Available online at www.sciencedirect.com

SCIENCE DIRECT®

ELSEVIER

Signal Processing: *Image Communication* 19 (2004) 405–419

SIGNAL PROCESSING:
IMAGE
COMMUNICATION

www.elsevier.com/locate/image

TCP-friendly flow control of wireless multimedia using ECN marking☆

Seong-jun Bae*, Song Chong

Department of Electrical Engineering and Computer Science, Korea Advanced Institute of Science and Technology (KAIST), 373-1 Guseong-dong, Yuseong-gu, Daejeon 305-701, Republic of Korea

Received 30 July 2003; accepted 21 January 2004

Abstract

In a wireless network packet losses can be caused not only by network congestion but also by unreliable error-prone wireless links. Therefore, flow control schemes which use packet loss as a congestion measure cannot be directly applicable to a wireless network because there is no way to distinguish congestion losses from wireless losses. In this paper, we extend the so-called TCP-friendly flow control scheme, which was originally developed for the flow control of multimedia flows in a wired IP network environment, to a wireless environment. The main idea behind our scheme is that by using explicit congestion notification (ECN) marking in conjunction with random early detection (RED) queue management scheme intelligently, it is possible that not only the degree of network congestion is notified to multimedia sources explicitly in the form of ECN-marked packet probability but also wireless losses are hidden from multimedia sources. We calculate TCP-friendly rate based on ECN-marked packet probability instead of packet loss probability, thereby effectively eliminating the effect of wireless losses in flow control and thus preventing throughput degradation of multimedia flows travelling through wireless links. In addition, we refine the well-known TCP throughput model which establishes TCP-friendliness of multimedia flows in a way that the refined model provides more accurate throughput estimate of a TCP flow particularly when the number of TCP flows sharing a bottleneck link increases. Through extensive simulations, we show that the proposed scheme indeed improves the quality of the delivered video significantly while maintaining TCP-friendliness in a wireless environment for the case of wireless MPEG-4 video.

Keywords: Multimedia flow control; Wireless loss; ECN marking; RED; TCP-friendliness; MPEG-4 video

1. Introduction

With the rapid growth of emerging demand and deployment of wireless infrastructures like IMT-2000 networks and wireless LANs, much of IP traffic including multimedia traffic is forced to undergo wireless links. Such a change in networking environments brings us a necessity to refine conventional flow control schemes not only for non-real-time elastic traffic but also for real-time multimedia traffic in a way that the schemes can cope with unreliable error-prone wireless links [5]. In this paper we focus on issues in multimedia flow control with a particular

☆This research was supported in part by University IT Research Center Project.

*Corresponding author. Tel.: +82-42-8695473.

E-mail addresses: mythox@netsys.kaist.ac.kr (S.-j. Bae), song@ee.kaist.ac.kr (S. Chong).

0923-5965/$ - see front matter
doi:10.1016/j.image.2004.01.003

emphasis on video transmission over wireless IP networks.

The quality of networked multimedia applications is more sensitive to packet delay and delay jitter than packet loss because interruption in playback due to packet delay annoys users more seriously than degradation of picture quality due to packet loss. Therefore, most of multimedia applications use UDP as its transport layer protocol because UDP incurs no retransmission delay and jitter. However, UDP itself provides no flow control mechanism so that sources cannot adapt its transmission rate to time-varying available bandwidth which depends on network loading. Therefore, it is necessary to have an application-layer flow control scheme for the adaptive transmission of multimedia over UDP [2,4,14,16]. In order to facilitate this necessity, the RTP/RTCP protocol [15], which enables measurement and calculation of available network bandwidth, has been introduced.[1]

What is the available bandwidth for a multimedia flow in the Internet? The answer to this question seems not clear at first glance because underlying TCP flows are flow-controlled so that they inherently try to consume most out of available network bandwidth. However, from an inter-flow fairness standpoint, one can make a reasonable assumption to this problem that multimedia flows must consume the same bandwidth as TCP flows at least in a long-term average sense because much of the internet traffic is constituted by the TCP traffic [8]. This is the key idea behind the so-called TCP-friendly approach to multimedia flow control [2,4]. In the TCP-friendly flow control, multimedia flows send data at the long-term average rate of underlying TCP flows, which is estimated based on a TCP throughput model. Depending on assumptions, the throughput of a TCP flow can be modeled in different forms [7,10,11] but it is basically a function of packet loss probability P_L and round trip time R of the flow, that is,

$$Y = f(P_L, R), \tag{1}$$

where Y is the long-term average throughput of a TCP flow and is decreasing with respect to both P_L and R. This equation implies that each multimedia source must measure P_L and R along its path in order to send data at the same rate Y as TCP flows provided that the throughput function $f(P_L, R)$ is known.

In a wireless network, packet losses can be caused not only by network congestion but also by unreliable error-prone wireless links. Therefore, flow control schemes which use packet loss probability P_L as a congestion measure cannot be directly applicable to a wireless network because there is no way to distinguish congestion losses from wireless losses. The same is true for TCP-friendly flow control scheme based on the Eq. (1) since it uses P_L as the congestion measure. In order to get around this problem, we modify TCP-friendly flow control scheme with the help of explicit congestion notification (ECN) marking capability [13] in conjunction with random early detection (RED) queue management scheme [1] in a way that the new scheme uses ECN-marked packet probability P_M as a congestion measure instead of packet loss probability P_L.

Suppose that TCP flows are ECN-capable (we call them ECN–TCP flows) and react to ECN-echoed ACKs in the same way as they react to dropped packets, halving congestion window [13]. Then, the long-term average throughput of an ECN–TCP can be expressed by (1) by substituting P_{LM} for P_L where P_{LM} denotes the probability that a packet is either dropped or ECN-marked. If we further suppose that RED routers in the network are tuned appropriately such that no packet is lost during congestion at the routers and ECN–TCP flows do not undergo wireless links, then the long-term average of an ECN–TCP can be expressed by (1) by substituting P_M for P_L because no packet is lost and thus the ECN–TCP source will see only ECN-echoed ACKs. Therefore, the throughput of an ECN–TCP would be given by $f(P_M, R)$ if it experiences neither wireless loss nor congestion loss. In this end, if a multimedia flow can measure P_M as experienced by

[1] Alternatively, there are on-going activities to define a new transport-layer protocol called DCCP with congestion control capability for datagram flows [6].

concurrent ECN–TCP flows and its own R and send data at the rate $f(P_M, R)$ no matter it experiences wireless loss or not, its throughput would be as much as that of an ECN–TCP flow experiencing no loss, and be greater than that of an ECN–TCP flow experiencing loss. In order to enable a multimedia source to measure P_M and R, we adopt the RTP/RTCP protocol and propose both sender and receiver algorithms to measure them.

We also refine the well-known Floyd's TCP throughput model [7] to take into account the congestion-window dormant period (where congestion window size is kept constant) following each congestion window reduction in the congestion avoidance phase. We show that this dormant period plays a significant role as the number of concurrent TCP flows sharing a bottleneck link increases and the refined model taking this period into account predicts TCP throughput more precisely than Floyd's model particularly in such a situation.

