Enhancing TCP Throughput over Lossy Links Using ECN-capable RED Gateways

Haowei Bai
AES Technology Centers of Excellence
Honeywell Aerospace
3660 Technology Drive, Minneapolis, MN 55418
E-mail: haowei.bai@honeywell.com

Mohammed Atiquzzaman
School of Computer Science
University of Oklahoma, Norman, OK 73019-6151
E-mail: atiq@ou.edu

Abstract— **Explicit Congestion Notification (ECN), when used with Random Early Detection (RED) gateways, reduces packet losses and delays of Transport Control Protocol (TCP) based applications. However, choosing the buffer size and optimum parameter values of RED buffers are still open research issues. In this paper, we first present a model to determine the optimal value of RED's maximum threshold to achieve zero packet loss at RED gateways. Secondly, as an application of our model, we propose a new TCP algorithm, called Differentiation Capable TCP (Diff-C-TCP) to improve the TCP performance over lossy satellite links. Since most of network congestion losses can be eliminated by our zero loss model, Diff-C-TCP assumes packet losses to be indicators of link corruption, and uses ECN to explicitly indicate network congestion. We have shown that our zero packet loss analytical model matches simulation results very well, and our proposed Diff-C-TCP algorithm significantly improve TCP throughput.**

I. Introduction

The Transmission Control Protocol assumes, by default, that every loss event is caused by network congestion, and reacts by reducing the transmission rate of senders. When lossy links, such as, satellite links, are involved in a TCP connection, packet losses due to corruption are more significant than congestion losses. In such a case, TCP congestion control algorithms unnecessarily waste time in slow-start or congestion avoidance procedures triggered by link errors. Consequently, the current congestion control algorithms in TCP result in poor transport layer throughput. Significant performance improvements can be achieved if losses due to network congestion and corruption in lossy satellite links could be appropriately differentiated [1].

ECN [2] has been proposed by IETF to explicitly inform TCP sources of congestion at routers, without requiring them to wait for either a retransmission timer timeout or three duplicate acknowledgements (ACKs). ECN has been recommended to be used in conjunction with RED [3].

RED uses an exponential weighted moving average to calculate average queue size from the instantaneous queue size, and two thresholds (*minimum* and *maximum*) to determine whether an arriving packet should be dropped. When ECN is used with RED (see Figure 1), if the average queue size is between the minimum and the maximum thresholds, the packet is marked with a certain probability as a Congestion Experienced (CE) packet, and sent to the TCP receiver. Upon

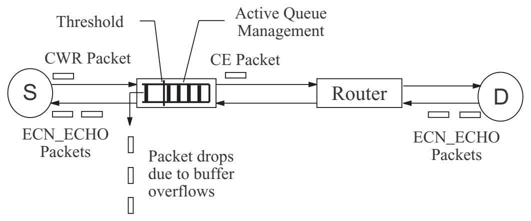

Fig. 1. The ECN mechanism illustration.

receiving the CE packet, the TCP receiver will keep sending ECN_ECHO packets back to the sender until it receives a Congestion Window Reduced (CWR) packet from the sender, which means the sender has responded to network congestion. The sender only responds to the first ECN_ECHO packet, and ignores others up to one RTT. If the average queue size is greater than the maximum threshold of the RED buffer, the packet is dropped.

ECN, when used with RED buffers at routers, reduces packet losses due to network congestion and delays of TCP based applications [4]. However, choosing the buffer size and optimum parameter values of RED buffers are still open research issues [5].

If the RED buffer is optimally dimensioned and the RED thresholds are appropriately set, zero congestion loss could be achieved by appropriately adjusting the source's congestion window size based on feedback from ECN signals (see [1], [6] for preliminary studies on achieving zero congestion loss). The *objective* of this paper is to propose a modification to TCP, called Diff-C-TCP, to improve the TCP performance over lossy links. We develop Diff-C-TCP based on an extensive analytical model for determining the optimal value of the maximum threshold, in order to achieve zero congestion loss at ECN-capable RED gateways. Since most of network congestion losses can be eliminated by our zero loss model, Diff-C-TCP assumes packet losses to be indicators of link corruption, and uses ECN to determine any losses that may occasionally happen due to network congestion.

