Hindawi Publishing Corporation
Advances in Multimedia
Volume 2007, Article ID 68502, 13 pages
doi:10.1155/2007/68502

Research Article

Cross-Layer Design of Source Rate Control and Congestion Control for Wireless Video Streaming

Peng Zhu,[1] Wenjun Zeng,[2] and Chunwen Li[3]

[1] *Hitachi (China) Research and Development Corporation, Beijing 100004, China*
[2] *Department of Computer Science, University of Missouri-Columbia, MO 65211, USA*
[3] *Department of Automation, Tsinghua University, Beijing 100084, China*

Received 30 December 2006; Revised 22 May 2007; Accepted 11 July 2007

Recommended by Zhu Han

Cross-layer design has been used in streaming video over the wireless channels to optimize the overall system performance. In this paper, we extend our previous work on joint design of source rate control and congestion control for video streaming over the wired channel, and propose a cross-layer design approach for wireless video streaming. First, we extend the QoS-aware congestion control mechanism (TFRCC) proposed in our previous work to the wireless scenario, and provide a detailed discussion about how to enhance the overall performance in terms of rate smoothness and responsiveness of the transport protocol. Then, we extend our previous joint design work to the wireless scenario, and a thorough performance evaluation is conducted to investigate its performance. Simulation results show that by cross-layer design of source rate control at application layer and congestion control at transport layer, and by taking advantage of the MAC layer information, our approach can avoid the throughput degradation caused by wireless link error, and better support the QoS requirements of the application. Thus, the playback quality is significantly improved, while good performance of the transport protocol is still preserved.

1. INTRODUCTION

Multimedia streaming over the wireless channels has been a very challenging issue due to the dynamic uncertain nature (e.g., variable available bandwidths and random packet losses) of the wireless channels [1]. To address this problem, many solutions have been proposed, of which congestion control for streaming media at the transport layer and source rate control at the application layer are two basic components.

At the transport layer, congestion control for streaming multimedia is adopted to make sure the users have a fair share of the network resources. Congestion control for the wireless scenario has to differentiate packet loss due to wireless link error from packet loss due to congestion, as the transport protocol needs to decrease the sending rate only when there is congestion in the network [2]. Source rate control is typically adopted at the application layer to optimize the playback quality, subject to the bandwidth constraint provided by the congestion control mechanism and the QoS requirements of the multimedia application (e.g., the end-to-end delay constraint).

However with the traditional layered design principle, source rate control and congestion control are usually designed separately without sufficient communication with each other, which imposes a limitation on the overall system performance. For example, traditional congestion control for streaming multimedia usually needs to smooth their sending rate to help the application achieve smooth playback quality. But this does not work all the time, because the coding complexity of the video frames may change abruptly. Moreover, source rate control alone cannot guarantee the end-to-end delay constraint at any time due to the minimum bandwidth requirement and the quality smoothness constraint of the video source. Actually, the end-to-end delay constraint also imposes constraints on the sending rate, which cannot be well supported in the layered design mode.

The cross-layer design approach, on the other hand, can achieve the better overall system performance, and there have been many cross-layer design solutions proposed for wireless video streaming [3]. However, most of them mainly concern about how to utilize the information from the MAC/physical layer. In this paper, we extend our work in [4] to the wireless scenario, and propose cross-layer design of source rate

control and congestion control for wireless video streaming. Our main contributions are as follows.

(1) We first extend the *QoS-aware* congestion control mechanism-TFRCC (TCP friendly rate control with compensation) that we proposed in [4] to better support the QoS requirements of multimedia applications, to the wireless scenario. We provide a detailed discussion about how to obtain a smooth measurement of the parameters, and how to enhance the overall performance of the transport protocol in terms of rate smoothness and responsiveness.

(2) Based on the above work, we extend our previous work, joint design of source rate control and QoS-aware congestion control, to wireless video streaming. How to enhance the cross-layer design mechanism is discussed and a thorough performance evaluation is conducted under different wireless scenarios. Simulation results show that our approach can significantly improve the playback quality of the application and maintain good long-term TCP-friendliness of the transport protocol, thus optimizing the overall performance.

The remainder of this paper is organized as follows. Section 2 discusses the related work. In Section 3, we briefly introduce our joint design work in [4]. Section 4 describes how to extend the QoS-aware congestion control algorithm to the wireless scenario. Simulation results are presented in Section 5. Section 6 gives the concluding remarks.

2. RELATED WORK

2.1. Congestion control for streaming multimedia

Congestion control for streaming multimedia has to take into account not only the fairness and responsiveness of the transport protocol, but also the smoothness of the sending rate to help the multimedia application achieve better playback quality [5]. A number of TCP-friendly congestion control schemes for the wired channels have been proposed to provide smooth sending rates. These include the window-based schemes [6, 7] and the rate-based schemes which can be further classified into the probe-based [8, 9] and equation-based schemes [10, 11].

The above approaches all assume that every packet loss is an indication of congestion, which is not held for the wireless channels. As in the wireless scenario, packet losses can also be attributed to link error [12]. So these mechanisms cannot be directly applied to the wireless scenario. To overcome this problem, several mechanisms have been proposed to distinguish packet losses due to link errors from those due to congestion [13]. For example, an agent is proposed to be installed at the edge of wired and wireless networks to measure the conditions of these two types of networks separately, thus the wireless loss can be differentiated from the packet loss due to congestion [14, 15]. In [12, 16], the proposed methods focus on differentiating the congestion loss from the erroneous packet loss by adopting some heuristic methods such as interarrival time or packet pair. Such solutions expect a packet to exhibit a certain behavior under network congestion or wireless errors. However, a specific behavior of a packet in the Internet reflects the joint effect of several factors, and it is hard to predict the behaviors of the packets by using a simple pattern. Akan and Akyildiz proposed an equation-based approach—the analytical rate control scheme (ARC) for multimedia traffic in wireless networks [2], which only requires the statistical information of the wireless losses, thus avoiding precise differentiation between the congestion loss and the erroneous packet loss. Chen and Zakhor have proposed a pure end-to-end approach in [17, 18], which creates multiple simultaneous TFRC connections on the same wireless path instead of distinguishing packet losses due to link errors from those due to congestion to fully utilize the bandwidth.