2. TCP-friendly wireless multimedia flow control: TF-WMFC

2.1. ECN–TCP throughput model

Two widely used TCP-Reno throughput model are Padhye's model [11] and Floyd's model [7]. In fact Padhye's model is a generalization of Floyd's model, taking into account the timeouts incurred by consecutive packet losses, and provides a better throughput estimate particularly when the TCP flows travel through drop-tail routers. However, if the routers are equipped with RED queue management scheme, the probability of timeout occurrence due to consecutive packet losses falls dramatically off because of the randomization effect of RED [1]. For this reason, we use Floyd's model given below as the starting point of our study, assuming that routers in the network are equipped with RED queue management scheme.

$$Y = f(P_L, R) = \frac{K}{\sqrt{P_L}}\frac{s}{R}, \tag{2}$$

where Y is the long-term average throughput of a TCP-Reno flow, P_L and R denote packet loss probability and round-trip time of the flow respectively, s is maximum segment size of TCP and K is a constant resulting from the modeling. In our study we let $K = 1.22$ assuming periodic loss [7].

An ECN-capable RED router marks ECN-bit in incoming packets' IP header in a probabilistic manner when it detects congestion. At the same time an ECN–TCP reacts to these ECN-marked packets in the similar way it does to dropped packets, halving its congestion window. This implies that the long-term average throughput of an ECN–TCP would be estimated by (2) by substituting P_{LM} for P_L where P_{LM} denotes the probability that a packet is either dropped or ECN-marked. If we suppose that RED routers in the network are tuned appropriately such that no packet is lost during congestion at the routers and ECN–TCP flows do not travel through wireless links, then the long-term average of an ECN–TCP would be estimated by $f(P_M, R)$ using ECN-marked packet probability P_M instead of P_{LM} because no packet is lost and thus the ECN–TCP source will see only ECN-echoed ACKs. Therefore, we conclude that the throughput of an ECN–TCP would be given by $f(P_M, R)$ if it experiences neither wireless loss nor congestion loss. In the end, we conjecture that if a multimedia flow can measure P_M as experienced by concurrent ECN–TCP flows and its own R and send data at the rate $f(P_M, R)$ no matter it experiences wireless loss or not, its throughput would be as much as that of an ECN–TCP flow experiencing no loss and be greater than that of an ECN–TCP flow experiencing loss. Through simulations in Section 3, we show that this conjecture is true.

In the congestion avoidance phase of ECN–TCP, each congestion window reduction is followed by a dormant period where congestion window size is kept constant as shown in Fig. 1. Through simulations we found that this dormant period plays a significant role as the number of ECN–TCP flows multiplexed in a link increases and, moreover, the Floyd's throughput model (2) which does not take this period into account exhibits poor performance in such a situation. We

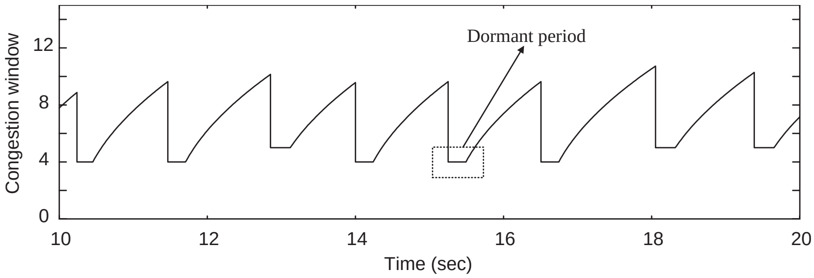

Fig. 1. Typical trace of congestion window in an ECN–TCP.

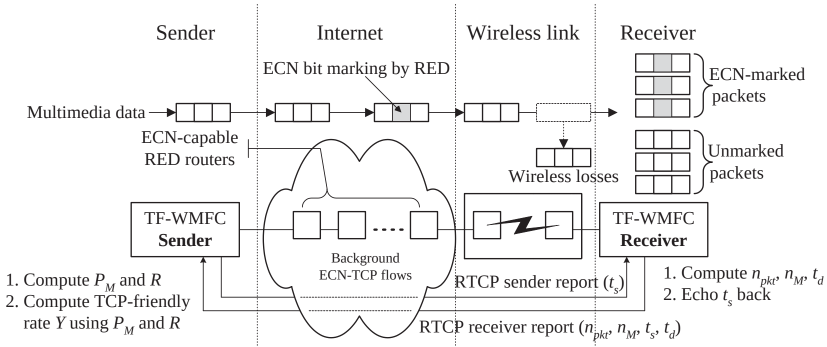

Fig. 2. Overview of ECN-based TF-WMFC.

consider this dormant period in modeling and obtain the following refined model:

$$Y = f(P_{\mathrm{M}}, R) = \frac{s}{P_{\mathrm{M}}(\sqrt{2/3P_{\mathrm{M}} + 25/36} + 7/6)R}. \quad (3)$$

This refined model predicts ECN–TCP throughput more precisely than the Floyd's model particularly as the number of ECN–TCP flows multiplexed in a link increases, as will be shown in Section 3. The derivation of the refined model and the detailed behavior of ECN–TCP is presented in the appendix.

2.2. Overview of ECN-based TF-WMFC

Fig. 2 shows the overall framework of proposed ECN-based TF-WMFC scheme. For the implementation of the proposed scheme, we adopt the RTP/RTCP protocol. The sender estimates P_{M} and R being experienced by its flow based on feedback information sent by the receiver and then computes the TCP-friendly rate Y using the ECN–TCP throughput model (3). On the other hand, the receiver keeps monitoring arriving packets, collects information including number of packets received and the number of ECN-marked packets received in a certain time interval and feeds this information back to the sender by sending a backward control packet called RTCP receiver report. For the estimation of R, the sender sends the receiver a forward control packet called RTCP sender report by stamping its current time on it and the receiver returns this timestamp to the sender via an RTCP receiver report. One of the key differences between ECN–TCP operation and ECN-based TF-WMFC operation is that in the former the ECN-marked packet is immediately echoed back to the sender via the corresponding ACK whereas in the latter the receiver collects the information on total number of packets received and total number of ECN-marked packets received in a certain interval and feeds this statistics back to the sender via an RTCP receiver report.

Routers in the network are assumed to be ECN-capable and equipped with RED queue management scheme, i.e., the routers mark the ECN bit on incoming packets in the forward direction with a certain probability depending on the level of congestion.

Packets can be corrupted in the wireless hop due to high bit errors as the channel status degrades.[2] For the sake of simplicity we assume in this paper that both ECN-marked packets and unmarked packets suffer wireless loss with the same probability so that the ECN-marked packet probability P_{M} remains unaffected as packets travel through the wireless hop.

In actual implementation, wireless hops employ link-level error recovery schemes like radio link control (RLC) [3] or forward error correction (FEC). These schemes incur additional delays such as retransmission delays, while reducing wireless losses. We assume that this link-level delay of wireless links is a part of the total round-trip time experienced by the flow.