The rest of the paper is organized as follows. In Sections II and III, we define the notations and assumptions used in our model which is presented in Section IV. Validation of our

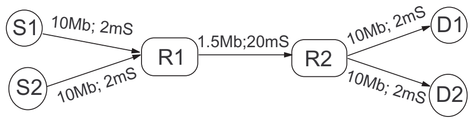

Fig. 2. Network topology for modeling.

proposed model using simulation is described in Section V. The proposed Diff-C-TCP algorithm and its evaluations are presented in Section VI, followed by the concluding remarks in Section VII.

II. Notations

We consider a RED buffer at R1 (see Figure 2) which is fed by multiple TCP sources. The link connecting R1 and R2 is the bottleneck link which causes congestion at R1. The sources, destinations and the RED buffer use ECN for end-to-end congestion control. The following notations will be used in our model (developed in Sec. IV):

- $Q(t)$: *Instantaneous* queue size of RED buffer at time t.
- $\overline{Q}$, $\overline{Q}_{max}$: *Average* and *maximum average* queue sizes respectively at the RED buffer.
- ω: *Weighting factor* for calculating $\overline{Q}$.
- $p(t)$: *Marking probability* at the RED buffer at time t.
- min_{th}, max_{th}: *Minimum* and *maximum* thresholds respectively of a RED buffer.
- L: *Buffer size* of RED buffer.
- m: total number of TCP flows.
- $W_i(t)$: *Window size* of the i^{th} TCP flow at time t, $t \geq 0$, $i = 1, ..., m$.
- $\overline{r_i}$: *Average Round Trip Time* (RTT) for the i^{th} TCP flow, $i = 1, ..., m$.
- $T[1]$: *Waiting time* for the first marking event after the average queue size exceeds min_{th} [6].
- β_i: Number of window size increasing times during time $T[1]$ for the i^{th} TCP flow, $i = 1, ..., m$.
- t_0: Time when the first packet is marked at the RED buffer [6].
- t_1: Time when the last packet, which was sent just before the first window size reduction, arrives at the RED buffer [6].

For every packet arrival, the RED buffer estimates $\overline{Q}$ using the following exponential weighted moving average algorithm [3]:

$$\overline{Q} \longleftarrow (1-\omega)\overline{Q} + Q(t)\omega. \tag{1}$$

The packet marking/dropping probability $p(t)$ is then calculated as follows:

$$p(t) = \begin{cases} 0, & 0 \leq \overline{Q} < min_{th} \\ \frac{(\overline{Q}-min_{th})max_p}{max_{th}-min_{th}}, & min_{th} \leq \overline{Q} \leq max_{th}. \\ 1, & max_{th} < \overline{Q} \leq L \end{cases} \tag{2}$$

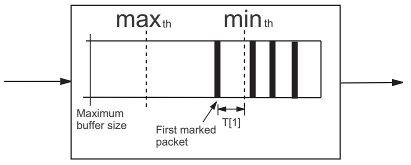

Fig. 3. Analytical model of a RED buffer.

III. Assumptions

We make the following assumptions regarding the RED buffer and TCP sources in our analytical model for zero buffer loss (in Sec. IV).

- For small ω (as suggested in [3]), $\overline{Q}$ varies very slowly, so that consecutive packets are likely to experience the same marking probability [7].
- The random packet marking in flow i is described by a Poisson process with time varying rate $\lambda_i(t) = p(t)W_i(t)/r_i(t)$ [8]. Accordingly, the waiting time ($T_i[n]$) for the $n-$th marking event of flow i, which is given by $T_i[n] = \sum_{k=1}^{n} X_i(k)$, is a Gamma distributed random variable. $X_i(k)$ is the time interval between the $(k-1)$ and the $k-$th marking events for flow i. Specifically, the expected value of the waiting time for the first marking event is $E[T_i[1]] = 1/\lambda_i(t)$.
- All TCP sources start sending at the same time, and all packets are of the same size (as used in [6]). (We are currently analyzing the performance of our model with multiple TCP sources starting randomly.) The queue size is measured in packets.
- Packet drops at an ECN-capable RED buffer are due to either $Q(t) > L$ (buffer overflows) or $\overline{Q} > max_{th}$. The discussion of packet drops due to buffer overflows is out of the scope of this paper.

IV. Proposed model for max_{th} at RED buffers

In this section, we develop a model to estimate the optimal value of max_{th} for zero packet loss at the RED buffer. We start with the recommended value of $max_{th} = 3 * min_{th}$ [3] to develop our model.

