An Adaptive End-to-End Loss Differentiation Scheme for TCP over Wired/Wireless Networks

Chang-hyeon Lim, *and* ***Ju-wook Jang***

Department of Electronic Engineering, Sogang University, Shin-soo 1 Mapo Seoul, Korea

Summary
We propose a robust end-to-end loss differentiation scheme to identify the packet losses due to congestion for TCP connections over wired/wireless networks. We use the measured RTT values in determining whether the cause of packet loss is due to congestion over wired path or regular bit errors over wireless paths. The classification should be as accurate as possible to achieve high throughput and maximum fairness for the TCP connections sharing the wired/wireless paths. The accuracies of previous schemes in the literature depends on varying network parameters such as RTT, buffer size, amount of cross traffic, wireless loss rate and congestion loss rate. The proposed scheme is robust in that the accuracy remains rather stable under varying network parameters. The basic idea behind our scheme is to set the threshold for the classification to be a function of the minimum RTT and the current sample RTT, so that it may automatically adapt itself to current congestion level. When the congestion level of the path is estimated to be low, the threshold for a packet loss to be classified as a congestion loss is increased. This avoids unnecessary halving of the congestion window on packet loss due to regular bit errors over wireless path and hence improves the TCP throughput. When the congestion level of the path is estimated to be high, the threshold for a packet loss to be classified as a congestion loss not to miss any congestion loss is decreased and hence improves the TCP fairness. In our ns 2 simulations, the proposed scheme correctly classifies the congestion losses under varying network parameters while previous schemes show some dependency on subsets of parameters.

Key words:
Loss differentiation, Congestion control, TCP, Wireless.

1. Introduction

TCP congestion control runs under the basic assumption that any packet loss is the indication of congestion. However, the assumption does not hold when the TCP flow path includes wireless part. In such a case the packet loss may not come from congestion but from regular bit errors over wireless path. TCP throughput may be unnecessarily degraded due to the packet loss from bit errors over wireless part even though there is little congestion.

Research on improving performance of TCP over hybrid wired/wireless paths has focused on differentiating packet losses using information readily available to TCP: congestion window size, inter-arrival time between ACK packets, and changes in round-trip time(RTT) [1][11][4][3]. However, correct classification based on these metrics has been found to be difficult [11]. The reason is that nature of losses is weakly correlated with the observable metrics(RTT, congestion window size, inter-arrival of ACKs, and Jitter) [12]. In [2], the timestamp option in TCP is used to measure inter-arrival Jitter between data packets. From our simulations, this scheme tends to regard most of packet loss as due to wireless channel errors.

We propose a new end-to-end scheme to precisely infer the nature of packet losses over wired/wireless networks. (From now on, we call packet losses due to congestion congestion losses and the losses due to wireless channel errors wireless losses.) Instead of fixed threshold in estimating the cause of individual packet loss, our scheme employs a moving threshold for relative change of RTT against the minimum RTT. In particular, we use the ratio of a difference between a sample RTT and a mean of RTT over a variance of RTT. We classify a packet loss as congestion loss if this value exceeds the threshold, or classify it as wireless loss, otherwise. The threshold is defined as a function of minimum and sample RTT, which decreases as congestion level increases. The moving threshold is lowered for the differentiator to be more sensitive to congestion loss when the network is congested while it is increased when the network is unloaded.

Our scheme is shown to be more stable in identifying the congestion losses under varying network conditions than previous schemes [1][11][4][3][2]. The accuracy of correctly detecting wireless loss on unloaded path is similar to previous schemes as far as congestion loss rate is kept low.

The rest of this paper is organized as follows. We describe related work in Section 2, and in Section 3 we propose a new scheme of differentiating the nature of packet loss and its evaluation is described in Section 4. In Section 5 we conclude.

Manuscript received March 5, 2007
Manuscript revised March 25, 2007

2. Related work

Known schemes to improve the TCP throughput over wired and wireless paths can be divided into three categories: First, network-based schemes locate an agent at the access point/base station on the TCP path to locally recover the wireless loss at transport or link layer. Network-based schemes at transport layer do not maintain the end-to-end semantics of TCP and may require states to be maintained and packets to be buffered at the base station [5] [6]. At link layer, they can be purely local or aware of the semantics of the TCP protocol [7]. In an explicit loss notification approach, the receivers/network routers mark the acknowledgements with appropriate notification of distinguishing the channel errors from congestion losses [13]. Then the senders respond to the notification. The explicit loss notification approaches require modifications to network infrastructure, the receiver, and the senders. Finally, end-to-end schemes modify the TCP congestion control algorithm to distinguish the losses due to congestion in the network from other random losses over wireless paths. They can be deployed easily via simple modification to the TCP congestion control at sender or receiver side [1] [2] [4] [3] [11]. Our scheme also falls into end-to-end schemes.

The Vegas predictor [11][1] estimates the outstanding packets over the network from the difference between the expected and the actual bandwidth. Biaz and Vaidya [11] study the accuracy of the Vegas predictor and show the accuracy is dependent on network parameters. TCP Veno [1] employs the vegas predictor to differentiate the cause of packet losses.

$$N = \left(\frac{cwnd}{BaseRTT} - \frac{cwnd}{RTT} \right) \times BaseRTT \qquad (1)$$

where cwnd is the current TCP window size, RTT is the smoothed round-trip time measures, and BaseRTT is the minimum of measured round-trip times. In this scheme, the network is considered in congestion if N exceeds threshold ß. In NS-2 simulation, we use the setting for ß equal to 3.

The spike[3] is a scheme based on RTT measurement. In this work, we use RTT instead of Relative One-way Trip Time(ROTT). It keeps track of the minimum and maximum RTT values and computes thresholds $B_{spikestart}$ and $B_{spikeend}$. The two thresholds are determined as some values between the minimum and maximum RTT values where $B_{spikestart} > B_{spikeend}$. TCP enters the spike state if RTT exceeds $B_{spikestart}$, but it exits the spike state if RTT drops below $B_{spikeend}$. In the spike state, all of the packet losses are regarded as due to congestion. Recently, TCP Westwood+ with bulk repeat(TCPW-BR) [15] employs this scheme to differentiate the cause of packet losses. The spike scheme has two thresholds $B_{spikestart}$ and $B_{endspike}$ defined as:

$$\begin{aligned} B_{spikestart} &= BaseRTT + \alpha \cdot (MaxRTT - BaseRTT) \\ {}_{*}B_{spikestart} &= BaseRTT + \beta \cdot (MaxRTT - BaseRTT) \end{aligned} \qquad (2)$$

where MaxRTT is the maximum of measured round-trip times. The parameters α and ß affect the sensitivity and aggressiveness of Spike. In TCPW-BR, the setting for α and ß in [15] is 0.4 and 0.05, respectively. In our simulations, this setting regards the most of packet losses as due to congestion. i.e. The accuracy of congestion classification is almost 100%, but of wireless classification is about 0%. Therefore we choose the setting in [3]. α and ß are set to $\frac{1}{2}$ and $\frac{1}{3}$, respectively.