2.3. Detailed behavior of sender and receiver

2.3.1. Sender behavior

Upon every epoch of Δ, the sender updates its TCP-friendly rate $Y[k]$ using the following equations:

$$Y[k] = \frac{s}{P_{\mathrm{M}}[k]\left(\sqrt{(2/3P_{\mathrm{M}}[k]) + (25/36)} + (7/6)\right)R[k]}, \tag{4}$$

$$P_{\mathrm{M}}[k] = (1-\alpha)P_{\mathrm{M}}[k-1] + \alpha P_{\mathrm{M}_s}, \tag{5}$$

$$R[k] = (1-\beta)R[k-1] + \beta R_s, \tag{6}$$

where P_{M_s} and R_s are the latest sample of ECN-marked packet probability and round-trip time which were obtained by the feedback information contained in the latest RTCP receiver report whereas $P_{\mathrm{M}}[k]$ and $R[k]$ are the exponentially weighted moving average (EWMA) of P_{M_s} and R_s with $0<\alpha<1$ and $0<\beta<1$. By selecting appropriate values for α and β, one can sufficiently smooth out the instantaneous fluctuation of P_{M_s} and R_s, thereby achieving slowly varying $Y[k]$ which may significantly improve the quality of delivered multimedia. In our simulation study in Section 3, we found that a good choice for these parameters is $\alpha = 0.01$, $\beta = 0.05$ and $\Delta = 0.1$ s.

On the other hand, upon every arrival of RTCP receiver report, the sender computes P_{M_s} and R_s as follows. Suppose that the sender receives ith receiver report at time t_i and $R[k]$ is the latest value of the EWMA of round-trip time at the given time t_i.

$$P_{\mathrm{M}_s} = \frac{\min[n_{\mathrm{M}}, (t_i - t_{i-1})R[k]]}{n_{\mathrm{pkt}}}, \tag{7}$$

$$R_s = (t_i - t_s) - t_d, \tag{8}$$

where n_{pkt} and n_{M} are the number of packets and the number of ECN-marked packets, which arrived at the receiver between the generation of $(i-1)$th and ith receiver reports. Recall that these values are fed back to the sender via the ith receiver report. The ith receiver report also contains t_s and t_d. t_s is the latest timestamp which was stamped by the sender on an RTCP sender report and echoed back to the sender by the receiver via the ith receiver report, and t_d is the time spent by the receiver between the receipt of the sender report and the generation of the ith receiver report. Thus, the round-trip time sample R_s can be calculated by subtracting $t_i - t_s$ by t_d as in (8).

If we follow the definition of P_{M}, the sample P_{M_s} must be calculated by $P_{\mathrm{M}_s} = n_{\mathrm{M}}/n_{\mathrm{pkt}}$. However, in our calculation of P_{M_s} in (7), we upper-bound n_{M} by the quantity $(t_i - t_{i-1})/R[k]$ which is interpreted as the number of round-trip times that can be contained in the interval between the arrival of $(i-1)$th receiver report and the arrival of ith receiver report. This is because in accordance with the ECN–TCP protocol [1] an ECN–TCP sender reacts to at most one ECN-marked packet in each round-trip time, halving its congestion window, although the sender may receive multiple ECN-echoed ACKs in a single round-trip time. Therefore, if the sender were the ECN–TCP sender, the number of congestion window reduction in an interval $t_i - t_{i-1}$ would be less than or equal to

[2] In this paper, we use the term wireless loss as any packet losses that have failed to be recovered by any involved link-level error recovery schemes.

$(t_i - t_{i-1})/R[k]$. In order to see the same statistics as ECN–TCP senders, ECN-based TF-WMFC senders should take into account this behavior of ECN–TCP senders. For this reason, we upper-bound n_{M} by $(t_i - t_{i-1})/R[k]$ in calculating P_{M_s}.

The sender sends the receiver an RTCP sender report periodically with an interval T. As discussed, the sender report contains the timestamp indicating its generation time. In our simulation study in Section 3 we let $T = 1.0$ s to minimize the bandwidth consumed by the sender reports.

2.3.2. Receiver behavior

The receiver generates an RTCP receiver report periodically with an interval T and sends it to the sender. The receiver report generation interval can be different from the sender report generation interval but in our study we make them equal.

Upon receipt of a data packet, the receiver increases n_{pkt} by 1 and if the packet is ECN-marked, increases n_{M} by 1 as well.

Upon receipt of a sender report, the receiver reads the timestamp t_s contained in the sender report and stores it until a new RTCP receiver report is generated to carry this timestamp to the sender.

Upon every epoch of T, the receiver sends a receiver report to the sender. Each receiver report contains information including the current n_{pkt}, n_{M}, t_s and the time during which the receiver stores the timestamp t_s, that is, t_d.

2.3.3. Initial sender behavior

Initially, the sender is not aware of ECN-marked packet probability $P_{\mathrm{M}}[k]$. This quantity can be known only after transmitting some portion of data to collect statistics. In our scheme we introduce a TCP-like rate adjustment scheme for the initial ramp-up of transmission rate.

Suppose that the sender updates $R[k]$ based on (6) and (8) upon every epoch of Δ. Until an RTCP receiver report containing nonzero n_{M} arrives, the sender keeps updating its transmission rate upon every epoch of round-trip time $R[k]$ as follows:

$$Y = \begin{cases} 2Y & \text{if } Y < \frac{W_{\mathrm{th}}}{R[k]}, \\ Y + \frac{s}{R[k]} & \text{if } Y \geqslant \frac{W_{\mathrm{th}}}{R[k]}, \end{cases} \tag{9}$$

where $R[k]$ is the latest round-trip time and $W_{\mathrm{th}}/R[k]$ is the threshold between the multiplicative increase and the additive increase in adjusting the transmission rate Y.

Upon arrival of a receiver report containing nonzero n_{M}, the initial sender behavior terminates.

3. Simulation results

In order to evaluate the performance of the proposed TF-WMFC scheme, we implement the scheme onto the *ns*-2 simulator [9] and adopt both real-time MPEG-4 codec and MPEG-4 FGS (fine-grained scalable) codec [12] as adaptive video applications. The real-time MPEG-4 codec is for live video applications where adaptive encoding is carried out via real-time encoder rate control so as to match the output rate of the encoder to the TCP-friendly rate $Y[k]$. In contrast, the MPEG-4 FGS codec is for stored video applications where a video is pre-encoded into two layers (base layer and enhancement layer) such that the output rate can be controlled by sending selectively a portion of enhancement-layer data.

3.1. Accuracy of ECN–TCP throughput model

First, we examine the accuracy of refined ECN–TCP throughput model (3) as compared to that of Floyd's model (2) when ECN-marked packet probability P_{M} is used as the congestion measure. Consider a wired network scenario with a single bottleneck link in Fig. 3 where all ECN–TCP flows share the 20 m s-long link and routers are

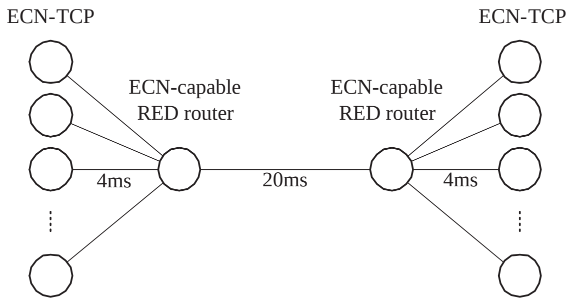

Fig. 3. A wired network scenario for the study of refined ECN–TCP model.

equipped with ECN-capable RED queue management scheme. The number of ECN–TCP flows and the bottleneck bandwidth are varied respectively from 8 to 128 and from 32 Mbps to 128 Mbps. We set packet size to be 1000 bytes. We change the RED parameters according to the bottleneck bandwidth, say C (Mbps), as similarly in [1] (see Table 1).