Figure 3 shows our analytical model of a RED buffer. When the average queue size is in the steady-state condition (during which the sources are in the congestion avoidance phase), the instantaneous queue size at time t_0 is

$$Q(t_0) = min_{th} + \sum_{i=1}^{m} \beta_i, \tag{3}$$

where β_i is given by

$$\beta_i = \frac{E[T[1]]}{\overline{r_i}} = \frac{1}{\lambda_i(t)\overline{r_i}} = \frac{1}{p(t)W_i(t)}, \quad i = 1, ..., m. \tag{4}$$

Since the difference between t_0 and t_1 is one RTT, and the window size of a source is increased by one per RTT during the congestion avoidance phase, the instantaneous queue size at time t_1 can be expressed as

$$Q(t_1) = min_{th} + \sum_{i=1}^{m}(\beta_i + 1). \quad (5)$$

The average queue size is estimated using an exponential weighted moving average as shown in Eqn. (1). If time is discretized into time slots with each slot being equal to one RTT, the RED's average queue size estimation algorithm at the $k-$th slot can be expressed as

$$\overline{Q}[k+1] = (1-\omega)\overline{Q}[k] + Q[k]\omega. \quad (6)$$

Similarly, if we assume t_1 is equal to slot k in time, Eqn. (5) can be rewritten as

$$Q[k] = min_{th} + \sum_{i=1}^{m}(\beta_i + 1). \quad (7)$$

In practice, ω is very small, and the congestion window size increases by one for every RTT during the congestion avoidance phase. Therefore, before the first marking event happens (i.e., no congestion control), it is reasonable to consider both the instantaneous queue size and the average queue size to be constant within a very short time period (see the first assumption in Section III). Thus, by plugging Eqn. (7) into Eqn. (6), and assuming that the average queue sizes during the two previous consecutive time slots are the same, the average queue size estimated at time t_1 can be solved iteratively to be:

$$\overline{Q}_{max} = \overline{Q} = min_{th} + \sum_{i=1}^{m}(\beta_i + \omega). \quad (8)$$

The first marking event is followed by many random ECN marking events resulting in congestion window size adjustment of TCP sources. The average queue size stays at a level which is smaller than the average queue size at time t_1, as will be shown by our simulation results in Figure 4 later. Therefore, **Eqn. (8) gives the maximum average queue size for achieving zero packet loss, i.e. this is our suggested value of max_{th}.**

V. Model Verification

We have simulated the topology of Figure 2 using *ns-2* (*ns* Version 2.1b6) simulation tool from Berkeley [9]. To make the different cases comparable, we choose RTT=59 ms for all TCP connections. The RED parameters vary depending on the case as described below.

A. Verification of max_{th}

We use three cases as shown in Table I to verify the validity of max_{th} as suggested by our model in Sec IV. Case 1 uses the recommended RED parameters [3]. It is seen that $\overline{Q}_{max}$ (which we have suggested in Eqn. (8) as the value to be used for max_{th}) agrees with the value obtained from simulation.

Figure 4 shows the congestion window size, and Figure 5 shows instantaneousness queue size, and average queue size for Case 1. We see that the instantaneous queue size is maximum when the congestion window size reaches the slow start threshold. The average queue size is maximum just before the first marking event at time $t = 1.9$ sec. Due to space limitations, we do not show congestion window and queue sizes for Cases 2 and 3, but we have observed similar relationships.

B. Verification of optimality of max_{th}

To prove that the value of max_{th}, as suggested by our model, is optimal in achieving zero packet loss at a RED buffer, we simulated three cases as shown in Table II. Case 1,

TABLE I
Comparison between simulation results and analysis results for maximum threshold (i.e., maximum average queue size).

Sim. cases	ω	min_{th} (Packets)	max_{th} (Packets)	max_p	$\overline{Q}_{max}$ (Packets) Analy.	$\overline{Q}_{max}$ (Packets) Sim.
Case 1	0.002	5	15	0.1	**7.3**	**7.6**
Case 2	0.002	5	15	0.2	**6.7**	**7.1**
Case 3	0.002	7	21	0.1	**9.6**	**9.9**

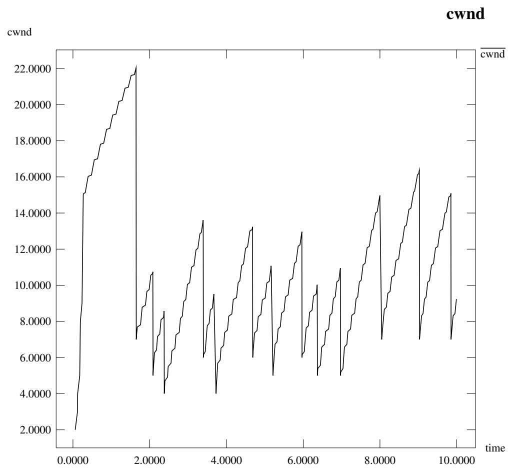

Fig. 4. Congestion window size of TCP source 1 for Case 1 in Table I.