NCPLD [4] is to find the knee point in throughput-load curve where the throughput decreases negligibly as the load increases. The TCP sender estimates the total number of segments in flight over the path to the receiver. This scheme tends to correctly detect the congestion loss, but tends to regard most of packet losses as congestion losses.

The TCP sender estimates the total number of segments in flight over the path to the receiver as follows:

$$TotalPipeSize = \frac{1}{2} \cdot T_k \cdot \frac{cwnd_k - cwnd_k - 1}{T_k - T_k - 1}$$

where T_k is the round-trip time measured on receipt of the k-th ACK. The NCPLD scheme requires also an estimate of the bandwidth-delay product. To this aim, they used an exponentially weighted moving average(EWMA) filtering the ACK reception rate to get an estimate of the transmission rate B_{TX}. Both estimates yield the current value of T at the knee-point as presented by Eq. (3). The NCPLD scheme regards packet loss due to congestion if the current T is greater than T_{kp}.

$${}^{v}T_{kp} = BaseRTT + \frac{1}{2} \cdot T \cdot \frac{B_{TX} \cdot BaseRTT}{TotalPipeSize} \qquad (3)$$

These loss differentiators estimate the change of the queuing delay to detect the packet loss due to congestion which occurs after build-up of the network buffer. However, assuming that round-trip path and location of the bottleneck do not change, the accuracy depends on network conditions. For accurate loss differentiation, favorable values of network conditions are as follows: round-trip time small, router queue size large, and input bandwidth to the bottleneck small [11].

In jitter-based TCP scheme (JTCP) [2], a TCP sender measures jitter ratio from the packet-by-packet interarrival time at the receiver side and the sending interval for one RTT. It requires the timestamp option in TCP header. The jitter ratio(Jr) can be written as following,

$$J_r = \frac{(t_T(i) - t_T(i-cwnd)) - (t_s(i) - t_s(i-cwnd))}{t_T(i) - t_r(i-cwnd)} \tag{4}$$

where t_r is the arrival time for i-th packet at a receiver, and $t_s(i)$ is the sending time for i-th packet at a sender. JTCP regards a packet loss as due to congestion if the jitter ratio is greater than the congestion window inverse(k/cwnd). A threshold constant k is recommended to set to 1 in [2]. This scheme improves the TCP throughput by improving the accuracy of correctly detecting the wireless losses.

These end-to-end schemes use the queue build-up for detection of congestion losses in end-to-end manner. However, their accuracies are unstable due to depending on network conditions [11] [12]. Some of them consider only the improvement in TCP throughput or the achievement of fairness, and then tend to regard most of packet losses as one type between congestion and wireless losses.

In this paper, a simple robust and adaptive scheme is proposed to distinguish the nature of packet loss for TCP using information readily available to TCP. The scheme's accuracy for congestion losses is shown to be stable without favorable network condition. The scheme can capture more wireless losses when the network is unloaded while it can capture more congestion losses when the network is congested. Furthermore, TCP connections using the scheme can improve the throughput over wired and wireless network, and can achieve the intraprotocol fairness and the TCP friendliness.

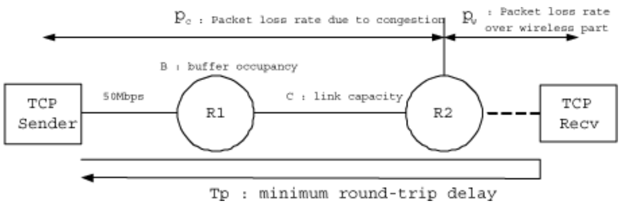

Pc : Packet loss rate due to congestion
Pw : Packet loss rate over wireless part
B : buffer occupancy
TCP Sender
50Mbps
R1
C : link capacity
R2
TCP Recv
Tp : minimum round-trip delay

Fig. 1. The heterogeneous network model of wired/wireless paths

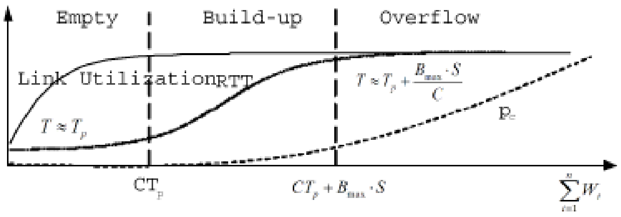


Fig. 2. Three-state of network buffer and the change of the RTT with the aggregation of congestion windows of individual flows

3. The Proposed scheme

In this section, the network model is introduced. We then propose a robust loss differentiation scheme based on the network model and find the optimal coefficient to satisfy both efficiency and fairness. Finally, we show that the proposed scheme is robust against the error of measured values in loss differentiation.

3.1 Network model and observation

Figure 1 shows a heterogeneous network model of wired and wireless paths to be used in the following description of our scheme. p_c denotes the probability of packet loss due to congestion in the wired path while p_w denotes a uniform random packet loss probability over the wireless path. We assume there is no congestion over the wireless path. C denotes the link bandwidth in Mbps and B denotes the number of packets of size S currently occupying the buffer. B_{max} denotes the size of the buffer in packets. Let T_p denote the propagation delay on the path between the sender and receiver. In practice, T_p can be the minimum of measured RTTs. The queuing delay is represented by $\frac{B \cdot S}{C}$. The RTT(T) can be written as Eq. (5).

$$T = T_p + \frac{B \cdot S}{C} \tag{5}$$

Figure 2 shows the relationship among the buffer occupancy, RTT and p_c against $\sum_{i=1}^{n} W_i$, the aggregate sum of the congestion windows of the TCP flows established. W_i denotes the congestion window size of i-th TCP flow out of the n flows sharing the path. When the buffer occupancy nears overflow, we have $T \gg T_p$ and the probability of congestion loss(p_c) becomes increasingly high.