2.2. Source rate control for streaming multimedia

Source rate control at the application layer is to make the source rate match the channel condition to achieve better video quality. At the receiver side, adaptive media playout mechanism is proposed in [19, 20] to make sure the end-to-end delay constraint is met by adaptively varying the playout speed at the receiver. At the sender side, adaptive adjustment of the source rate is proposed based on the channel condition and the QoS requirements of the application. For example, the proportional plus derivative (PD) controller is used in [1] to determine the source rate according to the encoder buffer state. In [21], both the encoder buffer state and the end-to-end delay constraint are considered using a virtual network buffer management algorithm for bitstream switching applications.[1] In [22], a global rate control model is adopted to take into account the encoder buffer state as well as the end-to-end delay constraint of the application. A new transport protocol building upon the TFRC protocol is also proposed to take into account the characteristics (e.g., variable packet size) of the multimedia flows and achieve better performance.

Compared to traditional congestion control and source rate control, our approach is unique in that it provides a more flexible framework to allow a joint decision of source rate and sending rate to optimize the overall system performance.

2.3. Cross-layer design for streaming multimedia

Cross-layer design has been proposed for wireless video streaming to improve the overall system performance [23, 24]. However most of these schemes mainly concern about how to utilize the information from the MAC/physical layer.

Recently, the cross-layer design principle is also introduced to make the congestion control take into account both TCP-friendliness and "multimedia-friendliness." For example, constrained TCP-friendly adaptation framework (CTFAF) is proposed in [9], where the design of the congestion

[1] Bitstream switching can be thought of as a kind of source rate control. In bitstream switching applications, there are several streams with different bit rate for the same content. The sender can intelligently switch to the stream with the appropriate bit rate according to the bandwidth constraint.

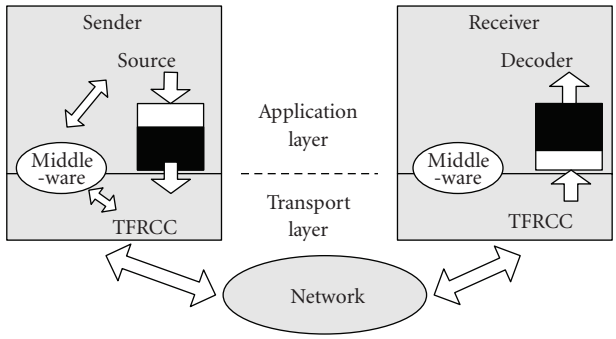


Figure 1: The system architecture.

control algorithm takes into account some QoS constraints (e.g., maximum and minimum bandwidth, rate adaptation granularity) of multimedia applications. In [25], a media-friendly congestion control mechanism is proposed to optimize the overall video quality by using a utility-based model. It adopts a two-timescale approach, which can optimize the video quality in short term (multimedia-friendliness) and meet the TCP-friendliness in long term. In [26], the bandwidth requirement of the multimedia application can be met by achieving proportional fairness with TCP, and multimedia-friendliness is regarded as TCP-friendliness with a weighting factor. The congestion control can adaptively adjust the weighting factor to get the necessary bandwidth and meet the QoS requirements of the multimedia application.

In [4], we propose a joint design algorithm of source rate control at application layer and congestion control at transport layer for streaming video over the Internet. By allowing the sending rate to temporarily violate TCP-friendliness to better support the QoS requirements of the application, the video quality is significantly improved. The long-term TCP-friendliness can be preserved by adopting the rate compensation algorithm. However, the proposed congestion control mechanism has the same assumption that packet loss is the sign of congestion, thus cannot be directly applied to the wireless scenario.

In this paper, we focus on cross-layer design of source rate control and congestion control for wireless video streaming, which can make transport protocol better support the QoS requirements (e.g., end-to-end delay constraint) of the application, and achieve the better overall performance.

3. JOINT DESIGN FOR THE WIRED SCENARIO

3.1. The system architecture

The system architecture is illustrated in Figure 1. At the transport layer, we proposed a QoS-aware congestion control mechanism, named TFRCC (TCP-friendly rate control with compensation), based on TFRC [10]. TFRCC can provide better support for the QoS requirements of the application by allowing temporal violation of TCP-friendliness, while the long-term TCP-friendliness can be preserved by introducing a rate compensation algorithm. At the application layer, the virtual network buffer management mechanism, denoted as VB (as described below), is used to translate the QoS requirements of the application to the constraint of source and sending rates. There is a middleware component located between the application layer and the transport layer. The joint decision of the source rate and the sending rate is done within the middleware at the sender.

3.2. The joint design algorithm

Next we will briefly introduce the joint design algorithm. One can refer to [4] for more details. From the application layer perspective, let us assume a virtual network buffer located between the sender and the receiver that abstracts the potentially complex network topology, and accounts for the delay and loss of packets introduced in the network. Denote $Be(k)$, $Bd(k)$, and $Bv(k)$,[2] respectively, as the encoder buffer, the decoder buffer, and the virtual network buffer occupancies at time k (when frame k is to be placed into the encoder buffer). Let $R(k)$, $Rs(k)$, and $C(k)$, respectively, be the kth video frame size, the amount of data sent by the sender, and the amount of data actually received by the receiver at time k. Denote BE and BD, respectively, as the encoder and decoder buffer sizes, and suppose that N is the end-to-end startup delay (in terms of frame number). Then it can be easily derived that if we can maintain the encoder buffer to meet (1) by selecting appropriate source and sending rates, the overflow and underflow of the encoder and decoder buffers can be avoided:

$$\max\left(0, \sum_{i=k+1}^{k+N} C(i) - Bv(k) - BD\right) \leq Be(k) \leq \min\left(BE, \sum_{i=k+1}^{k+N} C(i) - Bv(k)\right). \tag{1}$$

Let us count the feedback intervals of TFRCC as K. At time k, by using the nominal sending rate of current feedback interval $Ri(K)$ (bytes/frame) to estimate the receive rates of the future N frame periods in (1), we can derive the following two bounds for $Be(k)$ according to (1):

$$B_u = \min\left(N * Ri(K) - Bv(K), BE\right), \quad B_l = \max\left(0, N * Ri(K) - Bv(K) - BD\right). \tag{2}$$

Note that we will place extra safety margins for the bounds in the implementation to deal with the possible estimation error of $Bv(K)$ caused by the possible feedback loss or other factors. The above constraints are derived by VB at application layer.

[2] $Bv(k)$ can be estimated according to the difference between the amount of data sent at the sender and the amount of data received at the receiver.