Fig. 4 shows the ECN–TCP throughput predicted by both refined model and Floyd's model as normalized by the average of ECN–TCP flows' actual throughput. For the throughput prediction, we measure P_M as seen by the ECN–TCP source and apply this probability to Eqs. (3) and (2). The refined model significantly outperforms Floyd's model in predicting actual throughput of ECN–TCP flows particularly as the number of ECN–TCP flows multiplexed increases and/or the bottleneck bandwidth decreases. This is because the congestion-window dormant period following each congestion window reduction plays a significant role in determining the throughput of ECN–TCP and our model takes this effect into account.

Table 1
RED parameters used in simulations

Parameters	Value (pkts)	Parameters	Value
min_{th}	$5C/16$	max_p	1.0
max_{th}	$50C/16$	w_q	0.002
Queuesize	$400C/16$		

The accuracy of the ECN–TCP throughput model (3) depends on the values of RED parameters, in particular, that of max_p. In the original work on RED scheme [1] $max_p = 0.1$ is recommended but in our scheme $max_p = 1.0$ is found to be a good choice as used in the study of Fig. 4. For the same network loading, $max_p = 0.1$ would yield less ECN marks, thereby resulting in more packet drops than $max_p = 1.0$. ECN–TCP sources react to ECN marks as well as dropped packets, halving congestion window, whereas the ECN–TCP throughput model (3) takes only ECN marks into account. Therefore, the model would overestimate the actual throughput of ECN–TCP flows as packet drop rate at the RED routers increases. Fig. 5 shows the ECN–TCP throughput predicted by both the refined model and the Floyd's model as normalized by the average of ECN–TCP flows' actual throughput for the same network scenario used in Fig. 4 with $max_p = 0.1$. As compared to the case with $max_p = 1.0$ in Fig. 4,

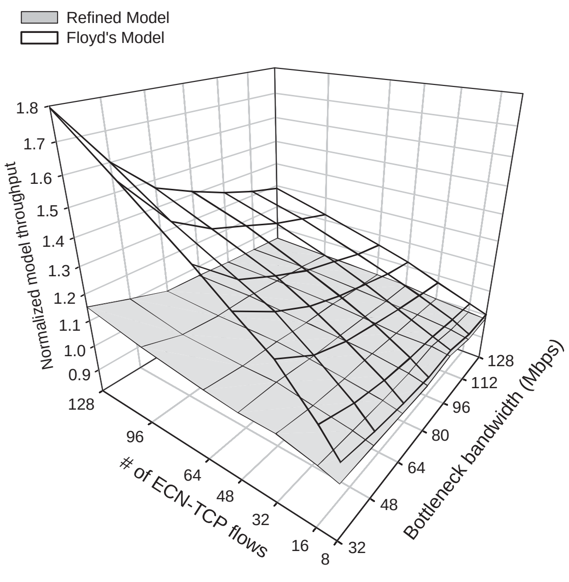

Fig. 4. Accuracy of ECN–TCP throughput models.

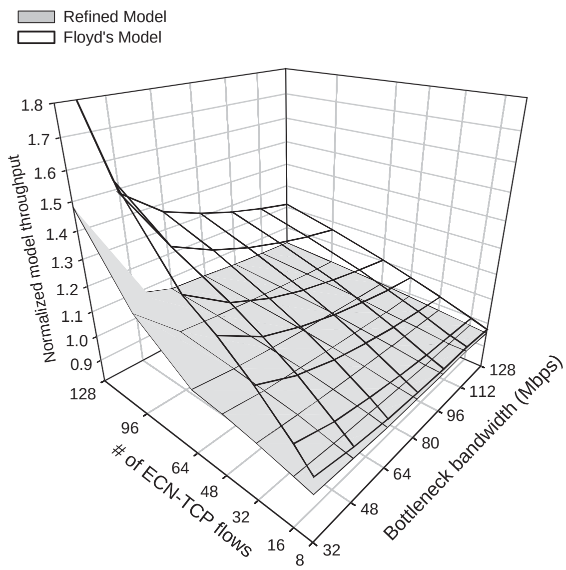

Fig. 5. Accuracy of ECN–TCP throughput models ($max_p = 0.1$ case).

in this case both models overestimate the actual ECN–TCP throughput particularly as the number of ECN flows increases and the bottleneck bandwidth decreases. This is because the actual throughput of ECN–TCP flows is reduced due to the increased packet drop rate at the RED router but the models do not react to the packet drops. In conclusion, it is necessary to minimize packet drops at the RED routers by marking packets earlier and rather aggressively in order to avoid such an overestimation. In this end we use $\max_p = 1.0$ throughout the paper.

The other factor that can degrade the accuracy of the proposed throughput model is the so-called ECN-backoff behavior of the ECN–TCP sources, which was not taken into account in the model. The ECN-backoff behavior occurs as the number of ECN–TCP flows sharing a link increases so that the congestion window size of each flow is forced to be less than 1. In Fig. 4 the normalized throughput predicted by the proposed model increases up to 1.15 as the number of ECN–TCP flows sharing 32 Mbps link increases up to 128, which is in fact the effect of this ECN-backoff behavior.

3.2. Performance of TF-WMFC over wired network

In this subsection we study the same wired network scenario as in Fig. 3 but we add ECN-based TF-WMFC flows to ECN–TCP flows with the ratio of the number of ECN-based TF-WMFC flows to that of ECN–TCP flows being 1 (see Fig. 6). Then, we control the TF-WMFC flows as proposed in previous section and see how they share the bottleneck bandwidth with concurrent ECN–TCP flows. For the simulation of TF-WMFC flows, we set the parameters as in Table 2.

Fig. 7 shows the average throughput of ECN-based TF-WMFC flows as normalized by the average throughput of ECN–TCP flows. When the proposed TF-WMFC scheme uses the refined throughput model, the normalized average throughput of TF-WMFC flows remains between 0.9 and 1.15, which implies that TF-WMFC flows equally share the bottleneck bandwidth with ECN–TCP flows, i.e., TF-WMFC flows are ECN–TCP friendly. In contrast, if the TF-WMFC scheme uses the Floyd's model, the performance degrades particularly as the number of ECN–TCP flows multiplexed increases and/or the bottleneck bandwidth decreases.

Table 2
TF-WMFC parameters used in simulations

Parameter	Value	Parameter	Value
α	0.01	T	1.0 sec
β	0.05	s	1000 bytes
Δ	0.1 s	W_{th}	64 k bytes

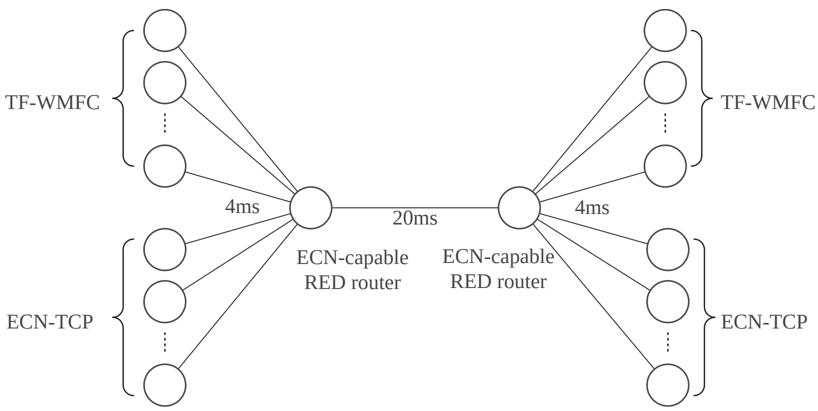

Fig. 6. A network scenario for the study of TF-WMFC over wired network.