3.2 The proposed scheme

We present a new robust scheme to infer the cause of each packet loss encountered whether it is due to congestion or not. The RTT(T) measured immediately before the current packet loss is used in Eq. (6) as an indicator. $\overline{T}$ and T$_{dev}$ are an exponentially weighted moving average of RTT(T) and the deviation, respectively. They are updated by $\overline{T} = \frac{7}{8}\overline{T} + \frac{1}{8}T$ and $T_{dev} = \frac{3}{4}T_{dev} + \frac{1}{4}\left|T - \overline{T}\right|$ as in most TCP implementations. The current packet loss is determined to be a congestion loss if Eq. (6) is satisfied.

$$T > \overline{T} + T_{dev} \cdot \left(2 \cdot \left(\frac{T_p}{T} \right)^k - 1 \right) \qquad (6)$$

This differentiation depends on four parameters: $\overline{T}$, T$_{dev}$, k, and T$_p$. Statistical values $\overline{T}$ and T$_{dev}$ can indicate the current network condition. A constant k can be chosen for the proposed differentiation to be accurate. T$_p$ can be a criteria to infer the cause of packet loss. We will study on effect of constant k and various network conditions ($\overline{T}$ and T$_{dev}$) on the decision rule. We then study the impact of an overestimated T$_p$ on the decision rule. Finally, we find effective k for the differentiation to be accurate via simulations.

Analysis on decision rule

This is to study on the effect on the decision rule as k, $\overline{T}$ and T$_{dev}$ change. We assume that $T_p \le T \le T_p + \frac{B_{\max} \cdot S}{C}$ on a TCP path with a single bottleneck where B_{max} denotes the maximum length of buffer in packets, S is the packet size in bytes, and C is the bandwidth of the bottleneck link in bps. Then, the decision rule of congestion loss is following:

$$f(T) = T \cdot \left(\frac{T - (\overline{T} - T_{dev})}{2 \cdot T_{dev}} \right)^{\frac{1}{k}} > T_p \qquad (7)$$

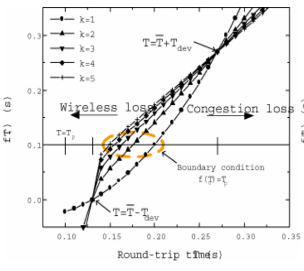
To prove Eq. (7), we can obtain followings from Eq. (6).

$$\frac{T - \overline{T}}{T_{dev}} > 2 \cdot \left(\frac{T_p}{T} \right)^k - 1$$

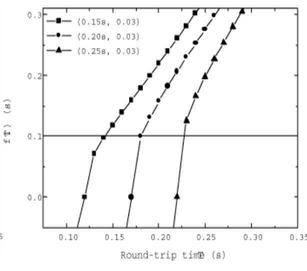
$$\Leftrightarrow \frac{T - \overline{T} + T_{dev}}{T_{dev}} > 2 \cdot \left(\frac{T_p}{T} \right)^k$$

$$\Leftrightarrow T^k \cdot \left(\frac{T - (\overline{T} - T_{dev})}{2 \cdot T_{dev}} \right) > T_p{}^k$$

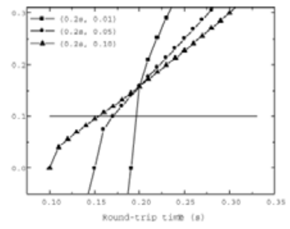
$$\therefore T \cdot \left(\frac{T - (\overline{T} - T_{dev})}{2 \cdot T_{dev}} \right)^{\frac{1}{k}} > T_p$$



(a) boundary vs k



(b) boundary vs $\overline{T}$



(c) boundary vs T$_{dev}$

Fig. 3. Illustrations of (a) decision rule when T$_p$=0.1s, $\overline{T}$ =0.2s and T$_{dev}$=0.07s, (b) the boundary as varying $\overline{T}$ (T$_{dev}$=0.03 and k=3), and (c) the boundary as varying T$_{dev}$ ($\overline{T}$ =0.2s and k=3)

Figure 3 illustrates the decision rule represented by Eq. (7). In Figure 3(a), the boundary T$_0$ of T for classification of the congestion loss decreases down to $\overline{T}$ − T$_{dev}$ as k increases. For k = 1, the boundary is as high s$\overline{T}$, and the differentiator can detect the wireless loss more correctly. When k goes higher than 3, the decrease in T$_0$ becomes negligible. The differentiation with k>3 can be more accurate in detecting the congestion loss. In Figure 3(c), T$_0$ increases, and ƒ(T) increases sharply as T$_{dev}$ increases compared to the change of $\overline{T}$ in Figure 3(b).

Impact of overestimated T_p on T_0

A measured T_p can be overestimated due to persistent congestion for the duration of a TCP connection. An overestimated T_p can result in different boundary compared to the real T_p. Then, we investigate the impact an overestimated T_p on T_0. To do this, we inject new 12 connections running the proposed scheme into the path where old 12 connections running the proposed scheme are already established. In Figure 4, the proposed scheme restricts the error of decision boundary T_0 within 20% even though T_p is overestimated over 100%. As C increases, the range which the RTT fluctuates is reduced because the queuing delay decreases. Then, $\overline{T}$ and T_{dev} also decrease. T_p measured by new connections can be as same as T_p of old connections for larger C.

Choice of k

We evaluated the accuracy of the indicator against the value of k through a simulation using packet losses of known causes. We increase the number of TCP connections sharing the same path as shown in Figure. 1. p_c is the ratio of the number of congestion losses over the total number of packet losses and p_w is similarly defined for wireless losses. Thus, we have $p = p_c+p_w$ where p is the total packet loss rate on the TCP path. A_c is the ratio of the number of congestion losses correctly classified over the total number of congestion losses. A_w is similarly defined for wireless losses. For higher values of k, the threshold drops more rapidly as T increases. Adaptation terms for k =1 and 3 are presented as $\frac{T_p}{T}$ and, $\left(\frac{T_p}{T}\right)^3$ respectively.

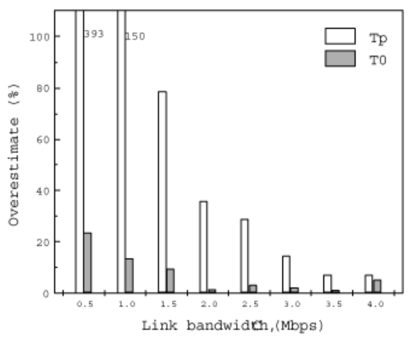
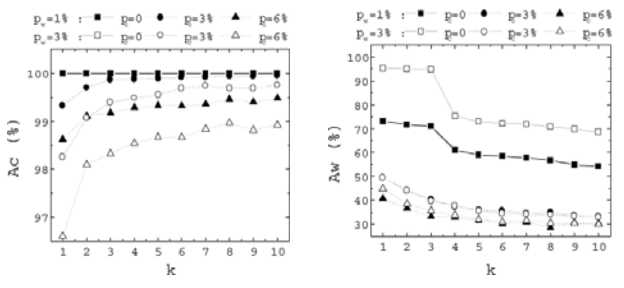


Fig. 4. Overestimate of T_p and T_0 in new connection for k=3 as varying C



(a) (b)

Fig. 5. Impact of k on accuracies of loss differentiator

The latter term decreases more rapidly as T increases, and then it can have the lower threshold in the congested path than the former term. The threshold with $k \geq 4$ can be close to zero as T increases slightly over T_p. k can be chosen through trade-off analysis for best performance in terms of throughput and fairness. Especially, A_c and A_w are important to share the link fairly and improve the throughput against the wireless packet loss, respectively. As shown in Figure 5(a) and 5(b), A_c begins to saturate as k becomes greater than 3. When there is no congestion(p_c = 0), A_w drops sharply as k goes higher than 3. Therefore, we choose k to be 3.