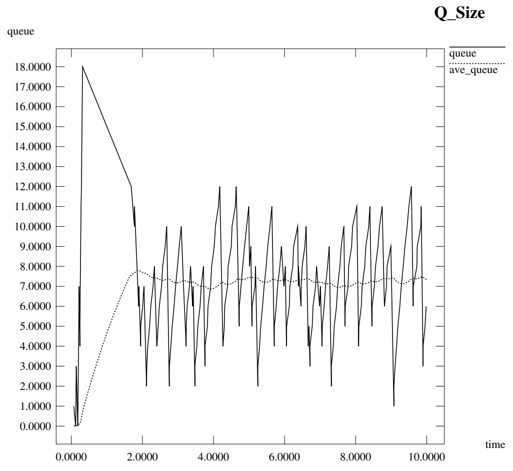

Fig. 5. Instantaneous queue size and average queue size for Case 1 in Table I.

TABLE II

PACKET DROPS WITH DIFFERENT RED PARAMETERS.

Simulation cases	ω	min_{th} (Packets)	max_{th} (Packets)	max_p	Packet drops
Case 1	0.002	5	**15**	0.1	**No**
Case 2	0.002	5	**8**	0.1	**No**
Case 3	0.002	5	**7**	0.1	**Yes**

which uses $max_{th} = 15$, as per the recommended RED parameters [3], results in zero packet loss. Case 2 which uses $max_{th} = 8$ (**almost half the recommended value of 15**) as recommended by our model also results in zero packet loss. We conclude that the smaller value of max_{th} suggested by our model achieves the same congestion control effects as earlier larger recommended values. **Using the value of max_{th} as suggested by our model thus results in smaller buffer size and queueing delay.**

Case 3, using $max_{th} = 7$ which is one less than the value suggested by our model, results in packet loss. We therefore, conclude that **the value of max_{th} as obtained from our model is the minimum (optimal) value required to ensure zero packet loss at a RED buffer**.

VI. DIFF-C-TCP: APPLICATION OF ZERO PACKET LOSS MODEL TO SATELLITE NETWORKS

From the previous discussion, we can eliminate almost all network congestion losses in a heterogeneous network environment involving lossy links. Therefore, with the negligible error all losses can be attributed to random losses. This leads to our proposed Diff-C-TCP algorithm discussed in this section.

A. The Proposed Diff-C-TCP

We assume that our proposed algorithm is used within a WAN or an enterprise network, where it is possible to make all routers and end-systems ECN-capable.

Authors in [10] pointed out that packet losses due to buffer overflows is relatively infrequent when a majority of end-systems become ECN-capable, and participate in TCP or other compatible congestion control mechanisms. Furthermore, by appropriately selecting the RED threshold as described in Section IV, we are able to eliminate congestion losses. In addition, corruption losses become more significant compared to packet losses due to buffer overflows in lossy satellite links. Therefore, it is reasonable to assume that all loss events are due to links errors, *unless* the congestion is explicitly reported by ECN.

Figure 6 shows the kernel of our proposed Diff-C-TCP algorithm at the sender's side. A Diff-C-TCP sender treats the situation that the retransmit timer times out without receiving any ECN_ECHO packet and (or) receiving duplicate acknowledgements as the indication of link errors. Most often, this is the case in a network with lossy links (packet losses due to link errors). In this case, the Diff-C-TCP source does not decrease *cwnd*. If the Diff-C-TCP sender receives the ECN_ECHO packet sent by the receiver, the sender treats it as network congestion and triggers the Fast Recovery algorithm [10] as in the current TCP.

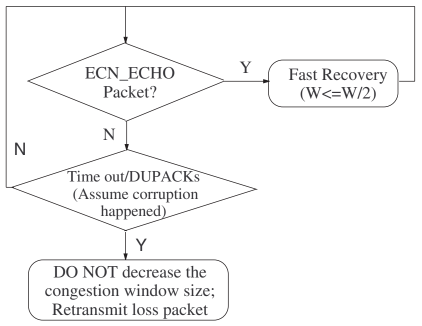
ECN_ECHO Packet?
Y
Fast Recovery (W<=W/2)
N
N
Time out/DUPACKs (Assume corruption happened)
Y
DO NOT decrease the congestion window size; Retransmit loss packet

Fig. 6. States transition diagram of Diff-C-TCP kernel.