At transport layer, TFRCC first uses the same algorithm as TFRC to calculate the TCP-friendly sending rate $B(K)$ (bytes/s). Note that the actual sending rate of TFRCC $Rs(k)$ is allowed to temporally violate TCP-friendliness, so TFRCC uses a rate compensation algorithm, based on the TCP-friendly sending rate $B(K)$ and the accumulated difference between the amount of data actually sent and the ideal TCP-friendly value, to determine the nominal sending rate $Ri(K)$ so as to preserve long-term TCP-friendliness.

Then with the encoder buffer constraint of (2) provided by VB and the long-term TCP-friendliness constraint of $Ri(K)$ provided by TFRCC, the source rate and sending rate are jointly determined in the middleware component of the sender.

3.2.1. Decision of the source rate and the sending rate

The actual sending rate $Rs(k)$ is usually set to $Ri(K)$ for good TCP-friendliness. Then the source rate is determined to maintain the encoder buffer within the bounds of (2), together with the consideration of the video quality smoothness constraint and the minimum acceptable/maximum necessary video quality of the video source.

3.2.2. Adaptation at the beginning of a new feedback interval

Suppose at time k_1, the sender receives a new feedback from the receiver, then the sending rate is updated as $Ri(K+1)$. Note that there exists the maximum admissible sending rate constraint, which is imposed by the encoder and decoder buffer sizes. If $Ri(K+1)$ is too large so that $N * Ri(K+1) > BE+BD+Bv(K+1)$, we can find that it will lead to $B_u < BE < B_l$ according to (2), which consequently causes the decoder buffer to overflow. In this case, we need to decrease the sending rate to make sure the maximum sending rate constraint is met (i.e., making $B_l < BE$). The corresponding change of the amount of sent data caused by the sending rate adjustment will be recorded and compensated later.

Moreover, at times $k_1-N,\ldots,k_1-1$, the estimation of the future receive rates using $Ri(K)$ might not have been accurate and the constraints of (1) might not actually be met, since the sending rate after time k_1 has been changed to $Ri(K+1)$. So if necessary, the readjustment of the size of the encoded frame $k_1-N,\ldots,k_1-1$ (if still available in the encoder buffer), subject to the quality smoothness constraint, is used to make sure that the decoder buffer at times $k_1,\ldots,k_1+N-1$ will not underflow and overflow. If this cannot prevent the decoder buffer from underflow or overflow, we will have to adjust the sending rate to pull back the decoder buffer fullness to within the safety region. For example, if the decoder buffer will underflow, we will temporarily increase the sending rate (i.e., making $Rs(k)$ larger than $Ri(K)$) to meet the end-to-end delay constraint of the application. This temporal adjustment of the sending rate will lead to un-TCP-friendliness, and the corresponding change of the amount of sent data will be recorded and compensated later.

4. CROSS-LAYER DESIGN FOR THE WIRELESS SCENARIO

In this section, we discuss how to extend our joint design mechanism for the wired scenario to the wireless scenario. In our previous work, TFRCC uses the same algorithm as TFRC to calculate the TCP-friendly sending rate $B(K)$ and has the same assumption that packet loss is the sign of congestion, so it cannot be directly applied to the wireless scenario. To overcome this problem, we incorporate ARC [2]—an equation-based congestion control mechanism for the wireless scenario—into our framework, which means that we use the same algorithm as ARC to calculate the TCP-friendly rate $B(K)$. Note that any other equation-based congestion control mechanism for the wireless scenario or any work that can differentiate the congestion loss from the erroneous packet loss can also be incorporated into our framework. To differentiate the extended work from the original work of [4], we denote the modified congestion control mechanism as TFRCC-W.

4.1. ARC

ARC is an equation-based mechanism. It models the ideal behavior of the TCP source over lossy links (i.e., reducing the send rate if packet loss is due to congestion, while performing no rate change if packet loss is due to wireless link error), and derives the following throughput formula:

$$B = \frac{s}{4 * \mathrm{RTT}}\left(3+\sqrt{25+24\left(\frac{1-\omega}{\pi-\omega}\right)}\right), \tag{3}$$

where B is the sending rate in bytes/s, s is the packet size, RTT is the round-trip time, ω is the packet loss ratio due to wireless link error, and π is the overall packet loss ratio (including packet losses due to congestion and wireless link error). Then the sender will perform rate control according to (3) to avoid the unnecessary rate reduction due to wireless link error. Note that the overall loss ratio π can be measured at the receiver, and the wireless loss ratio ω can be retrieved from the underlying MAC layer at the sender if the first link is wireless link. For the case in which the sender is not a mobile station, the information regarding the wireless portion of the end-to-end path, that is, the wireless loss ratio ω, should be conveyed to the sender through the feedback.

4.2. Details of TCP-friendly rate calculation

To make the sending rate change smoothly, we need to perform a smooth measurement of the parameters used in (3), which is not discussed in [2]. Here we propose to use the weighted average value over the last N feedback interval to obtain a smooth estimation of the loss ratio. Instead of directly smoothing ω and π, we define the "loss interval" l as

$$l = \frac{1-\omega}{\pi-\omega}, \tag{4}$$

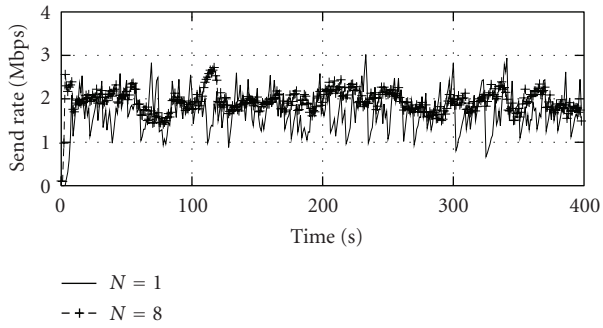


Figure 2: The sending rate of TFRCC-W.

and then compute the average "loss interval" $\hat{l}$[3] as

$$\hat{l} = \sum_{i=1}^{N} m_i * l_i, \tag{5}$$

where m_i is the weight assigned to the ith previous feedback interval,[4] π_i and ω_i are, respectively, the measured overall loss ratio and wireless loss ratio of the ith previous feedback interval, and l_i is computed according to π_i, ω_i, and (4). We first set N to 8, and use the following weights: $m_1, \ldots, m_4 = 1/6$; $m_5 = 2/15$; $m_6 = 0.1$; $m_7 = 1/15$; $m_8 = 1/30$. Then the TCP-friendly sending rate B can be calculated according to (3) and $\hat{l}$, which is computed according to (5) and (4).