3.3 Modification of TCP Congestion control

To apply the loss differentiator, we modify two blocks in TCP congestion control: they are RTT update and receipt of TDACKs. In RTT update, the TCP sender updates the minimum of RTT which indicates the propagation delay. The receipt of TDACKs is explained in Figure 6. Other related works are implemented in the similar way to compare between them in points of the accuracy and the accuracy's impact on the throughput improvement. Recently, TCP Westwood+ with bulk repeat [15] adopts the loss differentiation into the existing TCP Westwood+ protocol. In particular, they modify the receipt of TDACKs, partial ACK and the timer expiration. In veno [1], procedures of the additive increase of cwnd and the receipt of TDACKs are modified by adding the loss differentiation. JTCP [2] modifies the receipt of TDACKs and the timer expiration. The modification of TCP congestion control using LDA is still an open issue.

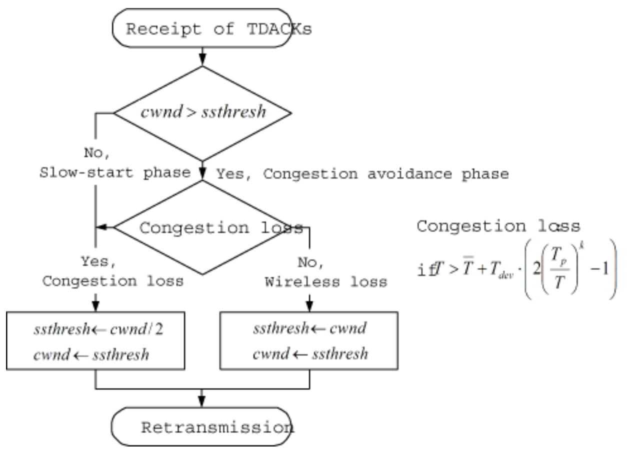

Receipt of TDACKs
$cwnd > ssthresh$
No, Slow-start phase
Yes, Congestion avoidance phase
Congestion loss
Yes, Congestion loss
No, Wireless loss
Congestion loss if $T > \bar{T} + T_{dev} \cdot \left(2\left(\frac{T_p}{T}\right)^k - 1\right)$
$ssthresh \leftarrow cwnd/2$
$cwnd \leftarrow ssthresh$
$ssthresh \leftarrow cwnd$
$cwnd \leftarrow ssthresh$
Retransmission

Fig. 6. Flow chart for the receipt of TDACKs

3.4 Throughput model

We derive the TCP throughput of the proposed scheme following arguments developed by Kelly [18]. For the simplicity, we do not model the behavior after a timeout.

The cwnd is updated upon ACK reception. Each time an ACK is received back by the sender the cwnd is increased by 1/cwnd. On the receipt of TDACKs, the proposed scheme involves to infer the cause of the packet loss. The cwnd is reduced by half if the proposed scheme regards the packet loss as due to congestion. Otherwise, the cwnd will be kept. Let $P_r(c|l)$ denote the probability of packet losses regarded as congestion losses. It results from the aggregate sum of the probability to correctly detect the congestion losses represented by $\frac{P_c}{P}A_c$ and the probability to wrongly infer the wireless loss into the congestion loss represented by $\frac{P_w}{P}(1-Aw)$ as shown in Eq. (12). Therefore, the change in cwnd is $\frac{1}{2}\cdot P_r(c|l)\cdot cwnd \cdot$

$$P_r(c|l) = \frac{1}{p}\left[P_c \cdot A_c + p_w \cdot (1-A_w)\right] \tag{8}$$

Since the total packet loss rate is p, it follows that the expected change of cwnd per update step is:

$$E[\Delta cwnd] = \frac{1-p}{cwnd} - \frac{1}{2}\cdot P_r(c|l)\cdot cwnd \cdot p \tag{9}$$

Since the time between update steps is about $\frac{T}{cwnd}$, by recalling Eq. (9), the expected change in the rate x per unit time is approximately:

$$\frac{\delta x(t)}{\delta t} = \frac{1-p}{T^2} - \frac{1}{2}\cdot P_r(c|l)\cdot p \cdot x^2(t) \tag{10}$$

Eq. (10) is a separable variable differential equation. By separating variables, Eq. (10) can be written as:

$$\frac{\delta x(t)}{x^2(t) - \dfrac{2(1-p)}{T^2\cdot P_r(c|l)\cdot p}} = -\frac{1}{2}\cdot P_r(c|l)\cdot p\cdot \delta t \tag{11}$$

and by integrating each member the following solution can be obtained

$$x(t) = \frac{x_1\cdot(1 + C_0 \cdot e^{-\frac{1}{2}\cdot P_r(c|l)\cdot p\cdot t})}{1 - C_0\cdot e^{-\frac{1}{2}\cdot P_r(c|l)\cdot p \cdot t}}$$

where $x_1 = \frac{1}{T}\sqrt{\frac{2(1-p)}{P_r(c|l)\cdot p}}$ is the root of the equation $x^2(t) - \frac{2(1-p)}{T^2\cdot P_r(c|l)\cdot p} = 0$ and a constant C_0 depends on the initial conditions. The steady state throughput of the proposed scheme is then,

$$\begin{aligned} x_{prd} &= \lim_{t\to\infty} x(t) = \frac{1}{T}\sqrt{\frac{2(1-p)}{P_r(c|l)\cdot p}} \\ &= \frac{1}{T}\sqrt{\frac{2(1-p)}{P_c\cdot A_c + p_w\cdot(1-A_w)}} \end{aligned} \tag{12}$$

From Eq.(12), the improvement in TCP throughput by the loss differentiation scheme depends on both A_c and A_w. When the path is unloaded, A_w is important to achieve higher throughput over wireless network. When the path is loaded, A_c is important to achieve fairness between connections or existing TCP variants.

The proposed scheme can adjust the threshold into the current congestion level: The threshold is increased on the loaded path and is lowered on the unloaded path, automatically. We will show that this property helps the proposed scheme to achieve both the improvement in throughput and the fairness.

4. Performance evaluation

In this section, we evaluate the accuracy and the throughput performance of the proposed and other LDAs on TCP-Sack based on simulations on the ns-2 [9].

