Robust end-to-end loss differentiation scheme for transport control protocol over wired/wireless networks

C.-H. Lim and J.-W. Jang

Abstract: The authors propose a robust end-to-end loss differentiation scheme to identify the packet losses because of congestion for transport control protocol (TCP) connections over wired/wireless networks. The authors use the measured round trip time (RTT) values to determine whether the cause of packet loss is because of the 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 the 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 the 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 because of the regular bit errors over the 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 the congestion loss not to miss any congestion loss is decreased and hence improves the TCP fairness. In ns 2 simulations, the proposed scheme correctly classifies the congestion losses under varying network parameters whereas the previous schemes show some dependency on subsets of parameters.

1 Introduction

Transport control protocol (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 because of the packet loss from the bit errors over wireless part even though there is little congestion.

Research on improving performance of the TCP over hybrid wired/wireless paths has focused on differentiating packet losses using information readily available to the TCP: congestion window size, inter-arrival time between ACK packets and changes in a round-trip time (RTT) [1–4]. However, correct classification based on these metrics has been found to be difficult [2]. The reason is that the nature of losses is weakly correlated with the observable metrics (RTT, congestion window size, inter-arrival of ACKs and Jitter) [5]. In [6], 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 the packet loss because of the wireless channel errors.

We propose a new end-to-end scheme to precisely infer the nature of packet losses over the wired/wireless networks. (From now on, we call packet losses because of the congestion as congestion losses and the wireless channel errors as 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 the congestion loss if this value exceeds the threshold or classify it as the 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 the congestion loss when the network is congested whereas 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 from previous schemes[1–4, 6]. The accuracy of correctly detecting wireless loss on unloaded path is similar to previous schemes as far as the congestion loss rate is kept low.

The rest of this paper is organised 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.

doi:10.1049/iet-com:20060172

Paper first received 1st April 2006 and in revised form 13th May 2007

The authors are with the Department of Electronic Engineering, Sogang University, Shinsoo 1, Mapo, Seoul, South Korea

C.-H. Lim is now with Telecommunication Research Center, Samsung Electronics Maetan-dong 416, Yeongtong-gu, Suwon-si, Gyeonggi-do, Korea

E-mail: changhyeon.im@samsung.com

2 Related work

Known schemes to improve the TCP throughput over the 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 [7, 8]. At link layer, they can be purely local or aware of the semantics of the TCP protocol [9]. In an explicit loss notification approach, the receivers/network routers mark the acknowledgments with an appropriate notification of distinguishing the channel errors from the congestion losses [10]. 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 because of the congestion in the network from the other random losses over wireless paths. They can be deployed easily via simple modification to the TCP congestion control at the sender or receiver side [1–4, 6]. Our scheme also falls into end-to-end schemes.

The Vegas predictor [1, 2] estimates the outstanding packets over the network from the difference between the expected and the actual bandwidth. Biaz and Vaidya [2] 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.

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

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

The spike [4] is a scheme based on RTT measurement. In this work, we use RTT instead of relative one-way trip time. It keeps track of the minimum and maximum RTT values and computes thresholds $B_{\text{spikestart}}$ and B_{spikeend}. The two thresholds are determined as some values between the minimum and maximum RTT values where $B_{\text{spikestart}} > B_{\text{spikeend}}$. TCP enters the spike state if RTT exceeds $B_{\text{spikestart}}$, but it exits the spike state if RTT drops below B_{spikeend}. In the spike state, all the packet losses are regarded because of the congestion. Recently, TCP Westwood+ with bulk repeat (TCPW-BR) [11] employs this scheme to differentiate the cause of packet losses. The spike scheme has two thresholds $B_{\text{spikestart}}$ and B_{spikeend} are defined as:

$$\begin{aligned} B_{\text{spikestart}} &= \text{BaseRTT} + \alpha \cdot (\text{MaxRTT} - \text{BaseRTT}) \\ B_{\text{spikeend}} &= \text{BaseRTT} + \beta \cdot (\text{MaxRTT} - \text{BaseRTT}) \end{aligned} \tag{2}$$

where MaxRTT is the maximum of measured RTTs. The parameters α and β affect the sensitivity and aggressiveness of spike. In TCPW-BR, the setting for α and β in [11] is 0.4 and 0.05, respectively. In our simulations, this setting regards the most of packet losses because of congestion. That is, the accuracy of the congestion classification is almost 100%, but of wireless classification is about 0%. Therefore we choose the setting in [4]. α and β are set to $1/2$ and $1/3$, respectively.

Non-congestion packet loss detection (NCPLD) [3] is to find the knee point in the 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 the packet losses as the congestion losses.

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

$$\text{TotalPipeSize} = \frac{1}{2} \cdot T_k \cdot \frac{\text{cwnd}_k - \text{cwnd}_{k-1}}{\text{T}_k - \text{T}_{k-1}}$$

where T_k is the round-trip time measured on recept of the kth 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 obtain an estimate of the transmission rate B_{TX}. Both estimates yield the current value of T at the knee-point as presented by (3). The NCPLD scheme regards the packet loss because of the congestion, if the current T is greater than T_{kp}.

$$T_{\text{kp}} = \text{BaseRTT} + \frac{1}{2} \cdot T \cdot \frac{B_{TX} \cdot \text{BaseRTT}}{\text{TotalPipeSize}} \tag{3}$$

These loss differentiators estimate the change of the queueing delay to detect the packet loss because of the congestion, which occur 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, favourable values of network conditions are as follows: small RTT, large router queue size and small input bandwidth to the bottleneck [2].