Next we study the transient behavior of TF-WMFC. Consider the network configuration in Fig. 6 with the bottleneck bandwidth being 128 Mbps. Initially there are 32 ECN–TCP flows in the network, 64 TF-WMFC flows join simultaneously at 10 s, and 96 ECN–TCP flows are added at 100 s. Fig. 8(a) shows the traces of average throughput of TF-WMFC flows and ECN–TCP flows, respectively. We see that TF-WMFC flows joining at 10 s immediately find the TCP-friendly rate (128 Mbps/96 flows = 1.33 Mbps) with bounded overshooting, and at 100 s when 96 ECN–TCP flows are added, rapidly reduce their rates to the new TCP-friendly rate (128 Mbps/192 flows = 0.66 Mbps). Fig. 8(b) shows the individual trace of three sample TF-WMFC flows out of 64 flows.

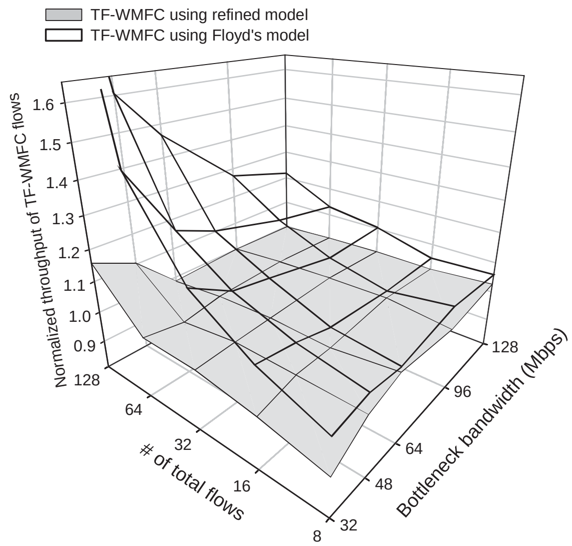

Fig. 7. Average throughput of TF-WMFC flows (normalized by the average throughput of ECN–TCP flows).

3.3. Performance of TF-WMFC over wireless hop

In the previous subsection we showed that the throughput of wired TF-WMFC flows is almost equal to that of wired ECN–TCP flows in a long-term average sense. In this subsection we show that the throughput of TF-WMFC flows does not depend on whether or not they travel through a

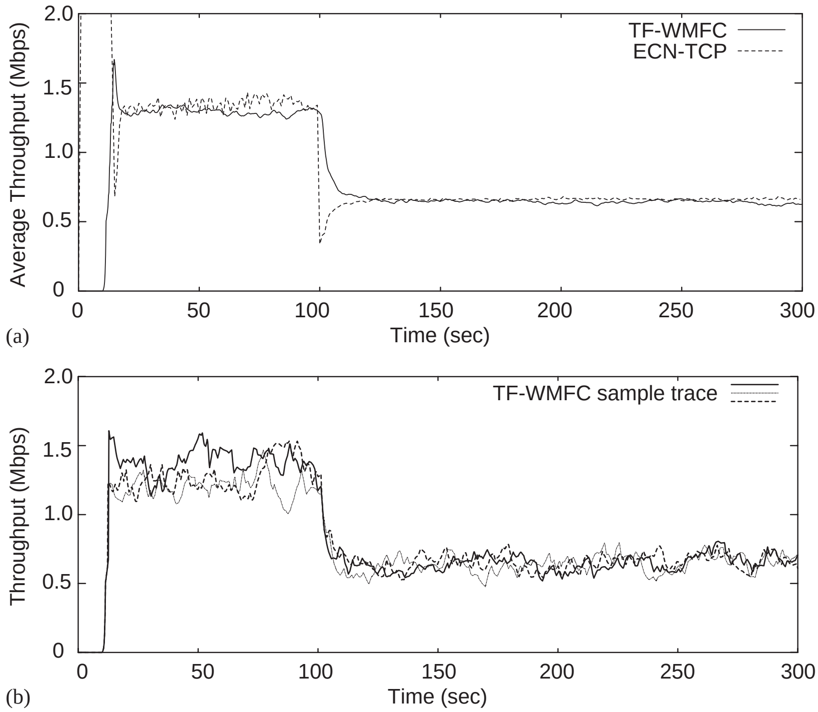

Fig. 8. Transient performance of TF-WMFC: (a) average throughput trace of TF-WMFC and ECN–TCP flows, (b) throughput trace of sample TF-WMFC flows.

wireless hop suffering wireless loss. By combining these two results, we will then conclude that the throughput of a TF-WMFC flow would be as much as that of a wired ECN–TCP flow, no matter it experiences wireless loss or not.

In contrast, if a multimedia flow is controlled by the so-called *loss-based* TFRC (TCP friendly rate control) scheme in [4] or its variants, it is quite obvious that its throughput would significantly degrade when it travels through a wireless hop because the TFRC scheme reacts to wireless loss, reducing its sending rate.

In order to quantify this throughput difference between ECN-based TF-WMFC flows and loss-based TFRC flows, we also implement a variant of TFRC scheme where $P_L[k]$ and $R[k]$ are known to the sender using the RTP/RTCP protocol and the sender calculates the TCP-friendly rate by substituting $P_L[k]$ for $P_M[k]$ in Eq. (4). Throughout the paper we call this scheme as the loss-based TFRC scheme for the sake of convenience, although several variants can exist depending on detailed implementation.

Consider the simulation scenario in Fig. 9 where four wired ECN-based TF-WMFC flows, four wireless ECN-based TF-WMFC flows and eight ECN–TCP flows share the 20 m s-long, 16 Mbps bottleneck link. We assume that packets are lost in the wireless hop according to the Bernoulli trials with a certain loss probability.

In Fig. 10 we plot the individual and the average throughput of wireless TF-WMFC flows with respect to the increasing wireless loss probability, as normalized by the average throughput of wired TF-WMFC flows. The normalized throughput of wireless TF-WMFC flows stays in the vicinity of 1 irrespective of wireless loss probability, which implies that the throughput of wireless TF-WMFC flows is not only independent of wireless loss probability but also almost equal to that of wired TF-WMFC flows and consequently that of wired ECN–TCP flows.

For comparison, we replace eight ECN-based TF-WMFC flows by eight loss-based TFRC flows correspondingly and simulate the same network scenario. In Fig. 10 we also plot the individual and the average throughput of wireless TFRC flows with respect to increasing wireless loss probability, as normalized by the average throughput of wired TFRC flows. In contrast to the TF-WMFC case, the normalized throughput of wireless TFRC flows drastically falls down as the wireless loss probability increases, which clearly shows that the ECN-based TF-WMFC scheme is superior to the loss-based TFRC scheme in the wireless environment.

