Investigation of the TCP Simultaneous-Send Problem in 802.11 Wireless Local Area Networks

Sumathi Gopal
WINLAB, Rutgers University
Piscataway, NJ 08854-8058
sumathi@winlab.rutgers.edu

Sanjoy Paul
Bell Labs, Lucent Technologies
Holmdel, NJ 07733-3030
sanjoy@lucent.com

Dipankar Raychaudhuri
WINLAB, Rutgers University
Piscataway, NJ 08854-8058
ray@winlab.rutgers.edu

***Abstract*—This paper investigates the TCP *simultaneous-send* problem which arises in infrastructure mode 802.11 wireless local area networks. In particular it has been observed that for file transfer traffic, 802.11 wireless nodes have a sustained supply of packets to send and hence experience a relatively high rate of MAC contention. For TCP, this results in competition among data and ACK packets for channel access which causes the simultaneous-send problem that deteriorates flow throughput. This simultaneous-send problem can be alleviated by skipping TCP ACKs. Detailed simulation results are presented to demonstrate the usefulness of ACK skipping in various network scenarios such as with MAC retries and multiple TCP flows. The largest improvement is seen for the case of a single TCP flow, and moderate gains are also achieved in cases with multiple streams. For the single TCP stream case with 1 ACK skip and no MAC retries, TCP throughput improves 30% for short-lived and 98% for long-lived TCP transmissions. The paper concludes with potential cross-layer solutions that potentially provide further improvements, including the use of the point coordination function (PCF) to reduce contention between multiple TCP streams and returning ACK packets.**

Keywords— *TCP, 802.11 MAC, DCF, skipped ACKs, delayed ACKs, simultaneous-send*

I. INTRODUCTION

802.11 wireless networks are now widely deployed in offices, homes and hotspots, supporting channel rates up to 54 Mbps. They operate in the Infrastructure mode where packets between any pair of nodes are relayed through an access point (AP). The Distributed Coordination Function (DCF) mode of MAC access is typically used in infrastructure networks, where all nodes including the AP have the same priority for channel access. Majority of the traffic in today's networks constitute TCP flows employed by applications including email, file transfer, web browsing and database access that require reliable end-to-end transport.

We evaluate TCP throughput in these networks under a variety of traffic scenarios such as single/multiple flows, with/without MAC retries, and short-lived/long-lived transmissions. It is shown in Section II that the bursty nature of TCP traffic causes the 802.11 MAC to often have a continuous supply of packets to send. This increases the likelihood of two or more nodes selecting the same backoff slot. For these reasons TCP data and ACK packets compete for channel access and cause MAC transmission failures. The problem is particularly significant with a single TCP flow. With reference to the network in Figure 1, the AP and TCP source node often transmit to each other in the same backoff slot. Neither node detects the packets, since the hardware implementation prevents them from sending and listening at the same time. This is the *simultaneous-send* problem. With no channel errors, disabled MAC retries and a single TCP flow*, simultaneous-send* is the sole cause of packet losses. For multiple nodes, the same-slot selection phenomenon also manifests as collisions where two or more nodes transmit in the same MAC backoff slot, and a third listener hears a combined garbed signal. If MAC retries are enabled, *simultaneous-send* problem is not experienced at the transport layer. Instead, packet losses now occur due to MAC queue overflow from TCP bursts.

This paper demonstrates how these problems can be alleviated at the transport layer, by *skipping TCP ACKs.* For no MAC retries, it reduces the number of ACK segments competing for channel access, and hence reduces MAC contention. When MAC retries are enabled, ACK skipping controls congestion window growth while in slow start, reducing packet bursts and minimizing MAC queue overflows. However, skipping too many consecutive ACKs has a detrimental effect on TCP throughput as less frequent cumulative ACKs may curtail the growth of the TCP congestion window. This tradeoff is investigated in the rest of this paper, for the various traffic conditions mentioned earlier.

Majority of the existing research on TCP over 802.11 focus on multihop ad-hoc scenarios. Few are modeling efforts[1],[10], while most examine effects of different parameters on TCP *goodput*. The latter include issues such as fairness/channel capturing[4],[5], instability problem with two or more simultaneous flows[12], performance variation with TCP packet sizes[5], upper limit on the congestion and receiver windows[3]-[7], effect of exposed/hidden nodes[6] and effect of different MAC retry limits[5].