In jitter-based TCP scheme (JTCP) [6], a TCP sender measures jitter ratio from the packet-by-packet inter-arrival time at the receiver side and the sending interval for one RTT. It requires the timestamp option in the TCP header. The jitter ratio (J_{r}) can be written as following

$$J_{\text{r}} = \frac{(t_{\text{r}}(i) - t_{\text{r}}(i - \text{cwnd})) - (t_{\text{s}}(i) - t_{\text{s}}(i - \text{cwnd}))}{t_{\text{r}}(i) - t_R(i - \text{cwnd})} \tag{4}$$

where $t_{\text{r}}(i)$ is the arrival time for ith packet at a receiver and $t_{\text{s}}(i)$ the sending time for ith packet at a sender. JTCP regards a packet loss because of the 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 [6]. This scheme improves the TCP throughput by improving the accuracy of correctly detecting the wireless losses.

According to Biaz and Vaidya [11], the throughput improvement with loss differentiation is derived approximately as follows

$$\begin{aligned} \text{Imp} &= 100 \times \left(\frac{\text{Thgt}_{\text{LDA}}}{\text{Thgt}_{\text{TCP}}} - 1\right) \\ &= 100 \times \left(\sqrt{1 + \frac{p_{\text{c}} + p_{\text{w}}}{A_{\text{c}} \cdot p_{\text{c}} + (1 - A_{\text{w}}) \cdot p_{\text{w}}}} - 1\right) \end{aligned} \tag{5}$$

where 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. 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. They consider that the receipt of triple-duplicate ACKs in the TCP congestion control is only modified. From this model, the TCP throughput with loss differentiation can

be improved by ignoring the some congestion loss as well as the wireless loss.

However, a good accuracy of correctly detecting congestion loss is important to help the TCP flows to reduce unnecessary congestion losses and to fairly share the bottleneck link between the TCP flows. Basically, the congestion control algorithm for the TCP achieves fairness because of an additive increase and multiplicative decrease. Loss differentiation is involved when a packet is lost and makes a TCP sender to halve the congestion window conditionally [1, 6, 11]. An inaccurate loss differentiation for the congestion loss can ignore the congestion loss frequently by making a TCP sender to increase its congestion window. Congestion losses therefore happen more frequently. This situation can affect other flows' congestion control. If other flows are a kind of TCP variants(TCP SACK, etc.), they would be expected to suffer from ignoring the congestion losses by the inaccurate loss differentiation. If there are homogeneous flows with the inaccurate loss differentiation on a shared link, they would be expected to behave to frequently ignore the congestion losses and then they would make more congestion loss.

3 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 the loss differentation.

3.1 Network model and observation

Fig. 1 shows a heterogeneous network model of the wired and wireless paths to be used in the following description of our scheme. p_c denotes the probability of packet loss because of the congestion in the wired path whereas p_w denotes an 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 $B{\cdot}S/C$. The RTT(T) can be written as (6).

$$T = T_p + \frac{B \cdot S}{C} \tag{6}$$

Fig. 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 denote the congestion window size of ith TCP flow out of the n flows sharing the path. When the buffer occupancy nears overflow, we have $T \gg T_p$ and the probabilty of the congestion loss (p_c) becomes increasingly high.

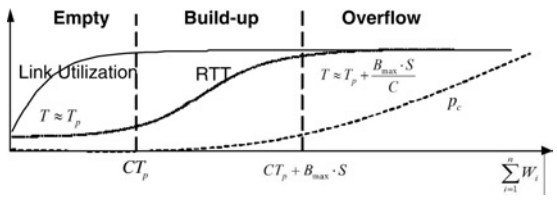

Fig. 1 *Three states of network buffer and the change of the RTT with the aggregation of congestion windows of individual flows*

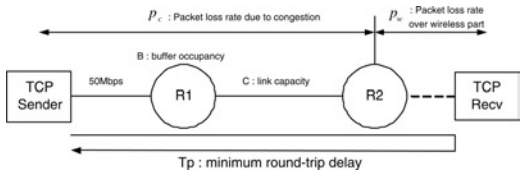

Fig. 2 *Heterogeneous network model of wired/wireless paths*

3.2 Proposed scheme

We present a new robust scheme to infer the cause of each packet loss encountered whether it is because of congestion or not. The RTT(T) measured immediately before the current packet loss is used in (7) as an indicator. $\bar{T}$ and T_{dev} are an EWMA of RTT(T) and the deviation, respectively. They are updated by $\bar{T} = (7/8)\bar{T} + (1/8)T$ and $T_{dev} = (3/4)T_{dev} + (1/4)|T - \bar{T}|$ as in most TCP implementations. The current packet loss is determined to be a congestion loss if (7) is satisfied. Fig. 3 illustrates the decision rule represented by (7).

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

This threshold depends on two parameters: k, and T_p. A constant k can be chosen for the proposed differentiation to be accurate. T_p can be one of the criteria to infer the cause of packet loss. We first find effective k for the differentiation to be accurate via simulations. We then investigate the impact of an overestimated T_p on the threshold T_0 of the cross point of $(T - \bar{T})/T_{dev}$ and $2\ (T_p/T)^k - 1$.