The result of simulations is presented in four separate categories:

1. Accuracies A_c and A_w

We present the result of comparison with accuracies of existing LDAs such as Spike, Jitter-based scheme and Vegas predictor with various wireless packet loss rates coexisting with congestion losses.

2. Throughput improvement

This category of simulations helps us understand the effect of LDAs on the throughput with uniform and correlated loss model. We present the result of comparison with accuracies of existing LDAs such as Spike, Jitter-based scheme and Vegas predictor with various wireless packet loss rates coexisting with congestion losses.

3. Interoperability

This part includes 4 types of evaluations of joining the network with persistent congestion, sudden changes in available bandwidth, intraprotocol fairness of injecting multiple LDA connections into the link, and the possibility if LDA connections can coexist with TCP-SACK. We present the result of comparison with accuracies of existing LDAs such as Spike, Jitter-based scheme and Vegas predictor with various wireless packet loss rates coexisting with congestion losses.

4.1 Experimental setup

We evaluate the performance of the proposed scheme via ns-2 (Ver 2.27) [9] simulation. The network model is shown in Figure 7. The bandwidth C is set to 2 Mbps. The size of the buffer B_{max} is set to 50 packets. The one way delay is set to 70ms, which is equal to an half of T_p. We set the packet size equal to 1000 bytes. To generate network traffic, several sets of TCP sources/sinks are used. Among the TCP source/sink pairs, the TCP1 connection is treated as an observable TCP source/sink pair, and all the others are treated as background TCP source/sink pairs used to create the forward(N1) and backward(N2) cross traffic. The TCP1 connection goes through a wired path terminated with a last hop wireless link. The wireless last hop models a mobile user accessing the Internet via a radio link. RTTs of the N1 TCP forward cross traffic connections and of the N2 TCP backward cross traffic connections are the same as the path of the TCP1 connection. The congestion packet loss rate p_c increases as N1 increases. We run simulations 30 times. The simulation time is 1000 seconds. In the each of the 30 runs, we estimate the average values of A_c and A_w.

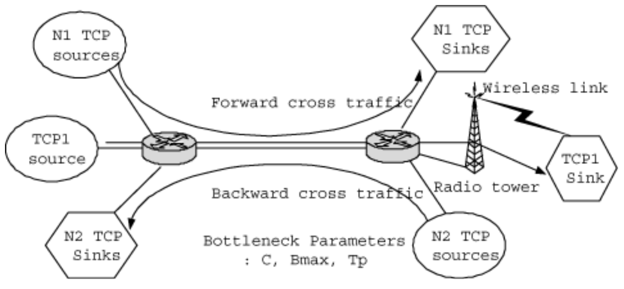


Fig. 7. Mixed wired/wireless scenario

4.2 Accuracies A_c and A_w

We present results of comparison with accuracies of existing LDAs such as Vegas predictor [1][11], Jitter ratio [2], Spike [3] and NCPLD(Non-Congestion Packet Loss Detection) [4]. We mainly focus on the dependency of A_c on p_c, p_w, B_{max}, T_p and wireless packet loss model because A_w is important to achieve higher TCP throughput only if there is no congestion.

Impact of p_c and p_w on A_c and A_w

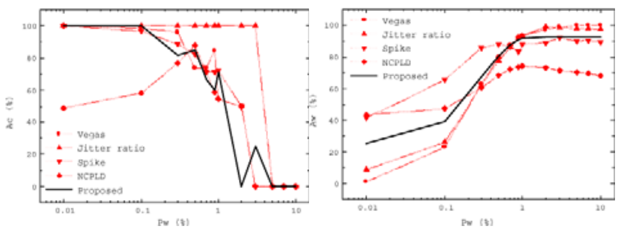
This scenario is to investigate the impact of cross traffic connections on accuracies Ac and Aw. The wireless link is modeled with the uniform packet loss rate p_w varying from 0.01% to 10%. We first simulate a scenario with no cross traffic where both N1 and N2 are set to 0. For p_c=0, our scheme shows similar performance as most of other algorithms as shown in Figure 8(a). For all schemes, A_w increases up to 90% on unloaded path. In the second scenario, we turn on N1 TCP forward cross traffic connections. Figure 8(b) and 8(c) show that A_c for the proposed scheme is much higher than other schemes and shows no dependency on p_w. For p_c=3%, Vegas predictor [1][11] and Spike [3] exhibit high A_c for low p_w and low A_c for high p_w while NCPLD [4] shows low A_c for low p_w and high A_c for high p_w. Especially, Vegas predictor [1][11] shows dependency of A_c on not only p_w but also p_c since A_c is lowered by the increase of p_c. Figure 9 shows the influence of backward cross traffic on A_c. We set the number of backward traffic connections N2 to 5. N1 is set to 4. NCPLD [4] excels all other schemes. Our scheme shows second best performance.

Impact of B_{max} and T_p on A_c and A_w

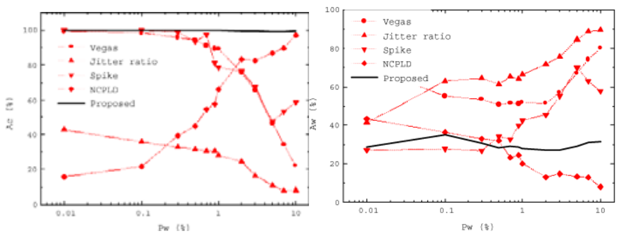
In this scenario, we set the number of forward cross traffic connections N1 to 6, and p_w to 1%. Figure 10(a) shows that the proposed scheme needs small length of network buffer to precisely detect the congestion loss than the Vegas predictor[1][11]. Figure 10(a) shows that A_c for the proposed scheme nears 100% at B_{max}=10, while the A_c for the Vegas predictor [1][11] reaches about 85\% when B_{max} is as high as 35 packets. Spike [3] and NCPLD [4] require the large B_{max} to achieve higher A_c. We then investigate the impact of T_p on accuracies. T_p ranges from 20 ms to 600 ms. Figure 10(b) shows that A_c for the proposed scheme does not depend on the increase of T_p. A_c for Vegas predictor[1][11], Spike [3] and NCPLD [4] decrease as T_p increases. The Jitter ratio [2] is poor at congestion loss, but a good at wireless loss.