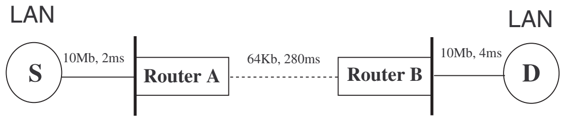

Fig. 7. LAN interconnection using a lossy wireless link.

In Diff-C-TCP, the congestion window size is appropriately controlled in the presence of either network congestion or corruption. Congestion window is halved using the Fast Recovery algorithm when there is network congestion (explicitly notified by ECN_ECHO packets), and maintains the previous value in the presence of corruption.

B. Simulation Methodology

We have evaluated the performance of our Diff-C-TCP algorithm using *ns 2*. The ECN implementation is based on RFC 3168 [10]. Our network topology for conducting simulations is shown in Figure 7. Two local area network (10 Mbps) are connected using a satellite link (64Kbps) with a propagation delay of 280ms. Like previous researchers [11], [12], a Uniform random error model is used to generate random errors on the wireless link. Instead of dropping packets at routers, ECN-capable RED gateway are used in our simulations to set the CE bit in the packet header.

All the links in Figure 7 are labelled with a (bandwidth, propagation delay) pair. The full-duplex link between router A and router B has a BER (bit-error rate), which varies between $1e^{-7}$ to $1.2e^{-4}$ in our simulation. The receiver's advertised window size, which is also equal to the initial *ssthresh* at the sender, is set to 30 segments. The packet size is set to 1000 bytes (when BER is below $5e^{-5}$) or 512 bytes (when BER exceeds $5e^{-5}$) as explained below. Ftp was used in our simulation to transfer data from the source to destination.

C. Simulation Results

In this section, we present and compare the *goodput* (bit/s) (the amount of useful information being received by the receiver per second, excluding errors) and the *normalized throughput* of our proposed Diff-C-TCP with the traditional TCP.

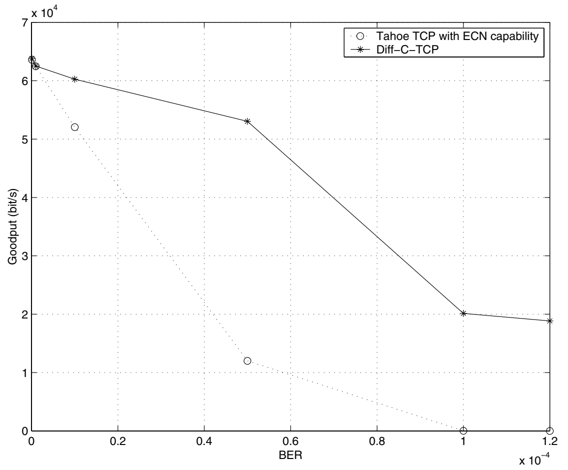

Fig. 8. Comparison of goodput (bit/s).

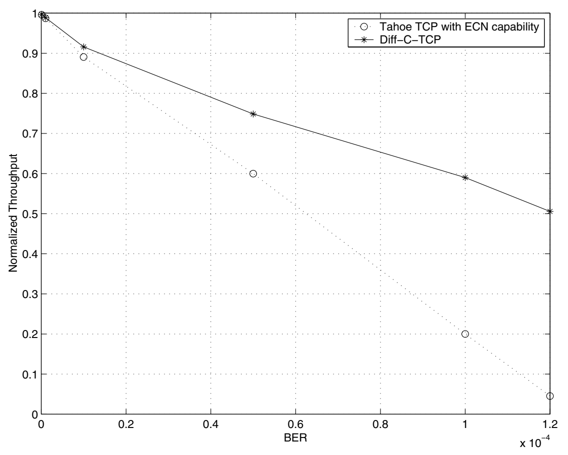

Fig. 9. Comparison of normalized throughput.