3.2.1 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 Fig. 2. p_c is the ratio of the number of the 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. The threshold with $k \geq 4$ can be below 0 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 Fig. 4, A_c begins to saturate as k becomes greater than 3. When

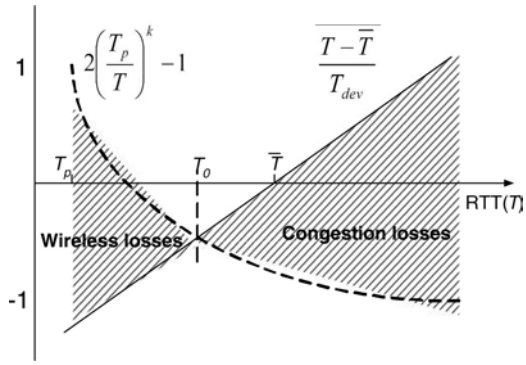

Fig. 3 *Decision rule*

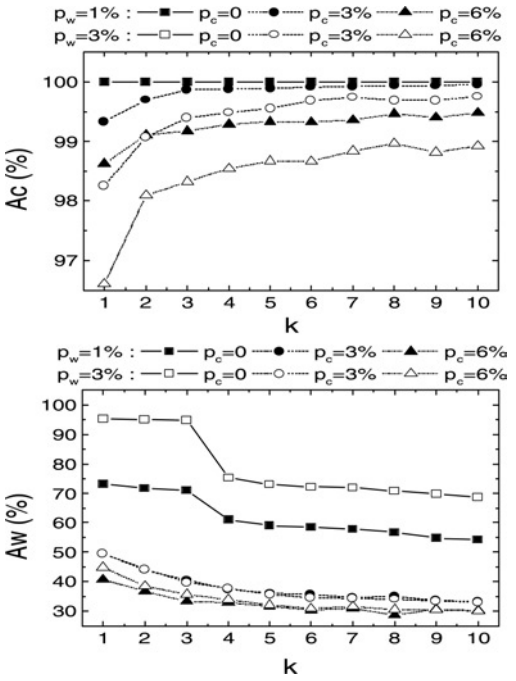

Fig. 4 *Impact of k on accuracies of loss differentiator over the 2 Mbps wireless link with the buffer size of 100% BDP*

there is no congestion ($p_c = 0$), A_w drops sharply as k goes higher than 3.

We extend to the simulation of accuracies over recent high-speed networks of 50 and 100 Mbps to find the optimal k. The buffer size is set to 20% bandwidth-delay product (BDP), which is chosen by [12, 13]. In high-speed networks, the buffer size at the router needs to be minimum 20% BDP because congestion losses can happen more frequently at the buffer with the size to be below 20% BDP [12]. Fig. 5 shows that A_c is shown to be over 97% for $k \geq 2$ over 50 and 100 Mbps link with buffer size to be 20% BDP.

From the two results as shown in Figs. 4 and 5, we choose a coefficient k to be 3.

3.2.2 Impact of overestimated T_p on threshold: A measured T_p can be overestimated because of 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 of an overestimated T_p on the threshold T_0 of T. 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 Fig. 6, the proposed scheme restricts the error of decision boundary T_0 within 25% even though T_p is overestimated over 100%. As C increases, the range which the RTT fluctuates is reduced because the queueing delay decreases. Then, $\bar{T}$ and T_{dev} also decrease. T_p measured by new connections can be as same as T_p of old connections for larger C.

4 Performance evaluation

In this section, we evaluate the accuracy of our scheme under various network conditions. There are two accuracies for loss differentiation, accuracy of correctly detecting the congestion loss and accuracy of correctly detecting wireless loss. We observe the impact of the wireless loss rate over the wireless link, the length of bottleneck link buffer, end-to-end propagation delay, the link capacity and packet loss model on accuracies of the proposed scheme.

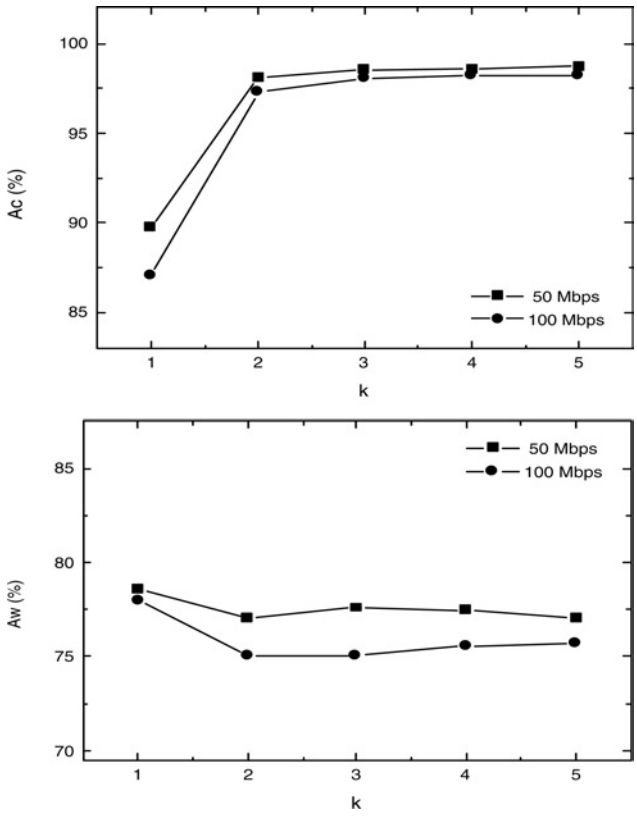

Fig. 5 *Impact of k on accuracies of loss differentiator over high-speed link of 50 and 100 Mbps with the buffer size of 20% BDP*

4.1 Experimental setup

We evaluate the performance of the proposed scheme via ns-2 (Ver. 2.27) [14] simulation. The network model is shown in Fig. 7. The bandwidth C is set to 2 Mbps. The size of the buffer B_{max} is set to 50 packets. The propagation delay T_p is set to 140 ms. We set the packet size equal to 1000 bytes. We extend our simulation to high-speed networks of 50 and 100 Mbps link and set B_{max} to 20% and 100% BDP, which are a fraction of BDP.