[10], [11] and[13] and are most relevant. In [10], Kherani and Shorey analytically show improved TCP throughput with 1 ACK skip for a single TCP flow, in lieu of a geometric distribution for backoff slot selection in 802.11 MAC. Altman et. al[11] simulate TCP skipped ACKs in an ad-hoc multihop scenario for a linear topology with a single TCP flow and restricted congestion/receiver window sizes. With MAC retries, they conclude that delaying any number of ACKs is better than not delaying at all. Wu et. al[13] observe TCP data-ACK competition for channel access in ad-hoc multihop networks and propose a DCF-MAC extension called DCF+ to

accommodate TCP ACKs. Results in this paper differ from the above particularly due to the nature of the network considered and the assumptions made.

The rest of this paper is organized as follows. Section II presents brief overviews of protocols and describes the TCP throughput problem in detail. Section III describes the set up of simulations. Results and their analysis are presented in Section IV. Finally, we conclude in Section V with directions for future work.

II. Problem description with relevant overview of protocols

A. 802.11 DCF-MAC overview

Distributed Coordination Function (DCF) mode of 802.11 Medium Access Control (MAC) uses Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA)[15]. Even in the infrastructure mode with a central Access Point (AP), all entities have equal priority for channel access. Channel contention happens on a per-packet basis. A node waiting to transmit a packet, first senses the channel to be idle for a certain duration (called DIFS [15]), then selects a backoff slot randomly based on a uniform distribution in [0,CW-1], where CW is the contention window. The packet is transmitted if the channel is still idle in the selected slot. If not, the node waits till the channel is idle again, backs off only for the requisite slots before attempting to transmit. The node learns of a successful transmission when it receives an acknowledgement (MAC-ACK) from the destination.

Throughput derivations of 802.11 MAC have typically assumed Poisson packet arrival per backoff slot [1]. Instead, suppose there are (N+1) nodes with a continuous supply of packets that causes them to contend for the channel far more consistently. A transmission is successful only if no other node transmits in the same backoff slot. This likelihood of all nodes selecting independent slots is

$$1*\left(\frac{CW-1}{CW}\right)\left(\frac{CW-2}{CW}\right)\cdots\left(\frac{CW-N}{CW}\right) = \left(\frac{(CW-1)!}{(CW-N-1)!*(CW)^N}\right)$$

Hence the likelihood of at least two nodes selecting the same slot is

$$1-\left(\frac{(CW-1)!}{(CW-N-1)!*(CW)^N}\right) \quad (1)$$

This is the likelihood of a failed transmission for nodes having a consistent supply of packets to send.

B. TCP overview

TCP is a reliable sliding window protocol. It allows several packets to be transmitted before an acknowledgement (TCP ACK) is received. Unacknowledged data packets are maintained in a congestion window (*cwnd)*. In steady state, *cwnd* is expected to be equal to the delay-bandwidth product of the network. TCP ACKs are cumulative. Hence it is not necessary to have separate ACKs for each packet. TCP operates in either of slow-start or congestion-avoidance modes. In slow-start, *cwnd* grows exponentially, increasing by one packet for every ACK received, while in congestion avoidance, *cwnd* increases linearly (one packet per round-trip-time) in proportion to the number of data segments acknowledged.

In case of a packet loss TCP reduces *cwnd* assuming network congestion. With TCP Reno, three or more duplicate ACKs trigger a *fast-retransmit* and cwnd drops to ½ its previous value. Then, *cwnd* grows conservatively in congestion avoidance mode. The situation is much worse in case of a *timeout,* when *cwnd* drops to 1. Then, TCP backs off exponentially before trying to send the next first packet. The backoff duration can be as high as 64 seconds. Venkatraman et. al[14] explain the derogatory effect of timeouts on TCP throughput.

C. TCP dynamics over 802.11 DCF MAC

Now we explain why TCP traffic causes nodes to have a consistent supply of packets to send. TCP generates data in bursts due to the sudden increase in *cwnd* resulting from cumulative ACKs. With an ethernet link, these packets are transmitted immediately. On the contrary, 802.11 DCF-MAC transmits packets sequentially after channel contention for each packet. The MAC may make several attempts before the TCP packet is successfully sent or eventually giving up. Hence the new burst of TCP packets could remain in the MAC queue, for a long time before being transmitted, causing a sustained occupation in the send queue.

