments are given in Figure 4 and Figure 5 to show that the single flow experiment exhibits similar behavior.

4.2 Multiple Flows

This experiment involves three TCP flows going through a multihop wireless network. Figure 3 depicts the total throughput normalized to that of regular TCP, the total loss fraction, the total number of RTS collisions and the round-trip time of one of the flows.

Over the 7 hop chain, regular TCP achieves 179 Kbps, restrained TCP with a window size of 1 achieve 262 Kbps, Adaptive Pacing improves throughput to 231 Kbps, and LDM achieves 220 Kbps. While Restrained TCP yields a signficantly lower RTT (148 ms) than either TCP (464 ms) or Adaptive Pacing (489 ms), LDM comes close to the restrained case with a round-trip time of 237 ms. Restrained TCP has the lowest loss fraction and regular TCP the highest. Both Adaptive Pacing and LDM have slightly higher loss fractions compared to restrained TCP, but LDM offers a lower loss fraction compared to Adaptive Pacing. The RTS collision fraction for Adaptive Pacing is highest while the RTS collision fraction for restrained TCP is lowest.

Over the 15 hop chain, regular TCP achieve 148 Kbps with the respective throughputs for restrained TCP with window size of 2, Adaptive Pacing and LDM being 213 Kbps, 215 Kbps and 188 Kbps respectively. With 15 hops, the gap between the two best treatments (Restrained TCP and LDM) and the two worse treatments (TCP and Adaptive Pacing) with respect to RTT grows. LDM has the lowest loss fraction and regular TCP the highest. The Adaptive Pacing loss fraction is about half way between these two fractions. The RTS collision fraction decreases by 26.9% for restrained TCP, Adaptive Pacing decreases the RTS collision fraction by 11.4%, and LDM reduces the RTS collision fraction by 26.4%.

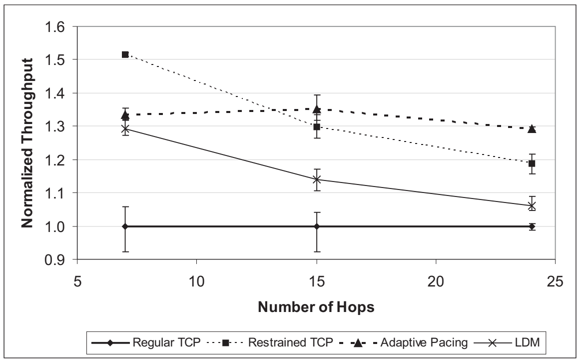
With a 24 hop chain, the throughputs are: TCP 176 Kbps, restrained TCP (window size of 2) 202 Kbps, Adaptive Pacing 227 Kbps, and LDM 186 Kbps. The RTTs for the four treatments all increase over the 15 hop chain and the only relative difference is that LDM is slightly higher than Restrained TCP. Restrained TCP has the lowest loss fraction and regular TCP the highest. Adaptive Pacing has higher loss fraction compared to the restrained TCP while LDM offers a slightly higher loss fraction compared to the restrained TCP, yet a lower loss fraction compared to Adaptive Pacing. At 24 hops, the RTS collision fractions are converging around 0.15 except for regular TCP.

4.3 Summary

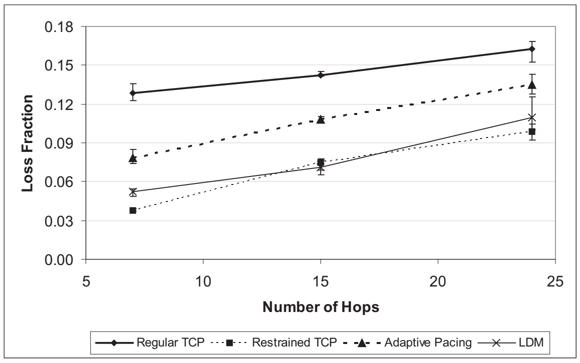
Category	Single Flow			Multiple Flows		
Hops	7	15	24	7	15	24
Throughput	+	0	0	+	+	0
Round-Trip Time	+	+	+	+	+	+
Loss Fraction	+	+	+	+	+	+
RTS Collision Fraction	+	+	+	+	+	+