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/

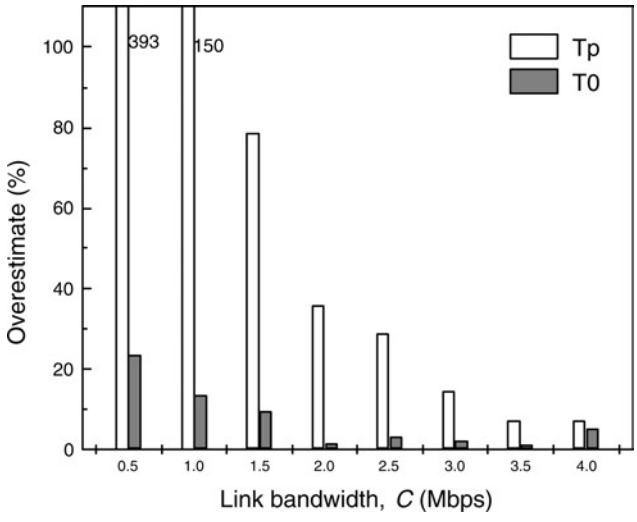

Fig. 6 *Overestimate of T_p and T_0 in new connections for $k = 3$ as varying C*

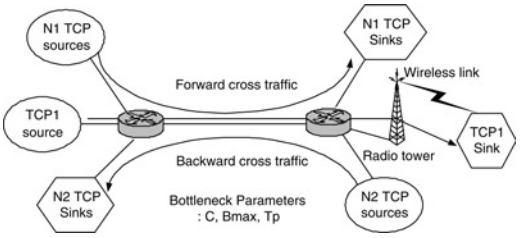

Fig. 7 *Mixed wired/wireless scenario*

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. The RTTs of the $N1$ TCP forward cross traffic connections and those 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 $N1$ and $N2$ increases.

We run simulations 30 times. The simulation time for each run is set to 1000 s. We estimate the average values of A_c and A_w in the each of the 30 runs.

4.2 Results

We present results of comparison with accuracies of existing LDAs such as Vegas predictor [1, 2], jitter ratio [6], spike [4] and NCPLD [3]. We mainly focus on the dependency of A_c on p_c, p_w, B_{max}, T_p, C and wireless packet loss model because A_w is important to achieve higher TCP throughput only if there is no congestion.

4.2.1 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 A_c and A_w. The wireless link is modelled with