Figure 8 compares the goodput in bit/s of both Diff-C-TCP and the current TCP with ECN capability. The normalized throughput of both the TCPs are shown in Figure 9. We see that Diff-C-TCP's throughput is much higher than that of the current TCP with ECN capability. At a BER of $5e^{-5}$, the goodput of our Diff-C-TCP is almost **5 times** higher than that of the current TCP. From Figures 8 and 9, this improvement is much higher at higher values of BER. In addition, the throughput of the current TCP with ECN suffers more severely than our Diff-C-TCP as the error rate increases. We can also see that, with the increase of the value of BER, the throughput of the current TCP with ECN capability decreases much faster than the throughput of our Diff-C-TCP. For example, according to Figure 8, when the value of BER increases from $1e^{-5}$ to $5e^{-5}$, the current TCP's goodput decreases by **77 %** in contrast to our Diff-C-TCP whose goodput only decreases by **12 %**. This is also valid for normalized throughput as shown in Figure 9.

As mentioned previously, the current TCP with ECN makes an assumption that every loss event is caused by network congestion, and a congestion control algorithm is triggered. As a result, the congestion window size must be reduced. Therefore, each loss event, regardless of whether it is due to congestion or corruption, degrades the throughput. With Diff-C-TCP, all packet losses are assumed to be caused by link errors, and network congestion is indicated by the receipt of ECN_ECHO packets. The congestion window will not be changed in the presence of corruption losses. Thus, only network congestion can affect its throughput. Furthermore, for the current TCP with ECN, a higher value of BER results in more packet drops and more frequent reduction of the congestion window. Because of this, higher values of BER result in high frequency of reduction of the congestion window size. It is therefore almost impossible for the congestion window size to reach a high value, even during a non-congestion-loss period.

VII. Conclusion

In order to achieve zero congestion loss at the ECN-capable RED gateway, we have developed an analytical model to estimate the optimal value of maximum threshold of a RED buffer Based on achieving zero congestion losses at RED gateways, we have proposed Diff-C-TCP to enhance the TCP throughput in the presence of non-congestion related losses in a lossy satellite network.

We have shown that the analytical model matches our simulation results very well. The proposed Diff-C-TCP has been found to significantly improve TCP performance in lossy satellite links. This work can be expanded by using more complex network topology for analysis, as well as simulation verification.

References

[1] S. Kunniyur and R. Srikant, "End-to-end congestion control schemes: Utility functions, random losses and ECN marks," in *INFOCOM*, Tel Aviv, Israel, March 2000, pp. 1323–1332.

[2] S. Floyd, "TCP and explicit congestion notification," *ACM Computer Communication Review*, vol. 24, no. 5, pp. 10–23, October 1994.

[3] S. Floyd and V. Jacobson, "Random early detection gateways for congestion avoidance," *IEEE/ACM Transaction on Networking*, vol. 1, pp. 397–413, August 1993.

[4] J. H. Salim and U. Ahmed, "Performance evaluation of explicit congestion notification (ECN) in IP networks," RFC 2884, July 2000.

[5] B. Zheng and M. Atiquzzaman, "Low pass filter/over drop avoidance (LPF/ODA): An algorithm to improve the performance of RED gateways," *International Journal of Communication Systems*, vol. 15, no. 10, pp. 899–902, 2002.

[6] C. Liu and R. Jain, "Improving explicit congestion notification with the mark-front strategy," *Computer Networks*, vol. 35, no. 2-3, pp. 185–201, February 2001.

[7] T. Bonald, M. May, and J. Bolot, "Analytic evaluation of RED performance," in *INFOCOM*, Tel-Aviv, Israel, March 2000, pp. 1415–1424.

[8] V. Misra, W. Gong, and D. Towsley, "Fluid-based analysis of a network of AQM routers supporting TCP flows with an applicaiton to RED," in *ACM SIGCOMM*, Stockholm, Sweden, 2000, pp. 151–160.

[9] V. P. U. Berkeley/LBNL, "ns v2.1b6: Network simulator," http://www-mash.cs.berkeley.edu/ns/, January 2000.

[10] K. K. Ramakrishnan, S. Floyd, and D. Black, "The addition of Explicit Congestion Notification (ECN) to IP," RFC 3168, September 2001.

[11] H. Balakrishnan, V. Padmanabhan, S. Seshan, and R. Katz, "A comparison of mechanisms for improving TCP performance over wireless links," *IEEE/ACM Transactions on Networking*, vol. 6, no. 5, pp. 756–769, December 1997.

[12] C. Parsa and J. Garcia-Luna-Aceves, "TULIP: A link-level protocol for improving TCP over wireless links," in *WCNC*, New Orleans, Louisana, September 1999, pp. 1253–1257.