We need to deal with the situation where the newest measurement ω^0 is no less than π^0 to avoid "divided-by-zero" problem. This means that there is no packet loss due to congestion within current feedback interval or there is a measurement error. In this case, we cannot directly use (4) any more. So we first let $\omega^0 = \pi^0$, then compute the "loss interval" by combining current interval and last interval. Denote the number of packets sent within current interval and last interval, respectively, as Num^0 and Num_1. Then we update ω_1 and π_1 as follows:

$$\begin{aligned} \omega_1 &= \frac{\omega^0 * \text{Num}^0 + \omega_1 * \text{Num}_1}{\text{Num}^0 + \text{Num}_1}, \\ \pi_1 &= \frac{\pi^0 * \text{Num}^0 + \pi_1 * \text{Num}_1}{\text{Num}^0 + \text{Num}_1}; \end{aligned} \tag{6}$$

l_1 can be updated according to the updated ω_1 and π_1.

Figure 2 shows the sending rate curves of one TFRCC-W flow without parameter smoothing (i.e., $N = 1$) and one flow with parameter smoothing ($N = 8$) when they compete for a bottleneck. It can be found that the protocol has satisfactory performance in terms of rate smoothness by using the proposed measurement mechanism.

[3] Note that the reason of not directly smoothing ω and π is that even with smoothed ω and π, l might sometimes be still not smooth enough to make the sending rate change smoothly, especially when the packet loss ratio due to congestion is very small.

[4] Note that we use the same weighing factors as TFRC [10], which have been proven to have good smoothing effect.

4.3. Performance enhancement of TFRCC-W

The responsiveness and rate smoothness of TFRCC-W depend largely on the sample count N. It is easily understood that with a large value of N, better rate smoothness can be achieved, but the responsiveness of the protocol will deteriorate; while with a small value of N, the protocol will have better responsiveness and worse rate smoothness. Figure 3(a) depicts the sending rate of one TFRCC-W flow with setting N to 3 and one with setting N to 16 when they compete for a bottleneck. It can be easily seen that compared to the flow with setting N to 3, the flow with setting N to 16 has better rate smoothness when the network is underloaded, but shows worse responsiveness when there is a sudden decrease of the available bandwidth.

So we need to investigate how to choose an appropriate value of N to achieve the reasonable tradeoff between the responsiveness and the rate smoothness. Obviously, a constant value of N is not the optimal choice. When the available bandwidth is high, the packet loss ratio is so low that the time interval between two consecutive packet losses is very long. So at this time, N should be sufficiently large to allow sufficient samples of packet loss for effective smoothing. On the other hand, when the available bandwidth is low, the packet loss ratio increases. A large value of N will make the packet loss samples too large to reflect the recent network condition, thus deteriorate the responsiveness of the protocol. At this time, the value of N is expected to be small.

Let us first see how TFRC [10] resolve this problem. TFRC uses the following formula to calculate the TCP-friendly sending rate:

$$B = \frac{s}{R\sqrt{2p/3} + 4R \min\left(1, 3\sqrt{3p/8}\right) p(1 + 32p^2)}, \tag{7}$$

where B is the sending rate in bytes/s, s is the packet size, R is the round-trip time, p is the steady-state loss event rate.

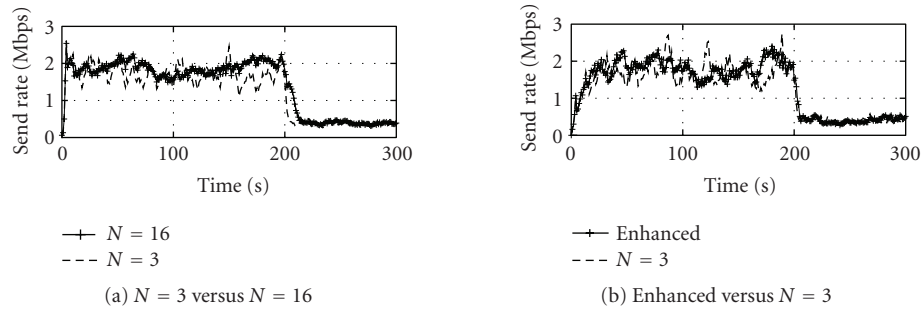
A loss event is defined as the first packet loss every one RTT, and p is measured in terms of loss intervals, spanning the number of packets between consecutive loss events. The most recent M (usually set to 8) loss intervals are averaged, using decaying weights to get the smoothed value $\hat{l}_t$. The loss event rate p is calculated as the inverse of the average loss interval $\hat{l}_t$.

The average time interval between two consecutive sending packets, t_p, is the inverse of the packet rate (packet/s), that is, B/s. So we have

$$t_p = \frac{1}{B/s} = R\sqrt{\frac{2p}{3}} + 4R \min\left(1, 3\sqrt{\frac{3p}{8}}\right) p(1 + 32p^2). \tag{8}$$

Let us denote the average time interval between two consecutive loss events as t_l. Then we can get

$$\begin{aligned} t_l &= \hat{l}_t * t_p = t_p / p \\ &= R\sqrt{\frac{2}{3p}} + 4R \min\left(1, 3\sqrt{\frac{3p}{8}}\right)(1 + 32p^2). \end{aligned} \tag{9}$$



(a) $N = 3$ versus $N = 16$ (b) Enhanced versus $N = 3$

Figure 3: Performance comparison between TFRCC-W flows with different values of N.

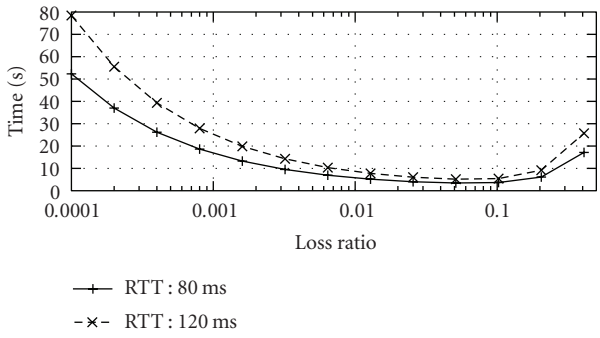


Figure 4: The sampling time length of TFRC.