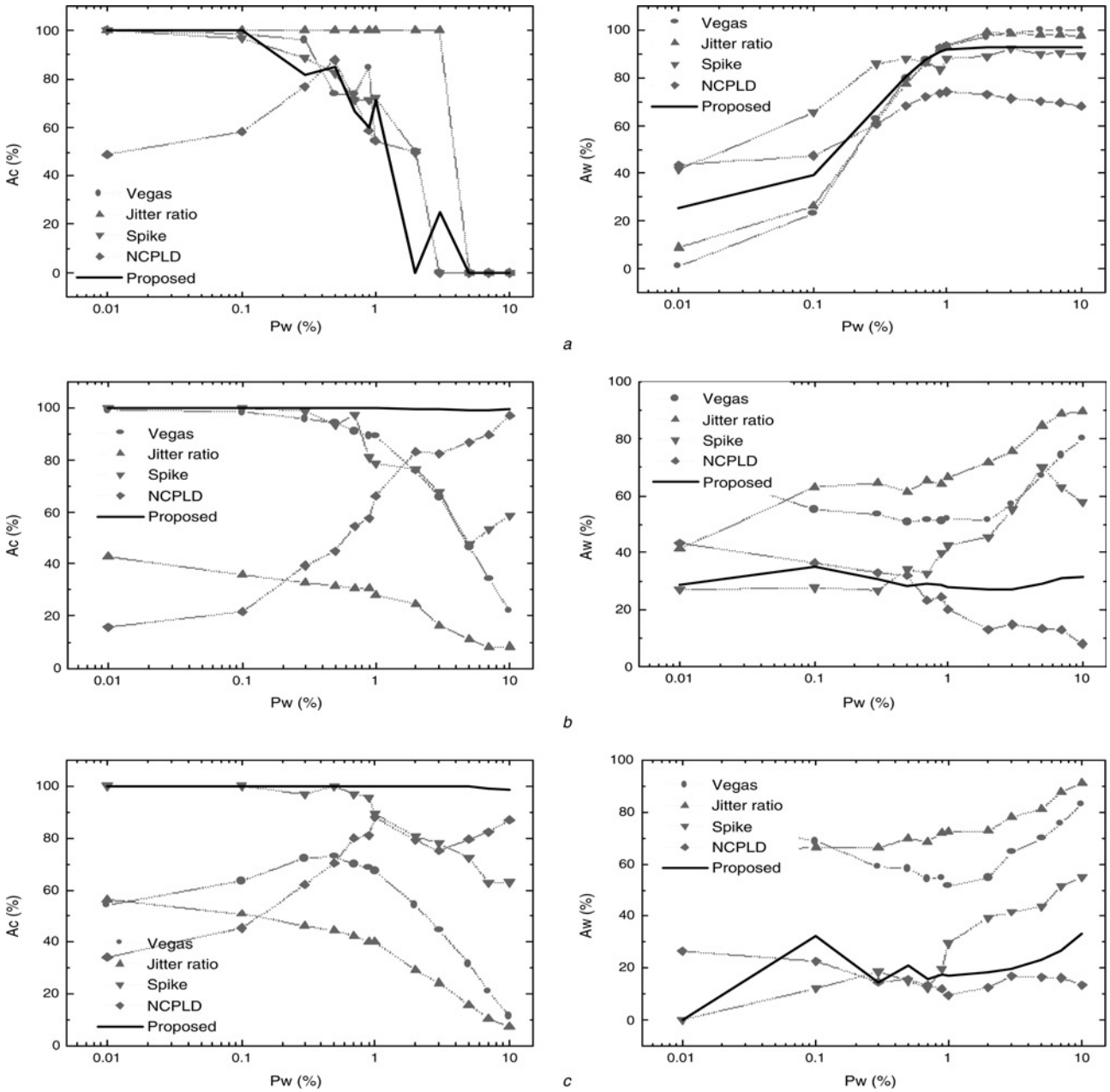

Fig. 8 *Accuracies A_c and A_w*

a No cross traffic (N1 = 0)
b $p_c = 3\%$ ($N1 = 15$)
c $p_c = 5\%$ ($N1 = 24$)

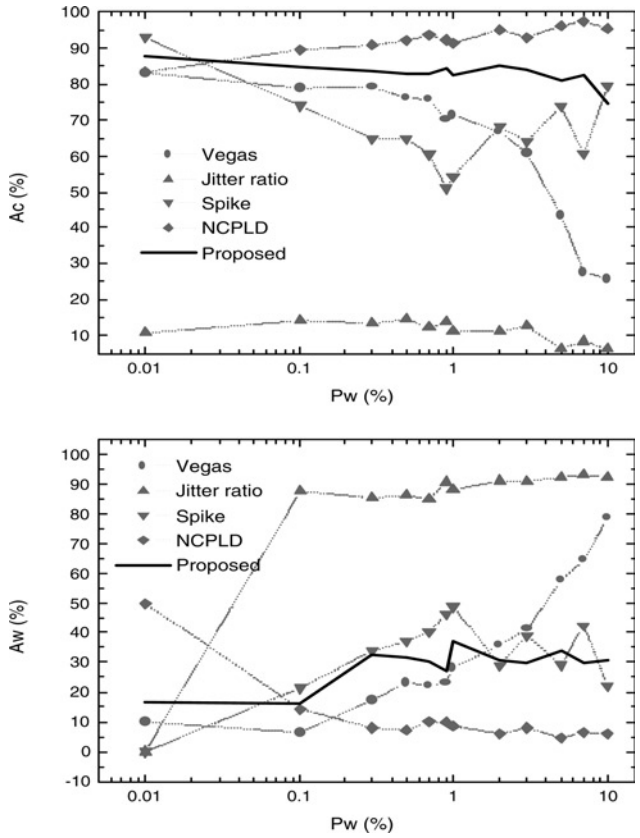

Fig. 9 *Accuracies A_c and A_w with coexistence of backward cross traffic*

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 Fig. 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. Figs. 8*b* and *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, 2] and Spike [4] exhibit high A_c for low p_w and low A_c for high p_w, whereas NCPLD [3] shows low A_c for low p_w and high A_c for high p_w. Especially, Vegas predictor [1, 2] shows a dependency of A_c on not only p_w but also p_c since A_c is lowered by the increase of p_c. Fig. 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 [3] excels all other schemes. Our scheme shows second best performance.

4.2.2 Impact of B_{max} and T_p on A_c and A_w: In this scenario, we aim to study the effect of buffer size on accuracy. First, the buffer size, B_{max}, is changed when BDP is fixed to be 35 packets. We set the number of forward cross traffic connections $N1$ to 6 and p_w to 1%. Fig. 10*a* shows that the proposed scheme needs small length of network buffer to precisely detect the congestion loss than the Vegas predictor[1, 2]. Fig. 10*a* shows that A_c for the proposed scheme nears 100% at $B_{max} = 10$, whereas the A_c for the Vegas predictor [1, 2] reaches about 85% when B_{max} is as high as 35 packets equal to 100% BDP. Spike [4] and NCPLD [3] require the large B_{max} to achieve higher A_c.