The condition can be explained similarly at the Access Point (AP). For the traffic flow depicted in Figure 1, AP contends for the channel only to relay TCP ACKs to respective wireless TCP data sources. We assume the default TCP sink implementation where 1 TCP-ACK is generated for every data packet. Hence for a single TCP flow in the network, the AP contends almost as often as the TCP source node and the AP MAC queue occupancy is similar to that at the TCP source node. Following the same argument for N TCP flows, the AP will have N times more queue occupancy than any of the other wireless nodes, resulting in a consistent supply of packets to MAC layer in the AP.

When there is a single TCP flow, the likelihood of same slot selection is simply (1/CW)*(1/CW)*CW. If the previous transmission was not a failure, CW=CWmin=32 and the likelihood is 3%. Otherwise, CW doubles (binary random backoff) and that likelihood halves to 1.5%. For N=3 (3 TCP flows), and CW=32, likelihood of same slot selection, and hence MAC failures is 17.6% from Equation (1).

With these observations, we differ from Kamerman and Aben[1] in their conclusion that with TCP traffic, the likelihood of two nodes selecting the same slot is miniscule. This assertion is corroborated by results from simulations.

D. Alleviating the problem with TCP ACK skipping

Observations made in earlier subsections motivate traffic reduction in the 802.11 wireless LAN. It is straight-forward to

do so by *skipping TCP ACKs* at the TCP sink. Just skipping alternate ACKs reduces traffic load at the AP by a factor of two and the AP contends half as much for channel access. This reduces interference with data packets, hence improving TCP throughput. With MAC retries enabled, ACK skipping regulates *cwnd* growth in slow start and hence reduces loss of packets from MAC queue overflows. However, TCP throughput is directly related to the growth of TCP congestion window, which in turn depends on regular arrival of TCP ACKs. Thus skipping too many ACKs impedes *cwnd* growth and curtails TCP throughput. Experiments and results explained in Section IV corroborate this insight.

III. Experimental setup

All experiments are simulations in NS version 2.1b9a with minor modifications in TCP code. Figure 1 depicts the network under consideration. Each node caters to a single TCP flow. DSDV is used as wireless routing protocol although there is no need for one in the infrastructure mode. Experiments were conducted with various link delays ranging from 2ms to 15ms.

Channel data rate of 11Mbps and channel basic rate of 1Mbps are selected along with the two ray ground channel model. All wireless nodes are within hearing range of each other.

MAC queue length in wireless nodes (including AP) are

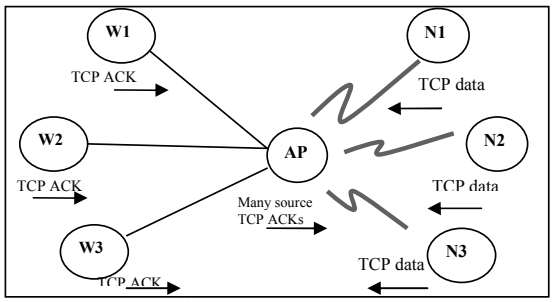

Figure 1: Network setup

retained at the default NS value of 50. The effect of this is explained in Section IV.A. 802.11 MAC standard suggests RTS/CTS disabled (dot11RTSThreshold = 3000; while Maximum MAC MTU = 2304). Our experiments with RTS/CTS in this network, showed reduced TCP throughput from additional overhead per packet. Hence RTS/CTS is disabled in all simulations. For experiments with MAC retries, the default values suggested in the 802.11 std.[15] are used (dot11ShortRetryLimit=7; dot11LongRetryLimit=4).

TCP receiver advertised window is chosen large to ensure that instantaneous TCP throughput is only paced by the TCP congestion window. To allow the congestion window to stabilize, ACK skipping is set to start only after TCP sequence number crosses a threshold (chosen as 50 here by trial and error). Long-lived (short-lived) TCP connections are set up with 10MB (1MB) file transfers using the File Transfer Protocol (FTP). Each TCP throughput value published here is an average of 5 runs, obtained by varying FTP start times. TCP Reno is used in all experiments with the duplicate ACK parameter set to 3.

Maximum duration of simulations was 1000 seconds. Some experiments, particularly with multiple skipped ACKs did not complete file transfers in this duration. In these cases, TCP *goodput* (net data transferred successfully in 1000s) was substituted for throughput.

IV. Results and analysis

Figures 2-6 (and more in the Appendix) present results from various simulations. Skipping TCP ACKs is found to indeed improve TCP throughput for most scenarios, corroborating our hypothesis. All scenarios gain from skipping at least 1 ACK. Figure 2 compares TCP throughput of long-lived TCP connections in the absence of MAC retries for different link delays with increasing ACK skips. This and other figures provided in the appendix show that there isn't a significant change in results with different link delays. Hence it suffices to explain results obtained with any one of them. All subsequent graphs depict results with 2ms link delay.