Impact of correlated packet loss model on A_c and A_w

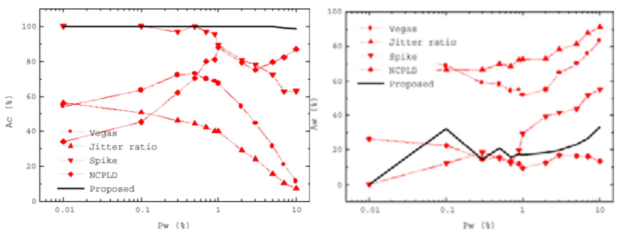
We simulate two cases of N1=0 and 6. Instead of uniform random loss model, we use the Gilbert two state Markov chain to model the loss process[14][17]. In particular, we assume that p_w is equal to 0.1%, when the channel is in the Good state, and is equal to 10\% when the channel is in the Bad state. The duration in the Good state is assumed deterministic and is equal to 1s whereas the duration in the Bad state is assumed also deterministic but this time we consider values ranging from 0.1ms to 100ms. When the duration in a state elapses, the state can transit to a Good or Bad state with a probability p=0.5. The correlated loss model is generally used for modeling the packet loss model of IEEE 802.11 wireless LAN [17].



(a) No cross traffic (N1 = 0)



(b) P_c = 3% (N1 = 15)



(c) P_c = 5% (N1 = 24)

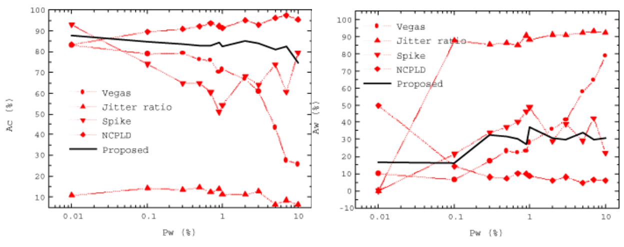
Fig. 8. Accuracies A_c and A_w

Figure 11 shows that the A_c for proposed scheme stays close to 100% against coexistence of correlated packet losses and congestion, and stays over 80\% on the unloaded path. In Figure 11(a), A_c for long duration of BAD state is meaningless due to lack of loss events, but the A_c for NCPLD [4] drops under 70% when congestion loss is dominant. In Figure 11(b), A_c for other schemes depends on the duration of BAD state. A_c for proposed scheme stays close to 100%

4.3 Throughput improvements

In this section, we investigate the effect of the accuracy of the loss differentiation scheme on the TCP throughput improvement as varying environmental variables. The throughput improvement indicates the ratio of the throughput of TCP with the loss differentiation over the throughput of TCP Sack. We write it as following,

$$I_{mp} = 100 \times \left(\frac{Thgt_{LDA}}{Thgt_{TCP}} - 1 \right)$$



(a) (b)

Fig. 9. Accuracies Ac and Aw with coexistence of backward cross traffic

Improvement vs p_w, T_p, C, and loss model

We show the improvement the TCP throughput as varying p_w, T_p, C and loss model under no congestion.

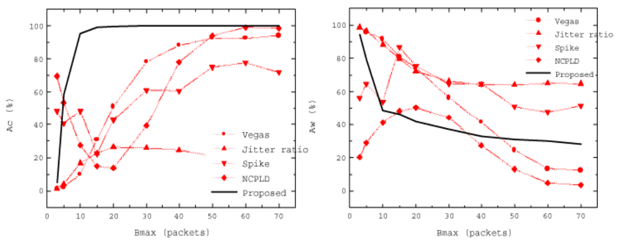
Figure 12(a) shows that the proposed scheme can achieve the highest improvement in TCP throughput for higher p_w in uniform model. Especially, for p_w=10%, the improvement of proposed scheme is shown to be about 30% higher than other schemes. Figure 12(b) shows that the Spike can respond to bursty losses in correlated model. The proposed scheme still achieves the second best improvement overall.

Figure 12(c) and 12(d) show that the improvement of proposed scheme can increase with T_p and C. In particular, for $T_p > 0.4$s or C > 4Mbps, the proposed scheme achieves the best improvements. Improvements of Spike and NCPLD are lowered from T_p = 0.5s as shown in Figure12(c). In Figure 12(d), other schemes except the Jitter ratio fail to increase the improvement for C > 6Mbps.

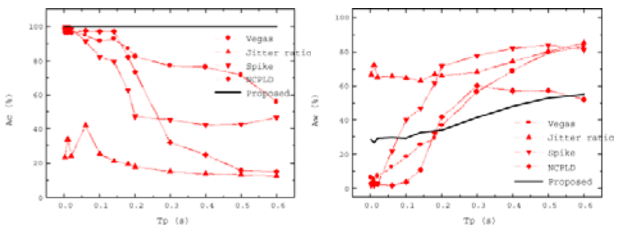
4.4 Interoperability between flows

We investigate the interoperability in four scenarios: new connections on the path which is already loaded, sudden changes in traffic, intraprotocol fairness and friendliness.

In the first scenario, a bunch of LDA connections is newly established on the path which is already loaded. The new connections can overestimate T_p . The Vegas predictor, Spike, NCPLD and proposed scheme can be affected by this. In the second scenario, a bunch of LDA connections is affected by sudden changes in available bandwidth. Particularly, in the first and the second scenario, the network consists of 24 connections. Half the flows do long-term ftp transfer starting at time 0 seconds. We call this traffic as old connections. The other half of the flows carry shorter ftp transfer (referred henceforth as new connections) starting at 100 seconds and lasting for 100 seconds. Thus, 100 seconds after the old connections are started, the available network bandwidth goes down by 50 %. At 200 seconds, the new connections stops, and the available bandwidth doubles back to the original level.



(a) Accuracies vs B_{max}



(a) Accuracies vs B_{max}

Fig. 10. Accuracies A_c and A_w as varying B_{max} and T_p

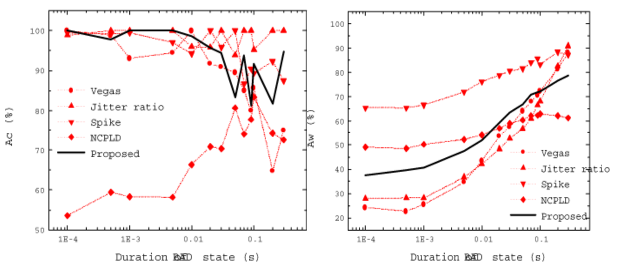
The third scenario is to evaluate the fairness between connections running the same LDA on the same path. We increase the number of flows from 1 to 30 for p_w=1% and 3%. Finally, the last scenario is to study the friendliness of LDA with existing TCP versions.