Figure 4: Performance of LDM compared to Regular TCP

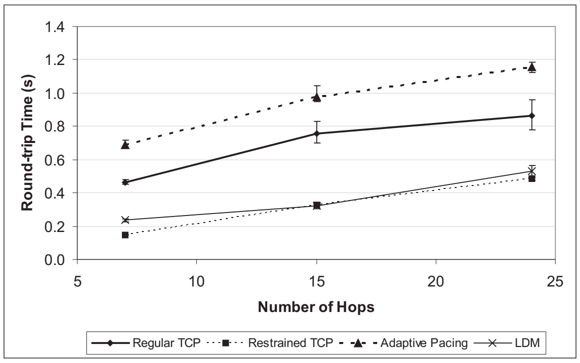
The performance of LDM compared with regular TCP from both the single flow and multi-flow exper-



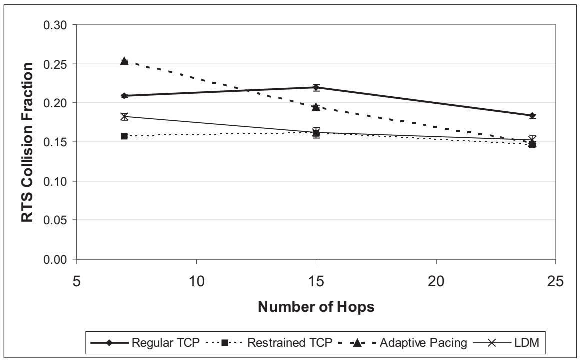
(a) Normalized Throughput



(b) Loss Fraction



(c) Round-trip Time



(d) RTS Collision Fraction

Figure 3: Three Flows over Multihop Chain Topology

iments are summarized in a table in Figure 4. A '+' denotes cases where LDM's performance is better by more than 10%, a '0' where LDM's performance is with within 10%, and a '-' where LDM is worse by more than 10%. From the table, LDM provides about the same or better throughput compared to regular TCP but provides a much lower round-trip time, loss fraction and RTS collision fraction.

The performance of LDM compared with Adaptive Pacing is summarized in Figure 5. LDM provides about the same or less throughput compared to adaptive pacing, but provides greatly reduced round-trip times, loss fractions and RTS collision fractions. These results are especially significant for applications that are sensitive to high delays.

Category	Single Flow			Multiple Flows		
Hops	7	15	24	7	15	24
Throughput	0	–	–	0	–	–
Round-Trip Time	+	+	+	+	+	+
Loss Faction	0	0	0	+	+	+
RTS Collision Fraction	+	0	0	+	+	0

Figure 5: Performance of LDM compared to Adaptive Pacing

5 Conclusion

This paper presents Low Delay Marking (LDM), an IP layer approach to enhance TCP performance towards lower delays and loss rates without sacrificing throughput. Building on knowledge of the optimal TCP window size discussed in [8], LDM marks packets with the probability calculated with the estimated number of hops and the number of flows. This forces TCP flows to reduce their window size closer to an optimal value, thus resulting in a reduced congestion at the MAC layer. Less MAC layer congestion leads to fewer collisions and therefore decreases round-trip times and loss rates for all flows in the network.

We simulated and evaluated LDM over multiple chain topologies with single and multiple-flows. The results show that LDM provides significantly better round-trip times (up to a 57.6% reduction) and loss rates (up to a 59.5% reduction) while still providing the same or better throughput compared to regular TCP. LDM also provides much better round-trip times (up to a 67.2% reduction) and loss rates (up to a 33.8% reduction) compared to Adaptive Pacing.

Currently, our evaluation is done over with the number of hops and number of flows known ahead of time by each router. Implementation of hop and flow counting techniques presented in Section 3 is our current ongoing investigation. Additionally, evaluations with more complex topologies such as crosses and grids is also under investigation.

References

[1] IEEE Computer Society LAN MAN Standard Committee, "IEEE 802.11, 1999 Edition, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications," .