In Fig. 11 we also compare the throughput traces of wireless and wired multimedia flows. Under the ECN-based TF-WMFC scheme the throughput of a wireless flow agrees with that of a

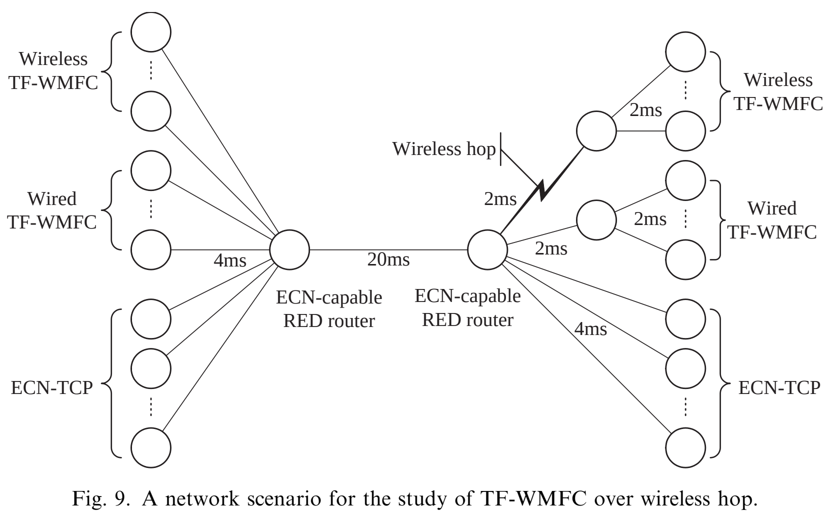

Fig. 9. A network scenario for the study of TF-WMFC over wireless hop.

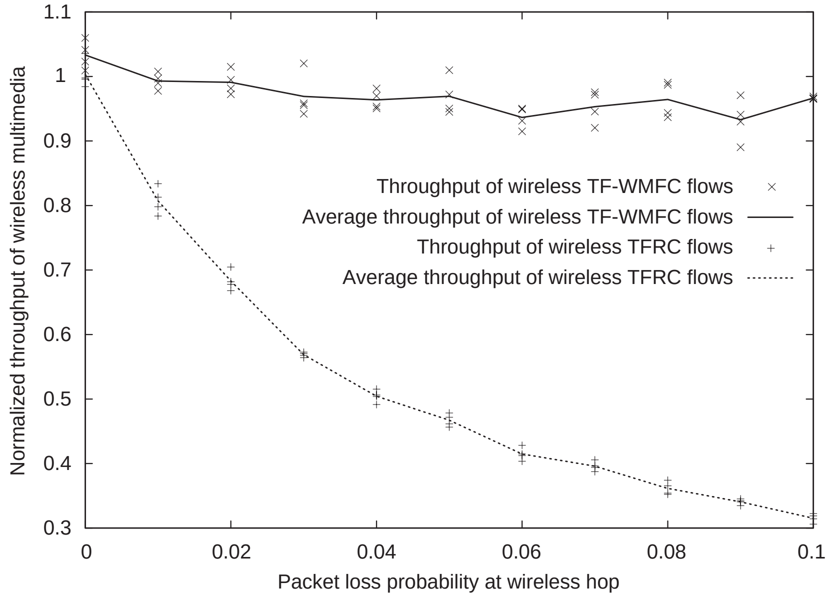

Fig. 10. Throughput of wireless multimedia flows under ECN-based TF-WMFC scheme and loss-based TFRC scheme.

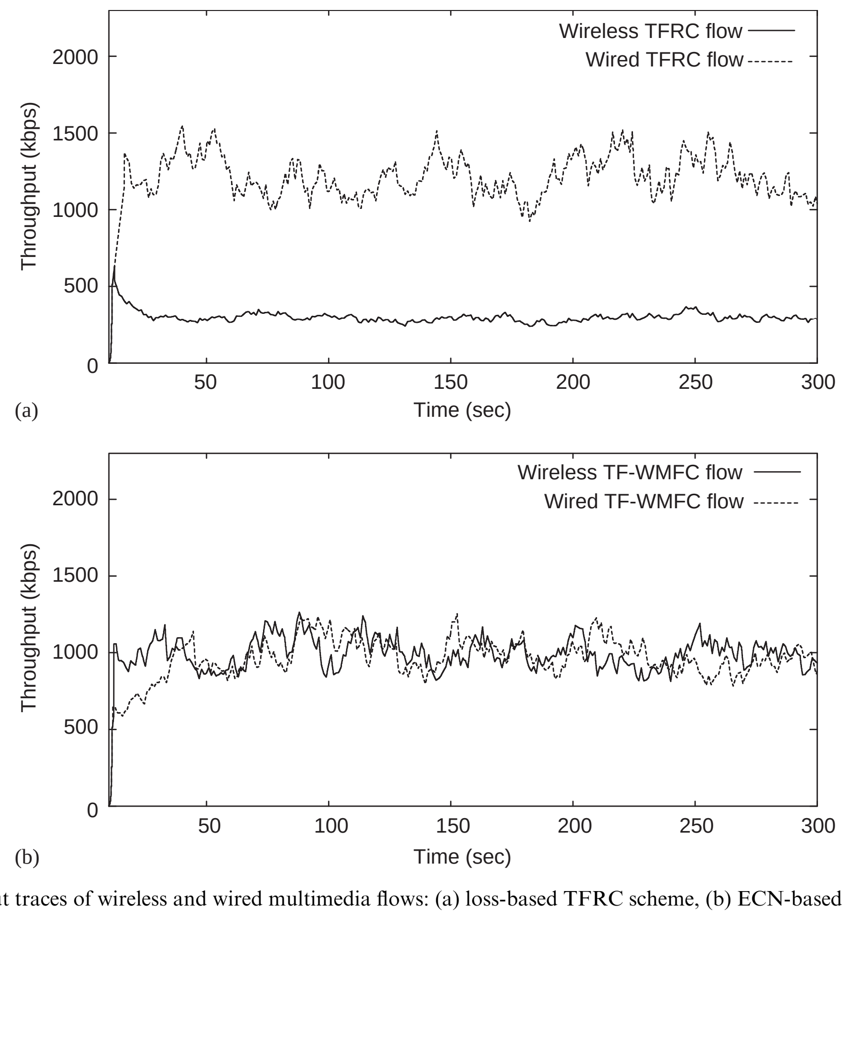

Fig. 11. Throughput traces of wireless and wired multimedia flows: (a) loss-based TFRC scheme, (b) ECN-based TF-WMFC scheme.

wired flow (see Fig. 11(b)) whereas under the loss-based TFRC scheme the throughput of a wireless flow is much lower than that of a wired flow (see Fig. 11(a)).

3.4. Wireless MPEG-4 video over TF-WMFC

In this subsection we quantify the improvement on video quality that can be achieved by the proposed ECN-based TF-WMFC scheme when the video is delivered over a wireless link. The comparison is made with the loss-based TFRC scheme. We consider both real-time MPEG-4 codec and MPEG-4 FGS codec which are the most popular video compression techniques today particularly for wireless adaptive video applications. The real-time MPEG-4 codec is for live video applications whereas the MPEG-4 FGS codec is for stored video applications.