Impact of overestimate of T_p

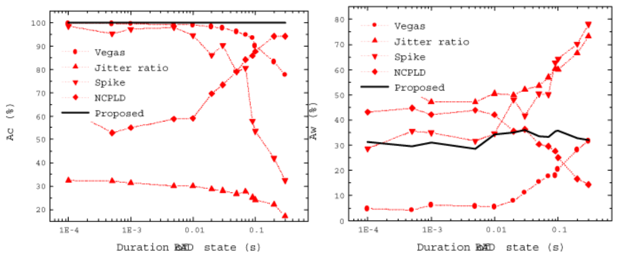
We study the impact of overestimate of T_p. In this scenario, the old connections run LDA, and the new connections run LDA. We evaluate the quantity of link occupied by new connections during their life. Figure 13(d) shows that the overestimate of T_p in the proposed scheme does not affect to join a loaded path compared with Vegas predictor and NCPLD. For C < 5Mbps, new connections of Vegas predictor and NCPLD occupy the link capacity over 50%. In Figure 13(b), old connections run TCP-Sack. TCP-Sack of existing TCP version suffers from new connections of Vegas predictor and NCPLD as C increases. The proposed scheme can limit its occupation of link capacity close to 50%.

Sudden changes in available bandwidth

In this scenario, old connections run LDA, and new connections run TCP-Sack. Similar to above scenario, Vegas predictor and NCPLD still occupy the link capacity upto 70% after new connections come. However, the proposed scheme can respond to the sudden change in traffic because it occupies close to 50%. The Jitter ratio and Spike can behave similar to the proposed scheme (See Figure 14).



(a) No cross traffic (N1 = 0)



(b) Pc = 1% (N1 = 6)

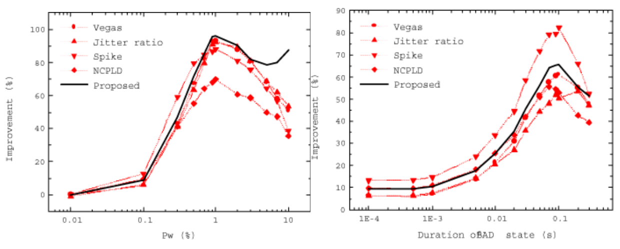
Fig. 11. Accuracies A_c and A_w with correlated packet loss model

Fairness

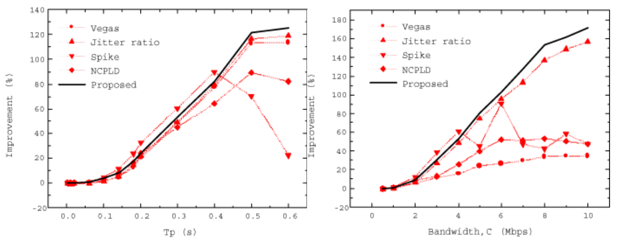
To share the network capacity fairly is another key performance requirement for a wireless Internet connection. No one can be permitted to consume most of the resources especially when the link capacity is insufficient. We use the Jain's fairness index [10] as in Eq. (13) where x_i and n denote the throughput of i-th flow and the number of flows, respectively.

$$FairnessIndex = \frac{\left(\sum_{i=1}^{n} x_i\right)^2}{n \cdot \sum_{i=1}^{n} x_i^2} \qquad (13)$$

In intraprotocol fairness scenario, we inject multiple flows (2-30) into the links and calculate their fairness values. Figure 15 shows the fairness index for each schemes with $p_w = 1\%$ and 3%. Fairness indices of the jitter ratio and the proposed scheme stay over 0.98 as same as TCP Sack. Connections of Vegas predictor, NCPLD and Spike cannot share the bottleneck fairly with increasing the number of connections on the path as shown in Figure 15(a). For $p_w = 1\%$, the index of Spike begins to be lowered under 0.98 from 3 connections. The Vegas predictor and NCPLD achieve poor fairness from 10 and 15 connections, respectively. For $p_w = 3\%$, the fairness index of NCPLD stays over 0.97, but it is still lower than indices of Jitter ratio and proposed scheme. For both Vegas predictor and Spike, decision rules depend on T_p measured by each flow. However, the proposed scheme using T_p for adapting the threshold is shown to behave like TCP-Sack on the congested path.



(a) uniform model (b) Correlated model



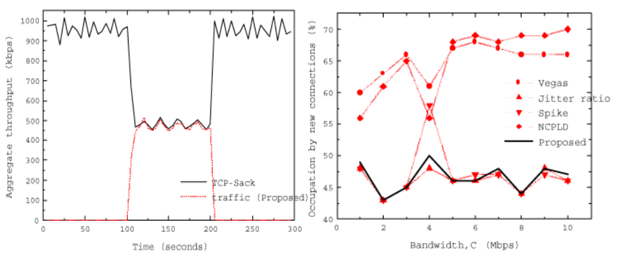
(c) RTT for $p_w = 0.1\%$ (d) Bandwidth for $p_w = 0.1\%$

Fig. 12. Improvement in TCP throughput as varying p_w, T_p and C

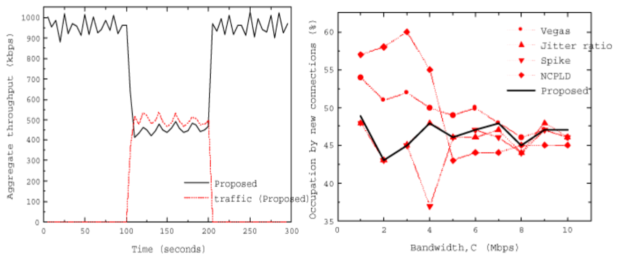
Friendliness

Friendliness is another important property of a TCP protocol. TCP with loss differentiation (LDA) must be "friendly" to other TCP variants. That is, LDA connections should not result in starvation of connections running other TCP variants.

Table 1 shows the throughput comparisons between LDA and TCP sharing the same path. The scenario is that n LDA connections and n TCP connections coexist on the same path. For only congestion losses ($p_w = 0\%$), the spike and the proposed scheme are best friendly with TCP connections. In particular, the throughput difference between proposed scheme and TCP connections is below 3%. However, the difference for the Jitter ratio is higher than 300%. The Vegas predictor occupies the bandwidth about 70% more than TCP connections when 20 of total connections are established (10:10). The proposed scheme can coexist with TCP connections fairly when the link is fully utilized (10:10) although p_w increases.