So we can get the sampling time length of TFRC as

$$L_{\text{TFRC}} = 8 * \left(R\sqrt{\frac{2}{3p}} + 4R \min\left(1, 3\sqrt{\frac{3p}{8}}\right)(1 + 32p^2)\right). \tag{10}$$

Then we plot the $L_{\text{TFRC}} - p$ curve (see Figure 4) when RTT is, respectively, set to 80 milliseconds and 120 milliseconds. From Figure 4, we can find that the sampling time length of TFRC is variable and depend on the network condition, which helps TFRC achieve reasonable tradeoff between the rate smoothness and responsiveness.

So in the enhanced version of TFRCC-W, we adopt the same sampling time length as TFRC, that is, first calculate the sample time length of TFRC according to (10),[5] then determine N according to the derived time length value. Figure 3(b) depicts the sending rate of one enhanced TFRCC-W flow and one with N of 3 when they compete for a bottleneck in the same scenario as in Figure 3(a). It can be easily seen that the enhanced TFRCC-W has better overall performance in terms of rate smoothness and responsiveness.

[5] Note that in (10), p means the packet loss ratio only due to congestion. So in the computation, we do not use the overall packet loss ratio π, but $1/l$, which means the packet loss ratio due to congestion (see details in [2]).

5. PERFORMANCE EVALUATION AND ENHANCEMENT FOR OUR PROPOSED CROSS-LAYER MECHANISM

5.1. Performance evaluation of TFRCC-W

We use NS-2 simulator [27] to investigate the performance of TFRCC-W. Note that in this subsection, we mainly concern about the performance of the transport layer. So TFRCC-W adapts its sending rate only according to (3), and does not consider how to support the QoS requirements of the application (i.e., as FTP applications run). We use the well-known dumbbell topology, where one TFRC or TFRCC-W flow and several TCP flows compete for one bottleneck link with a capacity of 15 Mbps and a transmission delay of τ millisecond. We assume that the senders of TFRC or TFRCC-W flows connect to the bottleneck via wireless links with the wireless loss ratio p_w, that is, the first link of every TFRC/TFRCC-W flow is wireless. The feedback interval of TFRCC-W is fixed to 1 second. We set the packet drop ratio in the bottleneck caused by congestion to be 0.001, and the transmission delay τ is, respectively, set to 50 milliseconds and 100 milliseconds. Then we record the average throughput of one TFRCC-W and one TFRC flows when they, respectively, run through the bottleneck with the wireless loss ratio p_w increasing from $1e-6$ to 0.01. Simulation results are depicted in Figure 5, and every point in the figure is the average value of ten runs.

The results show that TFRC and TFRCC-W have similar throughput when the wireless loss ratio is very low, which means that TFRCC-W has good TCP-friendliness in the wired scenario. With the increasing of the wireless loss ratio, TFRCC-W shows much better performance than TFRC, and can effectively avoid the throughput degradation caused by the wireless link error.

5.2. Performance evaluation of the proposed cross-layer mechanism

In this subsection, we investigate the performance of our proposed cross-layer design mechanism using NS-2 simulation. The simulation topology is depicted in Figure 6, where m multimedia mobile terminals are connected to the IP backbone via wireless access point. In the backbone, there are three links (R1-R2, R2-R3, and R3-R4), each of which has a capacity of R Mbps and a transmission delay of τ millisecond. All the wireless links have the same loss ratio of 0.5%. To

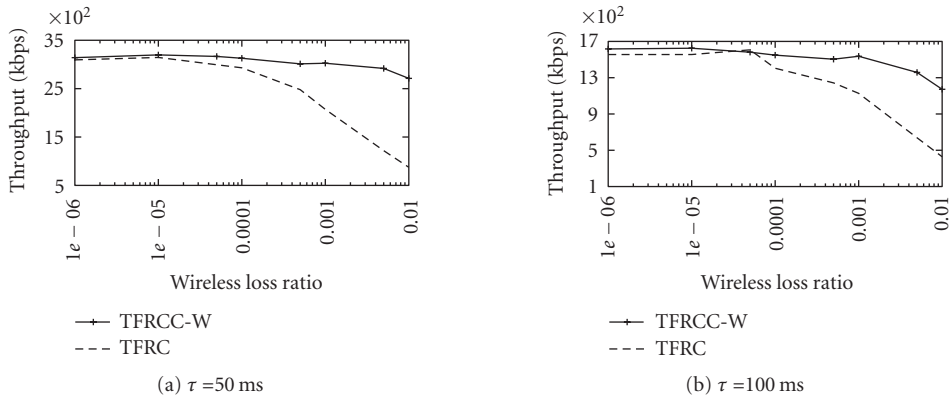


Figure 5: Performance comparison between TFRCC-W and TFRC in the wireless scenario.

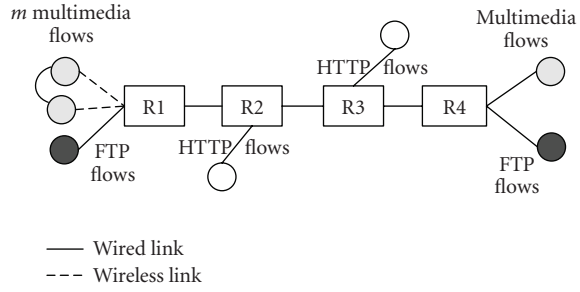


Figure 6: The simulation topology.

simulate the real Internet environment, various cross-traffic (FTP flows and WWW flows) combinations have been used. The FTP flows are modeled as greedy sources which can always send data at the available bandwidth. The WWW flows are modeled as on-off processes.

The standard video sequence of "Foreman" (300 frames) in QCIF format is circularly used as the video source, and is encoded using an MPEG-4 fine granularity scalable (FGS) coder [28]. The encoder uses interframe coding (with the GOP size of 10 and the frame type of I and P) and quantization step size of 31 to generate the base layer, which provides the minimum video quality. Then the FGS coder generates the embedded enhancement layer bitstream, which can be cut off at any bit to adapt the source rate with fine granularity. The frame rate is set to 25 frames per second (fps). The maximum necessary PSNR is set to 40 dB. We packetize the base layer and enhancement layer separately, and the MSS (maximum segment size) is set to 1000 bytes. In this paper, we use a simple error resilience algorithm. If the base layer of some frame is lost or late, the base layer of the previous frame will be used in decoding. If there is a packet loss in the enhancement layer, all less important packets in that frame will be discarded as they all depend on the lost, more important packet. Note that in MPEG-4 FGS, loss of enhancement layer packets is not going to cause distortion propagation to subsequent frames.