Consider the same simulation scenario shown in Fig. 9 with the packet loss probability at the wireless hop being 10%. First, we experiment with live video transmission under both ECN-based TF-WMFC and loss-based TFRC schemes respectively. The *Terminator-2* QCIF video sequence is used as the test sequence and encoded on-line using the real-time MPEG-4 encoder. Fig. 12 shows the PSNR trace and a sample frame of the test video decoded at the receiver. We found that the ECN-based TF-WMFC scheme significantly outperforms the loss-based TFRC scheme, improving the PSNR by 5.0 dB on average. As compared to Fig. 12(c), Fig. 12(b) shows that the loss-based scheme causes more available bandwidth degradation than the ECN-based scheme, thereby producing more blocking artifacts and hence degrading visual quality more significantly. Next, we experiment with stored video transmission under both schemes, respectively. The *Stefan*

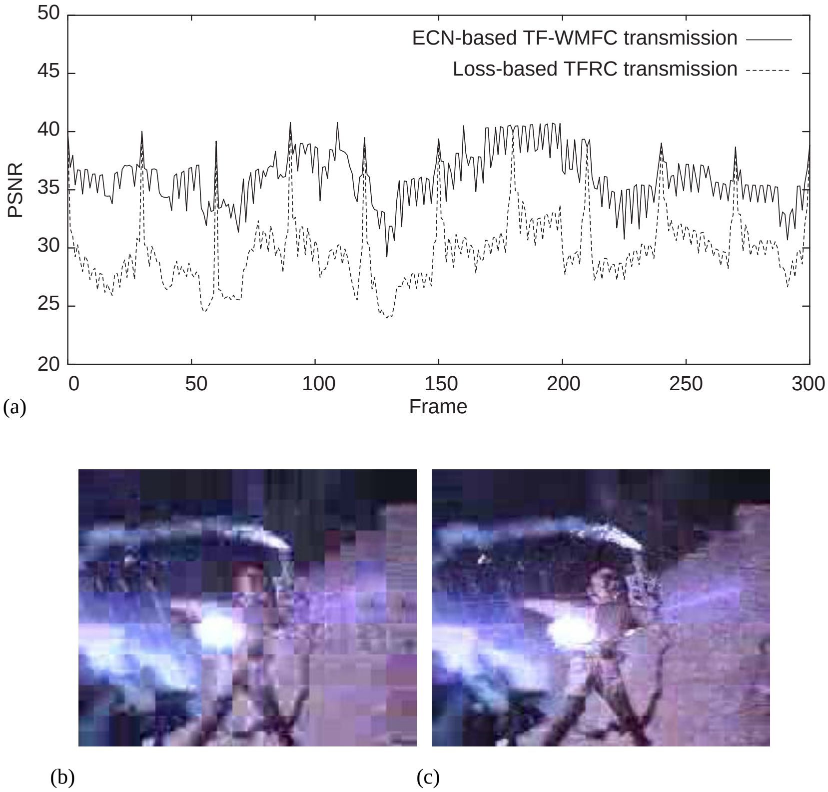

Fig. 12. Quality of real-time MPEG-4 video over wireless: (a) PSNR trace, (b) Loss-based TFRC, (c) ECN-based TF-WMFC.

QCIF video sequence is pre-encoded off-line by the MPEG-4 FGS encoder, stored at the stream DB and retrieved on-line by the FGS server at the given TCP-friendly rate for transmission. Fig. 13 shows the PSNR trace and a sample frame of the test video decoded at the receiver. Again, the ECN-based TF-WMFC scheme significantly outperforms the loss-based TFRC scheme, improving the PSNR by 8.7 dB on average.

4. Conclusion

The contribution of this paper is two folds. First, we refine the well-known Floyd's TCP throughput model by taking into account the congestion-window dormant period (where congestion window is kept constant) following each congestion window reduction in the congestion avoidance phase. We show that this refined model predicts the long-term average throughput of a TCP flow more precisely than Floyd's model, thereby serving as a good basis for TCP-friendly multimedia flow control. Second, in order to avoid the throughput degradation of a multimedia flow due to wireless loss, we propose an ECN-based TCP-friendly flow control scheme (called ECN-based TF-WMFC) where instead of packet loss, ECN-marked packet is used as a congestion indicator and the TCP-friendly rate is computed using the refined TCP throughput model. We show that the long-term average throughput of a TF-WMFC flow is always equal to that of concurrent wired ECN–TCP flows no matter it experiences wireless loss or not. Through simulation experiments with MPEG-4 video, we demonstrate that the ECN-based TF-WMFC scheme significantly improves the quality of delivered video in a wireless environment, compared with the conven-

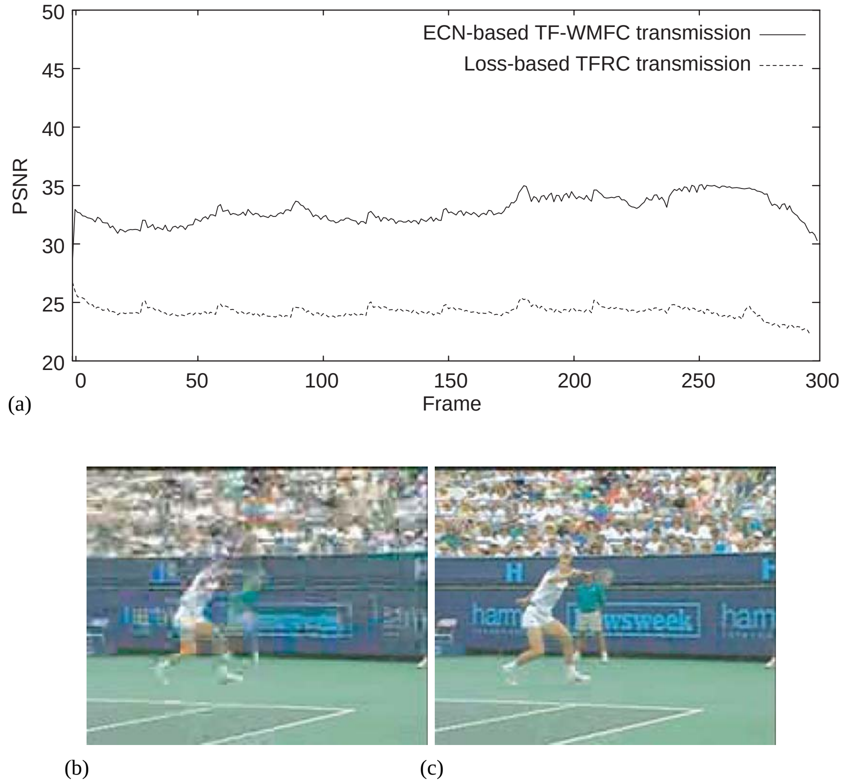

Fig. 13. Quality of MPEG-4 FGS video over wireless: (a) PSNR trace, (b) loss-based TFRC, (c) ECN-based TF-WMFC.

tional loss-based TCP-friendly flow control scheme.