(a) (b)

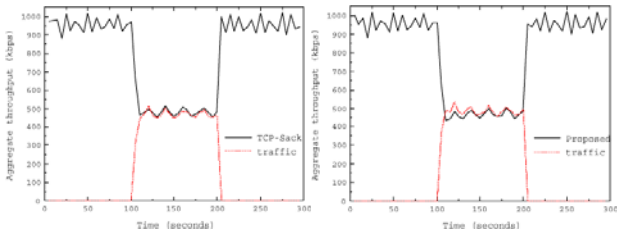


(c) (d)

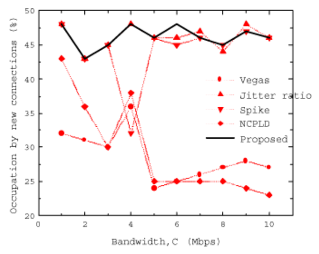
Fig. 13. Link occupation by new connections running LDA during lifetime on the loaded path (a) and (b) when old connections run TCP-SACK, and (c) and (d) when they run LDA

4. Conclusion

We propose a simple loss differentiation scheme to improve the accuracy through precise inference of the cause of packet loss. The proposed scheme is an end-to-end scheme which is easy to implement because it uses information readily available to TCP. We evaluated our differentiation scheme under various network conditions, including different network configurations, workload, error rates of the wireless medium, and different wireless error models. Accuracies of loss differentiation of other schemes proposed previously depend on network parameters. They favor long length of buffer, short propagation delay, low wireless loss rate and small number of connections on the bottleneck. However, the moving threshold enables proposed scheme to be accurate in correctly detecting congestion loss without favorable parameters. We found that our differentiation scheme is more stable under varying network conditions than vegas predictor, jitter ratio, NCPLD and spike scheme.

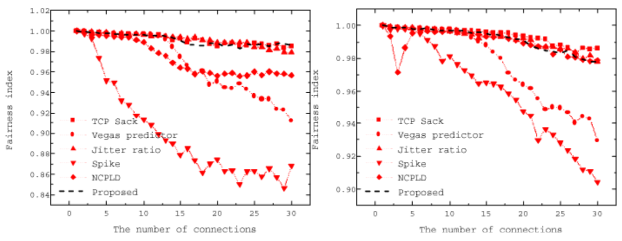


(a) (b)



(c)

Fig. 14. Link occupation by new connections running TCP-Sack during lifetime versus bandwidth



(a) $p_w = 1\%$ (b) $p_w = 3\%$

Fig. 15. Fairness index

References

[1] C. P. Fu and S. C . Liew, "TCP Veno: TCP Enhancement for Transmission Over Wireless Access Networks,", IEEE Journal on Selected Areas in Communications, vol. 21, pp216-228, Feb. 2003.

[2] Eric H.K. Wu and Mei-Zhen Chen, "JTCP: Jitter-Based TCP for Heterogeneous Wireless Networks", IEEE Journal on Selected Areas in Communications, Vol. 22, No. 4, pp757-766, May 2004.

[3] S. Cen, P. C. Cosman and G. M. Voelker, "End-to-end Differentiation of Congestion and Wireless Losses", IEEE/ACM Transactions on Networking, Vol. 11, No. 5, Oct. 2003, pp703-717.

[4] N. Samaraweera, "Non-congestion packet loss detection for TCP error recovery using wireless links", In IEEE Proceedings of Communications, Aug. 1999.

[5] Kevin Brown and Suresh Singh, “M-TCP: TCP for Mobile Cellular Networks”, ACM SIGCOMM 97, Cannes, France, July. 1997.

[6] Ajay Bakre and B. R. Bandrinath, “I-TCP: Indirect TCP for Mobile Hosts”, Proceedings of the 15th International Conference on Distributed Computing Systems (ICDCS '95), pp 136~143, Vancouver, May, 1995.

[7] Hari Balakrishnan, Srinivasan Seshan, and Randy H. Katz, “Improving Reliable Transport and Handoff Performance in Cellular Wireless Networks”, In ACM Wireless Networks Journal, 1(4), Dec. 1995.

[8] Sumitha Bhandarkar, Nauzad Erach Sadry, A.L. Narasimha Reddy, and Nitin H. Vaidya, "TCP-DCR: A Novel Protocol for Tolerating Wireless Channel Errors", IEEE Transactions on Mobile Computing, Vol. 4, No. 5, pp517-529, Sep. 2005.
[9] NS-2 network simulator (ver 2). LBL, http://www-mash.cs.berkeley.edu/ns
[10] R. Jain, D.M.Chiu and W.Hawe, "A Quantitative Measure of Fairness and Discrimination of Resource Allocation in Shared Systems", Technical Report, Digital Equipment Coporation, DEC-TR-301, 1984.
[11] Saad Biaz and Nitin H. Vaidya, "Distinguishing Congestion Losses from Wireless Transmission Losses : A Negative Result", In IEEE 7th Int'l Conf. on Computer Communications and Networks, Lafayette, LA, pp390-397, October 1998.
[12] Saad Biaz and Nitin H. Vaidya, "Is the Round-trip Time Correlated with the Number of Packets in Flight?", Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement, pp273-278, 2003.
[13] Saad Biaz and Nitin H. Vaidya, "'De-randomizing Congestion Losses' to Improve TCP Performance over Wired-Wireless Networks", ACM/IEEE Transactions on Networking, Vol. 13, No. 3, June 2005, pp596-608.
[14] C. Barakat and E. Altman, "Bandwidth Tradeoff between TCP and Link- level FEC", Computer Networks, 39, 2002, pp. 133-150.
[15] Guang Yang, Ren Wang, Mario Gerla and M. Y. Sanadidi "TCPW Bulk Repeat", Computer Communications(Elsvier), Vol. 28, Issue 5, Mar. 2005, pp507-518
[16] D. Aguayo, J. Bicket, S. Biswas, G. Judd and R. Morris, "Link-level Measurement from an 802.11b Mesh Network", ACM SIGCOMM 2004, Aug. 2004, Portland, Oregon, USA
[17] A. Gurtov and S. Floyd, "Modeling Wireless Links for Transport Protocols", ACM SIGCOMM Computer Communications Review, Vol. 34, Issue 2, April 2004, pp85-96
[18] Frank Kelly, "Mathematical Modeling of the Internet", Proceedings of the Fourth International Congress on Industrial and Applied Mathematics, 1999, pp105-116

Table 1: TCP friendliness using mean throughput comparisons between n LDA connections and n TCP connections on the same path. (VP: vegas predictor, Jr: jitter ratio. Unit:kbps)

Flows (n:n)		1:1		5:5		10:10	
P_w	Scheme	LDA	TCP	LDA	TCP	LDA	TCP
0%	VP	941	974	194	189	122	70
	Jr	1616	284	312	68	154	36
	Spike	941	974	194	189	97	95
	Ncpld	989	926	239	146	120	72
	Proposed	941	974	194	189	97	95
1%	VP	886	571	194	186	110	80
	Jr	969	567	280	97	146	42
	Spike	1064	539	232	148	116	74
	Ncpld	651	578	200	180	103	86
	Proposed	970	568	202	178	99	91
5%	VP	327	193	211	151	99	83
	Jr	327	198	223	137	118	63
	Spike	320	195	227	136	111	71
	Ncpld	237	196	191	162	92	90
	Proposed	320	194	204	156	99	83