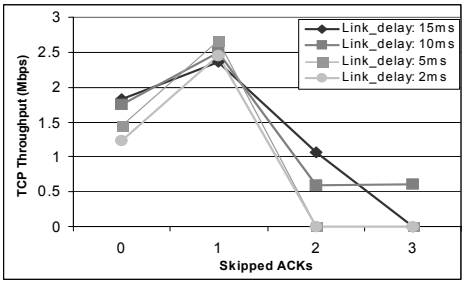

Figure 2: Variation of TCP throughput with number of skipped ACKs ; link delay is a parameter;
Single TCP flow; Long-lived TCP connection; NO MAC retries

Figures 3-6 present the main results. The significant gains achieved with a single TCP flow, are explained in Section IV.A., followed by the explanation for multiple flows. Long-lived TCP connections test the stability of the protocol adaptation and confirm its validity, while short-lived TCP connections supply a transient view of operation. Both lengths of connections are important for different applications and results with respect to them are explained in Section IV.C. Situations with and without MAC retries are compared in each of the subsections.

A. Case with a Single TCP flow

With no traffic in the network other than the single TCP flow, the AP contends for the channel to relay TCP ACK packets, while the wireless TCP source node contends to send TCP data packets. With no MAC retries, the results show meager throughputs of around 1Mbps with both short-lived and long lived TCP connections. This is because of the 3% likelihood of *simultaneous-send* that causes MAC failures, explained in Section II.C. Figures 3 & 4 show significant throughput improvement with 1 ACK skip. It improves by up to 30% for short-lived (1MB FTP) and up to 98% for long-lived (10MB) TCP connections. This is because the reduced MAC traffic simply cuts MAC failures by half! For higher ACK skips, many sessions failed to complete. This can be explained from a combination of two effects – TCP congestion

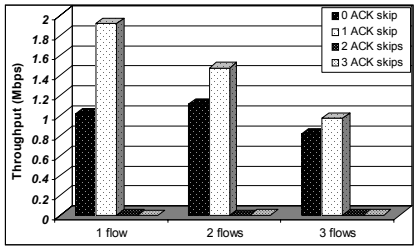

Figure 3: Short-lived TCP flow with NO MAC retries

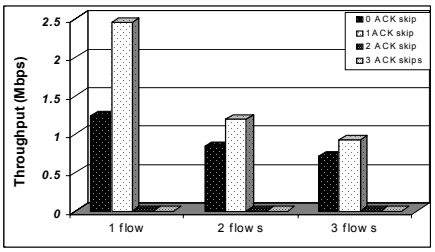

Figure 4: Long-lived TCP flow with NO MAC retries

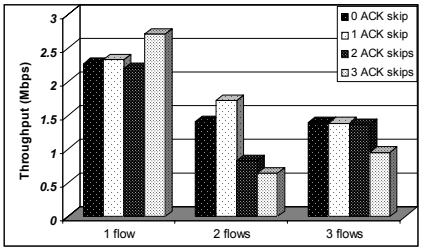

Figure 5: Short-lived TCP flow with MAC retries

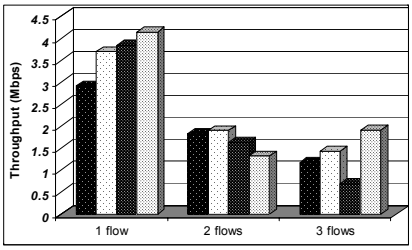

Figure 6: Long-lived TCP flow with MAC retries

window starving from lack of regular TCP ACKs as well as the loss of cumulative ACKs due to MAC failures. Multiple timeouts severely reduce TCP throughput because of TCP's exponential backoff.

In the case with MAC retries, NS trace files confirmed that TCP throughput degradation resulted from overflow of MAC interface queues, Here the *simultaneous-send* problem is handled by the MAC itself by multiple retransmissions.

Results depicted in figures 5 & 6 show consistent throughput gain with ACK skipping. Curiously, the reason behind this is TCP dynamics itself. The default operation in NS, does not restrict congestion window growth to a maximum slow start threshold (max_ssthresh). Thus *cwnd* remains in slow start until there is a packet loss, after which exponential backoff and congestion avoidance set in. With no ACK skipping, large bursts of data packets arriving at the MAC cause queue overflows. When ACKs are skipped, *cwnd* experiences slower growth because of the reduced number of incoming ACK segments. Hence TCP sends smaller bursts of packets, reducing or even eliminating MAC queue overflows. This is why we see upto 30% gain in TCP throughput with ACK skipping.