Appendix A. Derivation of ECN–TCP throughput model in (3)

In the congestion avoidance phase of ECN–TCP, each congestion window reduction is followed by a congestion-window dormant period where congestion window size is kept constant. More specifically, when an ECN–TCP receiver receives an ECN-marked packet, it starts and continues to send *requests* for congestion window reduction to the corresponding ECN–TCP sender by continuing to mark ECE-bit in the header of ACK packets, until it receives *confirm* from the sender by receiving a data packet with CWR-bit in the header being marked, which implies that the congestion window has been halved. On the other hand, upon receipt of the first ECN-echoed ACK packet with its ECE-bit being marked (i.e., the first *request*), the sender halves the congestion window, sends *confirm* to the receiver by marking CWR-bit in the header of first data packet, and enters a congestion-window dormant period by ceasing to respond to following ACKs until the last data packet it sent to the receiver before the receipt of the first *request*, is acknowledged. After it is acknowledged, the sender exits from the dormant period by resuming its additive increase of congestion window. From this behavior of an ECN–TCP sender and receiver pair, we immediately know that upon occurrence of an ECN-marked packet, the corresponding ECN–TCP source halves its congestion window, keeps this congestion window unchanged approximately for one round trip time, and following this congestion-window dormant period, increases its congestion window at the rate of 1 packet per one round trip time until the next ECN-marked packet occurs.

For ease of derivation, we assume that the round trip time R is constant and the occurrence of ECN-marked packet is periodic. We also assume that it takes one round trip time that an ECN–TCP source is informed of the occurrence of an ECN-marked packet since its window size reaches its maximum W. Fig. 14 shows the congestion window evolution of ECN–TCP in the congestion avoidance phase under these assumptions. Let the maximum value of the congestion window be W packets. Then, the total number of packets delivered per cycle, denoted by X, is

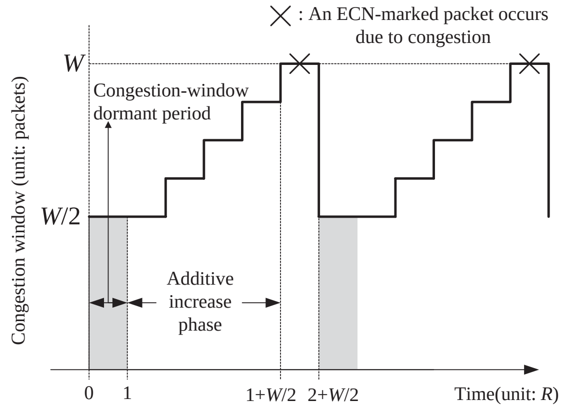

Fig. 14. Congestion window evolution of ECN–TCP under periodic occurrence of ECN-marked packet.

$$X = \frac{W}{2} + \sum_{n=0}^{W/2}\left(\frac{W}{2}+n\right) = \frac{3W^2}{8} + \frac{5W}{4} \tag{A.1}$$

and each cycle takes $2 + W/2$ round trip times. Since one packet is ECN-marked for every X packets, the ECN-marked packet probability is given by $P_M = 1/X$. Solving for W with this equation and (A.1), we get

$$W = \sqrt{\frac{8}{3P_M} + \frac{25}{9}} - \frac{5}{3}. \tag{A.2}$$

By definition, the throughput of ECN–TCP, denoted by Y, is given by

$$Y = \frac{(3W^2/8 + 5W/4)s}{(2 + W/2)R}. \tag{A.3}$$

Substituting (A.2) into (A.3), we get

$$Y = \frac{s}{P_M\left(\sqrt{\frac{2}{3P_M}} + 25/36 + 7/6\right)R}. \tag{A.4}$$

References

[1] S. Floyd, V. Jacobson, Random Early Detection Gateways for Congestion Avoidance, IEEE/ACM Trans. Networking 1(4) (August 1997) 397–413.

[2] S. Floyd, M. Handley, J. Padhye, J. Widmer, Equation-based congestion control for unicast applications, SIGCOMM 2000, August 2000, pp. 43–56.
[3] 3rd Generation Partnership Project, Radio link control (RLC) protocol specification (3GPP TS 25.322), 2002.
[4] M. Handley, S. Floyd, J. Padhye, J. Widmer, TCP friendly rate control (TFRC): protocol specification, RFC3448, January 2003.
[5] H. Inamura, G. Montenegro, R. Ludwig, A. Gurtov, F. Khafizov, TCP over second (2.5G) and third (3G) generation wireless networks, RFC3481, February 2003.
[6] E. Kohler, M. Handley, S. Floyd, J. Padhye, Datagram congestion control protocol (DCCP), IETF Internet Draft (work in progress) draft-ietf-dccp-spec-05.txt, October 2003.
[7] J. Mahdavi, S. Floyd, TCP-friendly unicast rate-based flow control, Note sent to end2end-interest mailing list, January 1997.
[8] S. McCreary, K. Claffy, Trends in wide area IP traffic patterns, http://www.caida.org/outreach/papers/AIX0005/.
[9] The network simulator—ns-2, http://www.isi.edu/nsnam/ns/.
[10] T. Ott, J. Kemperman, M. Mathis, The stationary behavior of ideal TCP congestion avoidance algorithm, ftp://ftp.bellcore.com/pub/tjo/TCPwindow.ps.
[11] J. Padhye, V. Firoiu, D. Towsley, J. Kurose, Modeling TCP throughput: a simple model and its empirical validation, SIGCOMM 98, 1998, pp. 303–314.
[12] H.M. Radha, M.V.D. Schaar, Y. Chen, MPEG-4 fine-grained scalable video coding method for multimedia streaming over IP, IEEE Trans. Multimedia 3(1) (March 2001) 53–68.
[13] K.K. Ramakrishnan, S. Floyd, D. Black, The addition of explicit congestion notification (ECN) to IP, RFC 2481, January 1999.
[14] R. Rejaie, M. Handley, D. Estrin, An end-to-end rate-based congestion control mechanism for realtime streams in the Internet, INFOCOM 99, 1999, pp. 1337–1345.
[15] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, RTP: a transport protocol for real-time applications, RFC1889, January 1996.
[16] D. Sisalem, H. Schulzrinne, The loss-delay based adjustment algorithm: a TCP-friendly adaptation scheme, NOSSDAV98, 1998.

Seong-jun Bae received the B.S. degree in Electronics Engineering from Korea University, Seoul, Korea, in 1997, and the M.S. degree in Electrical Engineering from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 1999. He is currently a Ph.D. candidate in Electrical Engineering and Computer Science at KAIST. His current research interests include video streaming over heterogeneous networks, multimedia flow control, peer-to-peer networks and distributed systems.

Prof. Song Chong received his B.S. and M.S. degrees in Control and Instrumentation Engineering from Seoul National University, Seoul, Korea, in 1988 and 1990, respectively, and his Ph.D. degree in Electrical and Computer Engineering from the University of Texas at Austin in 1995. From 1994 to 1996, he was a Member of Technical Staff in the Performance Analysis Department at AT&T Bell Laboratories, Holmdel, New Jersey, USA. He is currently an Associate Professor with the Department of Electrical Engineering and Computer Science, Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea. His research interests are in high-speed communication networks, high-performance switching/routing systems, multimedia networking and performance evaluation. He has published more than 25 papers in international journals and conferences and holds three U.S. patents with several others pending. He is an Editor and an Associate Publication Editor for the Journal of Communications and Networks. He has served as a Technical Program Committee member of IEEE INFOCOM ('97, '99, '03), an Organizing/Program Committee member of PV ('01–'04) and Technical Program Co-chair of ICBN '04.