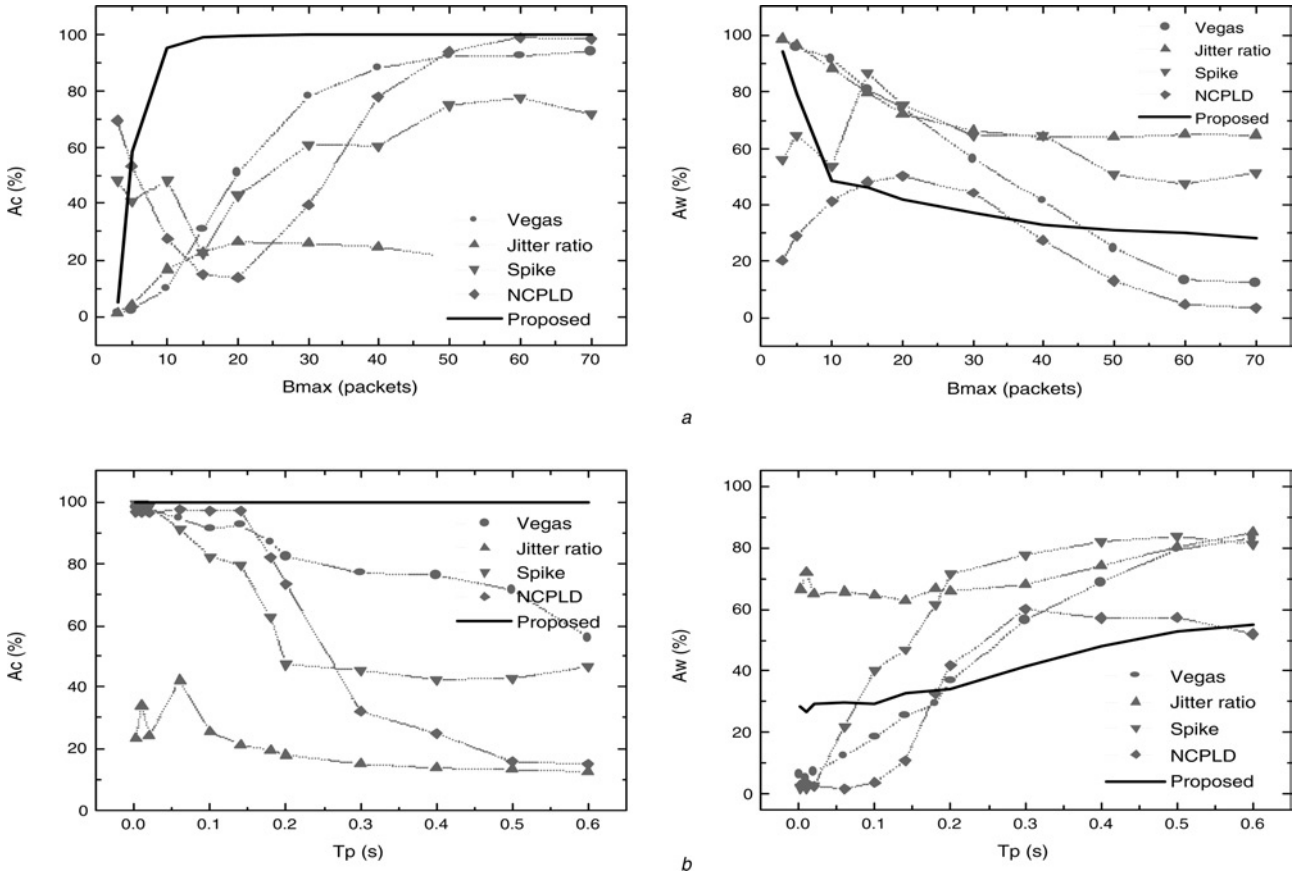

Fig. 10 *Accuracies A_c and A_w as varying B_{max} upto 200% BDP and T_p from 10 to 600 ms*

a Accuracies against B_{max} (BDP is 35 packets)
b Accuracies against T_p (for $T_p = 200$ ms, BDP is equal to 50 packets B_{max}

We then investigate the impact of T_p on accuracies. T_p ranges from 20 to 600 ms. The BDP for $T_p = 200$ ms becomes to be equal to the fixed 50 packets B_{max}. For $T_p = 600$ ms, BDP is three times larger than B_{max}. Fig. 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, 2], Spike [4] and NCPLD [3] decrease as T_p increases. The Jitter ratio [6] is poor at congestion loss, but good at wireless loss.

4.2.3 Impact of the bottleneck bandwidth, C, on A_c and A_w: In this scenario, we differently set the number of forward cross traffic connections $N1$ to make congestion loss ratio to be similar to each case of bandwidth. For $C = 2$ and 10 Mbps, $N1$ is set to 5 and 15, respectively. For $C = 50$ and 100 Mbps, $N1$ is set to 25 and 50, respectively. T_p is set to 100 ms. The buffer size, B_{max}, has two cases of 20 and 100% BDP. As described in the above Section, we choose a small buffer size as 20% BDP and a large buffer size as 100% BDP according to [12, 13]. p_w is set to 0.5%. The bottleneck bandwidth, C, is varied from 2 to 100 Mbps.

Fig. 11 shows that A_c of the proposed scheme is close to 100% with buffer size of 100% BDP, whereas keeping the A_w of our scheme as high as 30–40%. For Vegas predictors [1, 2], A_c is shown to be over 95%, but A_w is shown to be under 15% for $C \geq 10$ Mbps [2]. Jitter ratio [6] and spike [4] regard most packet losses as wireless loss. NCPLD [3] is outperformed by the proposed scheme.

For buffer size as small as 20% BDP, the result is shown in Fig. 12. A_c of the proposed scheme outperforms other schemes for $C \geq 10$ Mbps, whereas A_w is kept at 70–80%. Better performance with smaller buffer size can be explained by reduced queue build-up. Other schemes, except NCPLD [3], tend to regard most of packet losses as wireless losses, and thus A_cs of those schemes go as low as below 20%.

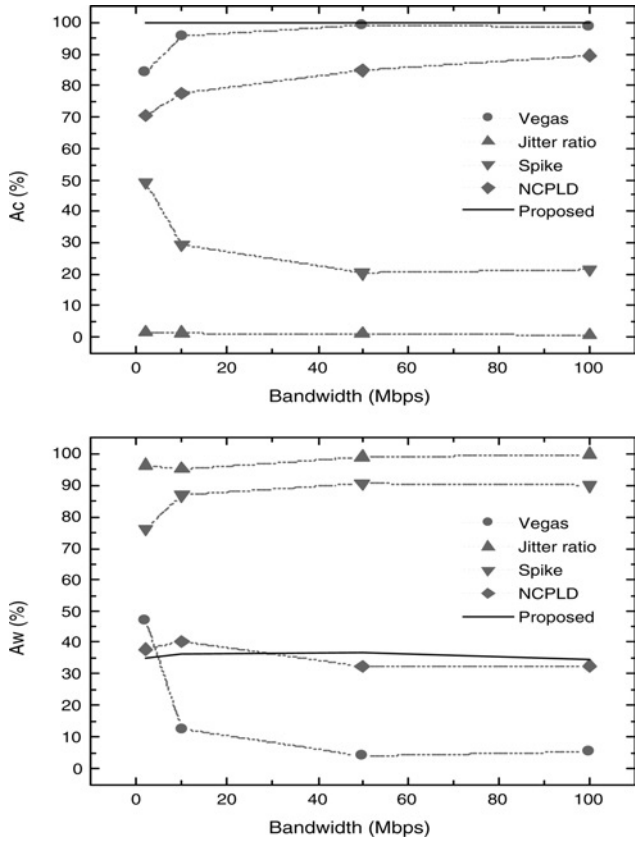

Fig. 11 *Accuracies A_c and A_w as varying the bottleneck bandwidth with $B_{max} = 100\%$ BDP*