B. *Situation with multiple TCP flows*

Contention among multiple wireless nodes causes collisions in addition to simultaneous-send. Simulations showed that the most common cause of packet loss is a combination of both problems. Referring Figure 1, Node N1 transmits a TCP data packet to the AP at the same time when the AP transmits a packet to node N2. N2 sees a garbled signal due to simultaneous signals from N1 and the AP (collision), while the AP fails to detect N1's transmission (*simultaneous-send*). Thus the AP is found to contend for channel access far more often than other nodes. This corroborates our hypothesis that the AP will have many more packets to transmit than any one of the other wireless nodes. With MAC retries, ACK skipping results in throughput gains for the same reasons explained in the case of a single TCP flow.

C. *Short-lived and Long-lived TCP connections*

Comparing Figures 3 with Figure 4 and Figure 5 with Figure 6, it is clear that the pattern of throughput improvement is very similar for short-lived and long-lived TCP connections. They differ only in the extent of their gains. It may be inferred that longer the TCP connection, more will be the gain from ACK skipping. TCP congestion window achieves steady-state in the long run, when its size is close to the delay-bandwidth product of the network between source and destination. For the network setup here, it would be the product of the average RTT and the net bandwidth available. However, NS trace files show that with packets lost due to MAC contention and queue overflows, the congestion window seldom reaches this steady state.

V. Conclusion and Directions for Future Work

We end this paper with the conclusion that MAC transmission failures caused by competing TCP data and ACK packets, is a significant cause of TCP throughput degradation in 802.11 DCF infrastructure networks. We have explored the viability of ACK skipping for various scenarios with simulations and explanation of results. While skipping alternate TCP ACKs (1-ACK skip) does reduce MAC contention and improves TCP throughput, skipping too many has a detrimental effect due to less frequent cumulative ACKs that slow down growth of the TCP congestion window.

The ACK skipping mechanism presented here may be regarded as a rudimentary form of cross-layer adaptation, where the TCP sink statically adapts to transmission over 802.11, by skipping requisite ACKs. A better adaptation would be to dynamically select the number of ACKs to skip depending on MAC layer feedback on congestion in the wireless link. Additionally if the TCP layer is informed by its underlying MAC layer of a failed transmission, it may adapt proactively and prevent a retransmission timeout. If the packet loss was caused due to physical channel error, TCP may freeze the congestion window from reducing its size, thus conserving overall throughput.

Finally, the MAC contention and transmission failures may be alleviated significantly by migrating to the Point Coordination Function (PCF) mode of 802.11. Here the AP polls nodes for data to send and allocates contention-free slots for transmission. The PCF mechanism can also be used to prevent contention between TCP data packets and returning ACKs.

ACKNOWLEDGEMENTS

We would like to thank Ivan Seskar and Hongbo Liu of WINLAB, Rutgers University, for their valuable insights in the analysis of results.

REFERENCES

[1] Kamerman, A.; Aben, G., "Net throughput with IEEE 802.11 wireless LANs", IEEE WCNC. Sept 2000. Volume 2

[2] R de Oliveira and T. Braun, "TCP in wireless mobile ad hoc networks", Tech. Report IAM-02-003, univ. of Bern, Switzerland, July, 2002.

[3] K. Chen, Y. Xue, and K. Nahrstedt "On Setting TCP's Congestion Window Limit in Mobile Ad Hoc Networks". ICC'03.

[4] Pilosof, S. Ramjee, R., Raz, D. Shavitt, Y. , Sinha, P., "Understanding TCP fairness over wireless LAN", IEEE INFOCOM 2003. 30 March-3 April 2003

[5] Rui Jiang; Gupta, V.; Ravishankar, C.V., "Interactions between TCP and the IEEE 802.11 MAC protocol", Proceedings of DARPA Information Survivability Conference and Exposition, April 2003. Volume: 1

[6] Kanth, K. Ansari, S. Melikri, M.H. , " Performance enhancement of TCP on multihop ad hoc wireless networks", IEEE International Conference on Personal Wireless Communications, Dec 2002

[7] Xu, S.; Saadawi, T., "Revealing and solving the TCP instability problem in 802.11 based multi-hop mobile ad hoc networks", 54th Vehicular Technology Conference, 2001. Volume: 1

[8] Qixiang Pang; Liew, S.C.; Cheng Peng Fu; Wei Wang; Li, V.O.K.; "Performance study of TCP Veno over WLAN and RED router", IEEE GLOBECOM '03. , Volume: 6 Pages:3237 – 3241.