We compare the performance of three source rate/congestion control algorithms. One uses the global rate control model proposed in [22] with TFRC as the congestion control mechanism, denoted as GM-TFRC, and one uses the global model and ARC [2], denoted as GM-ARC. The other is our proposed algorithm, denoted as VB-TFRCC-W. Note that GM-TFRC and GM-ARC belong to traditional separate design approaches. For fair comparisons, all of the congestion control mechanisms have the same feedback interval of 1 second. In Section 5.2.1, we mainly intend to show that VB-TFRCC-W can provide better QoS support for the application. The overall system performance will be evaluated in Section 5.2.2.

5.2.1. Support of the QoS requirements of the application

In this scenario, each of the three links (R1-R2, R2-R3 and R3-R4) has a capacity of 14 Mbps, a transmission delay of 40 milliseconds, and an RED queue with the maximum threshold of 120 packets. Mobile terminals are supposed to be the receivers of the multimedia flows, that is, the last links of all the multimedia flows are wireless. The startup delay is set to a large value, that is, 125 frames (5 seconds), and the encoder and decoder buffer sizes are both set to 400 kB. We adopt a dynamic scenario, which lasts 600 seconds. There are 3 GM-TFRC flows, 3 GM-ARC flows, 3 VB-TFRCC-W flows, and 3 FTP flows running throughout the entire simulation. As the background flows, 100 FTP flows join at around 350 seconds, and 10 WWW flows join at 300 seconds. Here we use the average PSNR and PSNR deviation to evaluate the video quality, where the average PSNR deviation of one video sequence is calculated by averaging the PSNR difference between every two adjacent frames.

Because TFRC is mainly designed for the wired channels, it will reduce the sending rate as long as there is one packet loss. As a result, it shows poor TCP-friendliness when there exist wireless packet losses (see Figure 8). Consequently, the low throughput leads to poor video quality (see Table 1[6]). ARC and TFRCC-W, on the other hand, can take

[6] Note that all the entries in Table 1 are the average value of all the flows using the same source rate/congestion control algorithm.

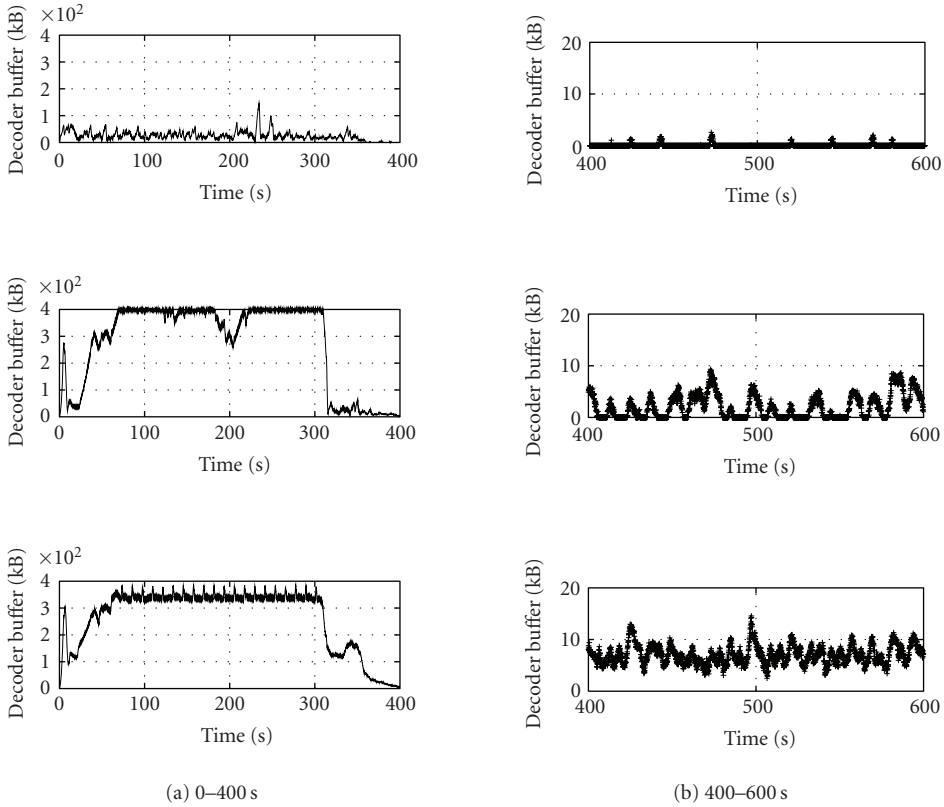


Figure 7: The decoder buffer occupancies of one GM-TFRC flow, one GM-ARC flow, and one VB-TFRCC-W flow; top: GM-TFRC, middle: GM-ARC, bottom: VB-TFRCC-W.

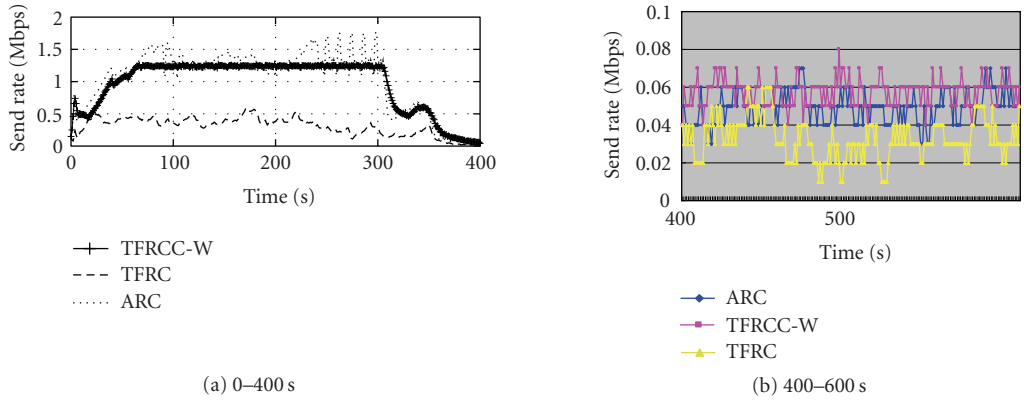


Figure 8: The sending rate of one GM-TFRC flow, one GM-ARC flow, and one VB-TFRCC-W flow.

into account the effect of wireless losses, and achieve higher throughput.