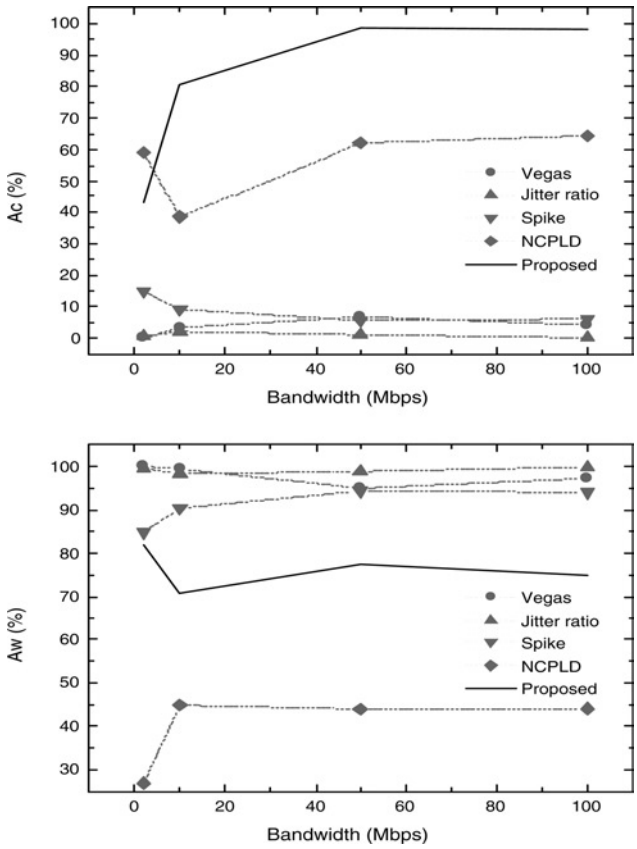

Fig. 12 *Accuracies A_c and A_w as varying the bottleneck bandwidth with $B_{max} = 20\%$ BDP*

4.2.4 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 [15, 16]. In particular, we assume that p_w is equal to 0.1%, when the channel is in good state, and is equal to 10% when the channel is in bad state as shown in Fig. 13. The duration in the good state is assumed deterministic and is equal to 1 s, whereas the duration in the bad state is assumed also deterministic, but this time we consider values ranging from 0.1 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 modelling the packet loss model of IEEE 802.11 wireless LAN [16].

Fig. 14 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 Fig. 14*a*, A_c for long duration of bad state is meaningless because of lack of loss events, but the A_c for NCPLD [3] drops under 70% when congestion loss is dominant. In Fig. 14*b*, A_c for other schemes depends on the duration of bad state. A_c for proposed scheme stays close to 100%.