[9] Garcia, M. Choque, J. Sanchez, L. Munoz, L. "An experimental study of Snoop TCP performance over the IEEE 802.11b WLAN", The 5th International Symposium on Wireless Personal Multimedia Communications, Oct 2002.

[10] Kherani, A.A.; Shorey, R.; "Performance improvement of TCP with delayed ACKs in IEEE 802.11 wireless LANs", Wireless Communications and Networking Conference, March 2004, Volume 3

[11] Eitan Altman, Tania Jiménez "Novel Delayed ACK Techniques for Improving TCP Performance in Multihop Wireless Networks" Personal Wireless Communications 03, Venice Italy.

[12] Shugong Xu; Tarek Saadawi; Myung Lee; "On TCP over wireless multi-hop networks", IEEE Military Communications Conference, Oct 2001.

[13] Wu, H.; Peng, Y.; Long, K.; Cheng, S.; Ma, J; "Performance of reliable transport protocol over IEEE 802.11 wireless LAN: analysis and enhancement", INFOCOM 2002.

[14] Sinha P., Venkitaraman N., Sivakumar R. and Bharghavan V., "WTCP: A Reliable Transport Protocol for Wireless Wide-Area Networks", Wireless Networks, March 2002, Volume 8 Issue 2/3.

[15] IEEE 802.11, 1999 Edition (ISO/IEC 8802-11: 1999) Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications

APPENDIX

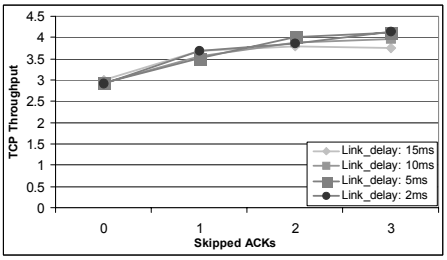

Figure A1: 1 TCP flow; Long-lived TCP connection (10MB) with MAC retries

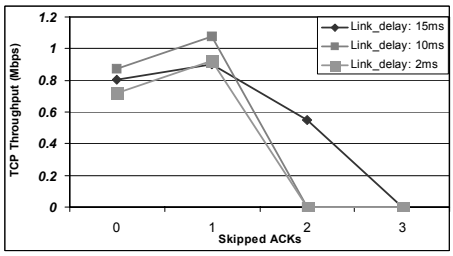

Figure A2: 3 TCP flows; Long-lived TCP connection (10MB); no MAC retries

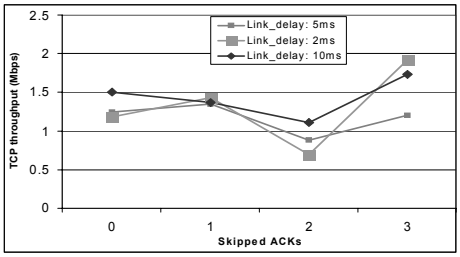

Figure A3: 3 TCP flows; Long-lived TCP connection(10MB) with MAC retries

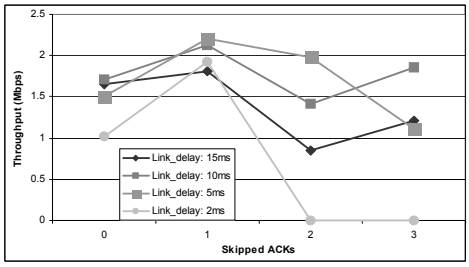

Figure A4: 1 TCP flow; Short-lived TCP connection (1MB) No MAC retries

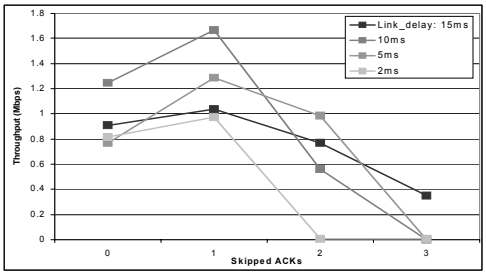

Figure A5: 3 TCP flows; Short-lived TCP connection (1MB) No MAC retries