Furthermore, our cross-layer design approach can provide better support for the QoS requirement of the application than GM-ARC. Within the first 300 seconds, the network is underloaded, and the available bandwidth may occasionally be higher than the maximum admissible sending rate constrained by buffer sizes (see the discussion in Section 3.2.2). So from Figure 7, we can find that the decoder buffer of GM-ARC overflows. VB-TFRCC-W, on the other hand, takes into account this sending rate constraint (see Figure 8), and successfully avoids the decoder buffer overflow. With the joining of 100 FTP flows around 350 seconds, the available bandwidth becomes so low that source rate control alone cannot guarantee the end-to-end delay constraint being met because of the minimum bandwidth requirement and quality smoothness constraint of the video source. So the decoder buffer underflow occurs for GM-ARC (see Figure 7). However VB-TFRCC-W can meet the end-to-end delay constraint by making the sending rate temporarily larger than

Table 1: PSNR of GM-TFRC, GM-ARC, and VB-TFRCC-W.

	Sender side		Receiver side	
	PSNR	Variation	PSNR	Variation
GM-TFRC	29.67	0.26	24.70	0.35
GM-ARC	33.75	0.2	30.05	0.97
VB-TFRCC-W	33.51	0.17	32.74	0.57

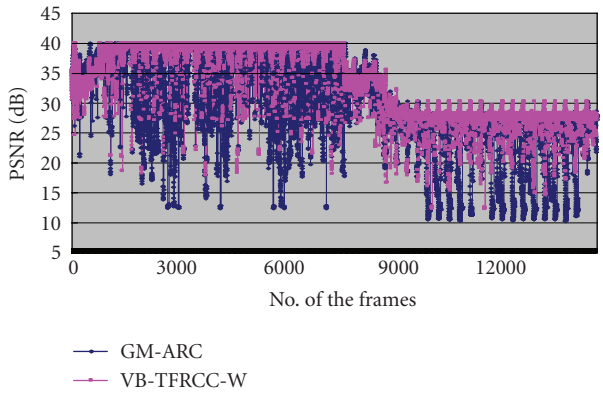


Figure 9: The PSNR curve of the sequences decoded by the decoder of one GM-ARC flow and one VB-TFRCC-W flow.

the TCP-friendly value when necessary (see Figure 8). So VB-TFRCC-W can almost avoid the decoder buffer underflow. As a result, VB-TFRCC-W can significantly reduce the video quality degradation due to lost/late packets between the sender and the receiver, and achieve better playback quality (higher average PSNR and smoother PSNR variation) than GM-ARC (see Table 1 and Figure 9). Note that very low PSNR values (e.g., less than 25 dB) in Figure 9 typically indicate an effective loss of base layer packet for a frame, which introduces significant quality degradation for the lost frame and the subsequent frames.

5.2.2. *Overall system performance evaluation*

We make a comparison for the overall system performance of GM-ARC and VB-TFRCC-W. Each of the three links (R1-R2, R2-R3, and R3-R4) has a capacity of 10 Mbps. We would like to test the overall system performance, respectively, under RED queues and under Drop Tail queues. When the queues of the three links (R1-R2, R2-R3 and R3-R4) are RED, the startup delay is set to a small value, that is, 15 frames, and the encoder and decoder buffer sizes are both set to 100 kB. When the queues are Drop Tail, the startup delay is set to 125 frames (5 seconds), and the encoder and decoder buffer sizes are both set to 500 kB. Mobile terminals are supposed to be the senders of the multimedia flows, that is, the first links of all the multimedia flows are wireless. In this scenario, we would like to simulate the scenario where the available bandwidth is very low. The simulation lasts 600 seconds. There are 5 GM-ARC flows, 5 VB-TFRCC-W flows, and 5 FTP flows running throughout the entire simulation. As the background flows, 70 FTP flows join at 50 seconds, and depart at 300 seconds.