[2] C.Ware, T.Wysocki, and J.F.Chicharo, "On the Hidden Terminal Jamming Problem in IEEE 802.11 Mobile Ad Hoc Networks," in *Proceedings of IEEE ICC'01*, 2001.

[3] Saikat Ray, Jeffrey Carruthers, and David Starobinski, "RTS/CTS-induced Congestion in Ad-Hoc Wireless LANs," in *Proceedings of IEEE WCNC 2003*, Mar. 2003, pp. 1516–1521.

[4] Kartik Chandran, Sudarshan Raghunathan, S. Venkatesan, and Ravi Prakash, "A feedback based scheme for improving TCP performance in adhoc wireless networks," in *Proceedings of ICDCS'98*, 1997, pp. 472–479.

[5] Gavin Holland and Nitin H. Vaidya, "Analysis of TCP performance over mobile ad hoc networks," in *Proceedings of IEEE/ACM MOBICOM '99*, August 1999, pp. 219–230.

[6] Venkatesh Ramarathinam and Miguel A. Labrador, "Performance analysis of tcp over static ad hoc wireless networks," in *Proceedings of the ISCA 15th PDCS*, September 2002, pp. 410–415.

[7] M. Gerla, K. Tang, and R. Bagrodia, "Tcp performance in wireless multihop networks," in *Proceedings of IEEE WMCSA'99*, Feb. 1999.

[8] Zhenghua Fu, Petros Zerfos, Haiyun Luo, Songwu Lu, Lixia Zhang, and Mario Gerla, "The Impact of Multihop Wireless Channel on TCP Throughput and Loss ," in *Proceedings of IEEE Infocom Conference*, Mar 2003.

[9] Frederico Cali, Marco Conti, and Enrico Gregori, "Dynamic tuning of the ieee 802.11 protocol to achieve a theoretical throughput limit," *IEEE/ACM Trans. Netw.*, vol. 8, no. 6, pp. 785–799, 2000.

[10] K. Chen, Y. Xue, and K. Nahrstedt, "On Setting TCP's Congestion Window Limit in Mobile Ad Hoc Networks," in *ICC*, 2003.

[11] Sally Floyd, "TCP and Explicit Congestion Notification," *Computer Communication Review*, Oct. 1994.

[12] S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," *IEEE/ACM Transactions on Networking*, Aug. 1993.

[13] M. Christiansen, K. Jeffay, D. Ott, and F.D. Smith, "Tuning RED for Web Traffic," in *Proceedings of ACM SIGCOMM Conference*, Aug. 2000.

[14] G. Iannaccone, M. May, and C. Diot, "Aggregate Traffic Performance with Active Queue Management and Drop from Tail," *ACM Computer Communication Review*, July 2001.

[15] W. Feng, D. Kandlur, D. Saha, and K. Shin, "A Self-Configuring RED Gateway," in *Proceedings of IEEE Infocom*, Mar. 1999.

[16] Choong-Soo, Emmanuel Agu, Mark Claypool, and Robert Kinicki, "Low Delay Marking for TCP in Wireless Ad Hoc Networks," Tech. Rep. WPI-CS-TR-03-34, CS Department, Worcester Polytechnic Institute, Dec. 2003.

[17] Robert Morris, "Scalable TCP Congestion Control," in *Proceedings of IEEE Infocom*, Mar. 2000.

[18] S. Floyd, "Connections with multiple congested gateways in packet-switched networks part 1: One-way traffic," *ACM Computer Communication Reviews*, pp. 30 – 47, Oct. 1991.

[19] Matthew Mathis, Jeffrey Semke, Jamshid Madhavi, and Teunis Ott, "The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm," *ACM Computer Communication Review*, vol. 21, no. 5, Oct 1991.

[20] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP Throughput: A Simple Model and Its Empirical Validation," in *Proceedings of ACM SIGCOMM*, 1998.

[21] Neal Cardwell, Stefan Savage, and Tom Anderson, "Modeling the Performance of SHort TCP Connections," Tech. Rep., University of Washington, 1989.

[22] Universiy of California Berkeley, "The Network Simulator - ns-2," Online at http://www.isi.edu/nsnam/ns/.