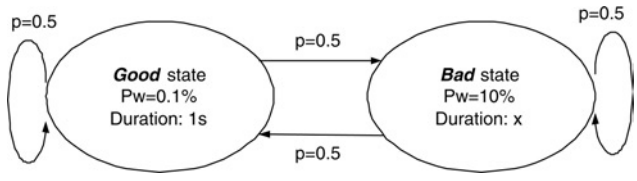

Fig. 13 *2-state Markov chain packet loss model*

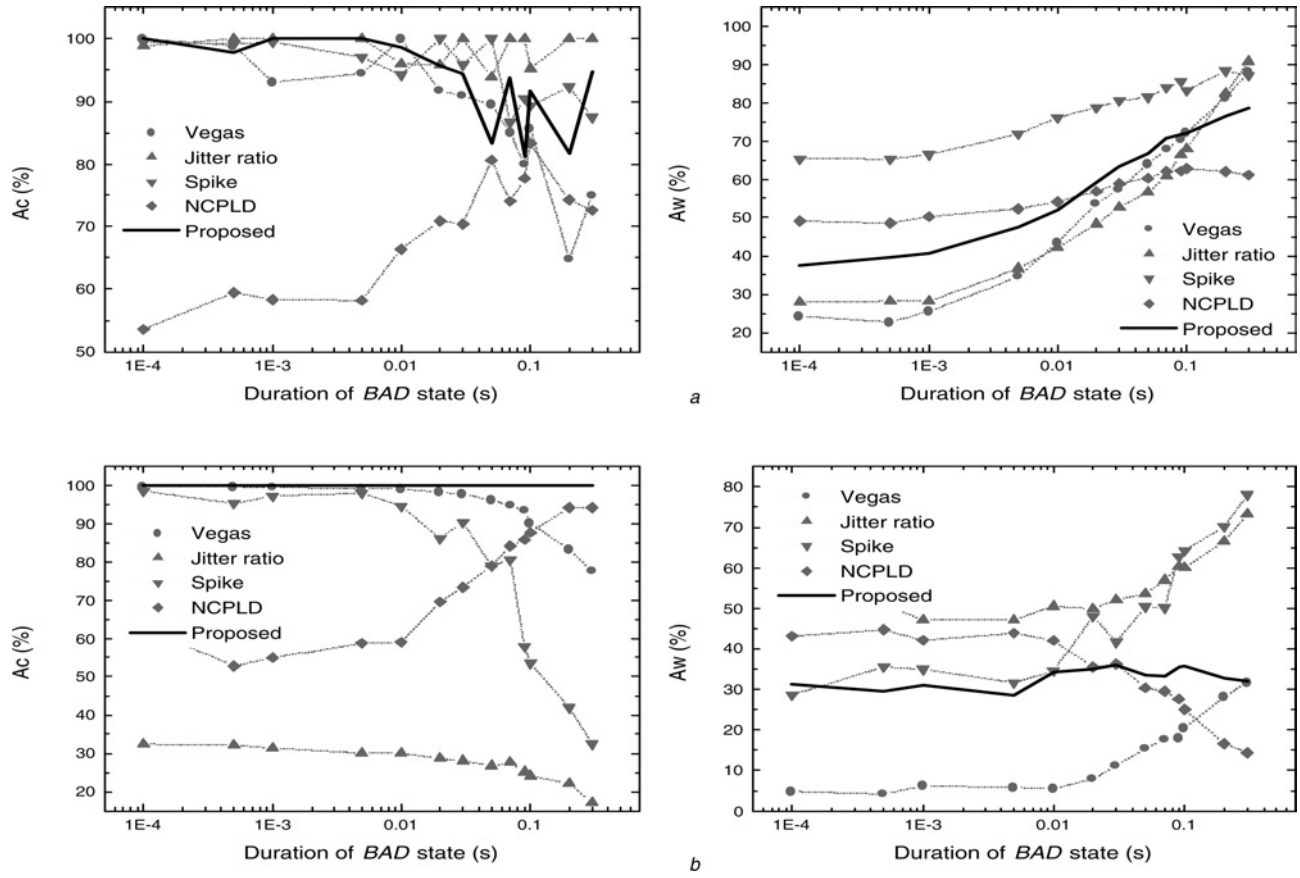

Fig. 14 *Accuracies A_c and A_w with correlated packet loss model*
a No cross traffic ($N1 = 0$)
b $p_c = 1\%$ ($N1 = 6$)

5 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 the 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 favour 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 favourable parameters. We found that our differentiation scheme is more stable under varying network conditions than Vegas predictor [1, 2], jitter ratio [6], NCPLD [3] and spike [4] scheme.

6 References

1 Fu, C.P., and Liew, S.C.: 'TCP veno: TCP enhancement for transmission over wireless access networks', *IEEE J. Sel. Areas Commun.*, 2003, **21**, pp. 216–228
2 Biaz, S., and Vaidya, N.H.: 'Distinguishing congestion losses from wireless transmission losses: a negative result'. IEEE 7th Int. Conf. Computer Communications and Networks, Lafayette, LA, October 1998, pp. 390–397
3 Samaraweera, N.: 'Non-congestion packet loss detection for TCP error recovery using wireless links'. IEEE Proc. Communications, August 1999
4 Cen, S., Cosman, P.C., and Voelker, G.M.: 'End-to-end differentiation of congestion and wireless losses', *IEEE/ACM Trans. Netw.*, 2003, **11**, (5), pp. 703–717
5 Biaz, S., and Vaidya, N.H.: 'Is the round-trip time correlated with the number of packets in flight?'. Proc. 3rd ACM SIGCOMM Conf. Internet Measurement, 2003, pp. 273–278
6 Wu, E.H.K., and Chen, M.-Z.: 'JTCP: jitter-based TCP for heterogeneous wireless networks', *IEEE J. Sel. Areas Commun.*, 2004, **22**, (4), pp. 757–766
7 Brown, K., and Singh, S.: 'M-TCP: TCP for mobile cellular networks'. ACM SIGCOMM 97, Cannes, France, July 1997
8 Bakre, A., and Bandrinath, B.R.: 'I-TCP: indirect TCP for mobile hosts'. Proc. 15th Int. Conf. Distributed Computing Systems (ICDCS '95), Vancouver, May 1995, pp. 136–143
9 Balakrishnan, H., Seshan, S., and Katz, R.H.: 'Improving reliable transport and Handoff performance in cellular wireless networks', *ACM Wirel. Netw. J.*, 1995, **1**, (4), pp. 469–481
10 Biaz, S., and Vaidya, N.H.: '"De-randomizing congestion losses' to improve TCP performance over wired-wireless networks", *ACM/IEEE Trans. Netw.*, 2005, **13**, (3), pp. 596–608
11 Yang, G., Wang, R., Gerla, M. *et al.*: 'TCPW bulk repeat', *Comput. Commun.*, 2005, **28**, (5), pp. 507–518
12 Barman, D., Smaragdakis, G., and Matta, I.: 'The effect of router buffer size on highspeed TCP performance'. IEEE Globecom, 2004
13 Li, Y.-T., Leith, D., and Shorten, R.N.: 'Evaluating the performance of TCP stacks for high-speed networks'. Proc. Protocols for Fast Long Distance Networks, Nara, Japan, 2006
14 NS-2 network simulator (ver. 2). LBL, available at: http://www-mash.cs.berkeley.edu/ns
15 Barakat, C., and Altman, E.: 'Bandwidth tradeoff between TCP and link-level FEC', *Comput. Netw.*, 2002, **39**, pp. 133–150
16 Gurtov, A., and Floyd, S.: 'Modeling wireless links for transport protocols', *ACM SIGCOMM Comput. Commun. Rev.*, 2004, **34**, (2), pp. 85–96