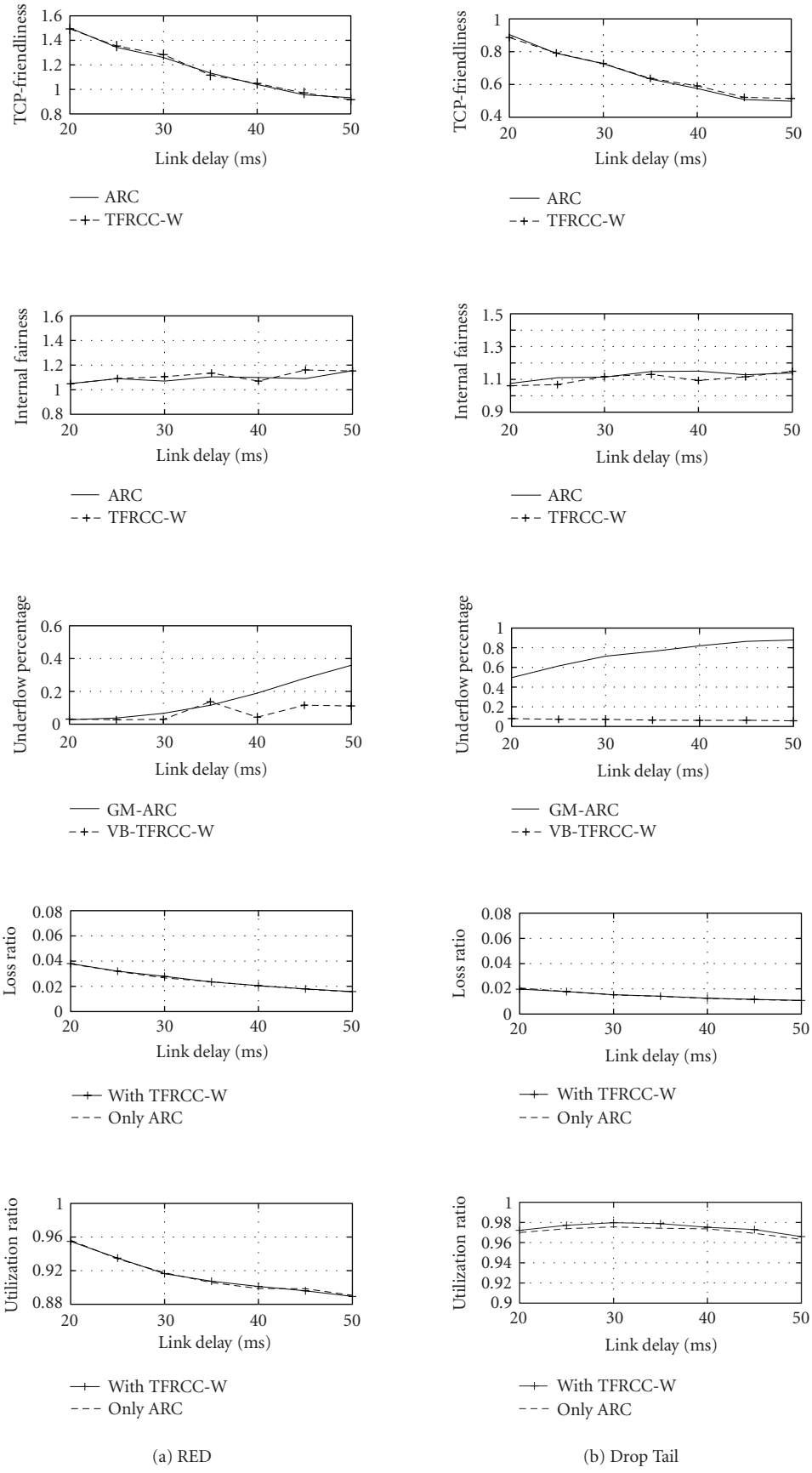
To evaluate the long-term TCP-friendliness and internal fairness (i.e., the fairness among the flows using the same congestion control mechanism) of the transport protocol, we adopt the metrics defined in [29, Chapter 4], where a value close to 1 indicates a good TCP-friendliness or internal fairness. The underflow percentage of the decoder buffer between 50 seconds to 300 seconds (i.e., when the available bandwidth is low), is used to evaluate the support for the end-to-end delay constraint of the application. The underflow percentage of the decoder buffer is computed as the percentage of the frames which are lost or arrive at the decoder buffer later than the prescribed time. We run the simulations under different link transmission delay τ (varying from 20 milliseconds to 50 milliseconds), and simulation results are depicted in Figure 10. Note that every point in Figure 10 is the average value of 5 runs. From the simulation results, it can be found that with the increase of the link delay τ, the TCP-friendliness (i.e., the throughtput) of GM-ARC decreases for both the RED case and the DropTail case. So the underflow percentage of the decoder buffer increases for GM-ARC. VB-TFRCC-W, on the other hand, can provide better QoS support for the application and maintain the underflow percentage of the decoder buffer at a low level, although its throughput also decreases. For the performance of the transport protocol, VB-TFRCC-W shows similar long-term TCP-friendliness and internal fairness as GM-ARC.

We also concern about how the overall network performance is affected with different mechanisms. Here we use the overall packet loss ratio introduced in *the wired channels*, and the utilization ratio of the bottleneck bandwidth to evaluate the overall network performance. We replace 5 VB-TFRCC-W flows with 5 GM-ARC flows (which means that there are 10 GM-ARC flows in the simulations) and repeat the above simulations. Then we compare the results to the previous simulation results when there exist VB-TFRCC-W flows. From Figure 10, we can find that there is almost no difference in the overall network performance between using GM-ARC and VB-TFRCC-W. So our proposed algorithm will not deteriorate the overall network performance although it sometimes exhibits temporal un-TCP-friendly behaviors.

5.3. *Enhancement of the cross-layer mechanism*

Our cross-layer mechanism allows the sending rate to temporarily violate TCP-friendliness to support the QoS requirements of the application, as described in Section 3. When the TCP-friendly bandwidth is too low to make sure the end-to-end delay constraint of the application is met, the sending rate can be temporarily larger than the TCP-friendly value to help the application meet the end-to-end delay constraint, which effectively prevent the decoder buffer from underflow.

However this does not work well under all the network scenarios. Let us suppose that the network is congested and the TCP-friendly bandwidth is B. The minimum bandwidth which can help the application to meet the end-to-end delay constraint is assumed to be RI ($RI > B$). Suppose that the application adopts our cross-layer mechanism and sends the data with the rate of RI. Obviously,



(a) RED

(b) Drop Tail

Figure 10: Overall system performance evaluation between GM-ARC and VB-TFRCC-W.

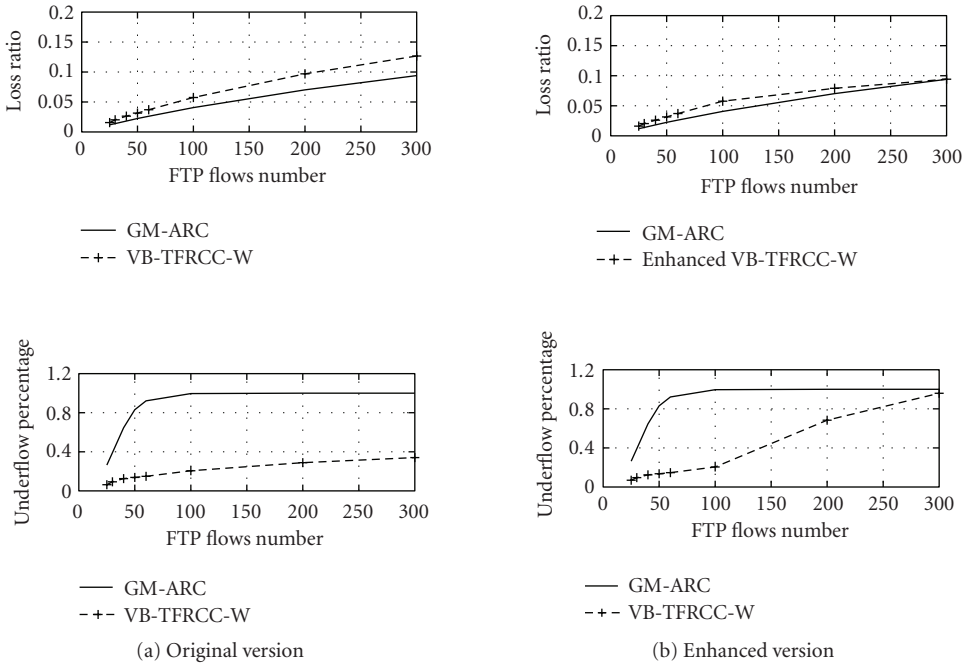


Figure 11: Performance evaluation for the enhanced VB-TFRCC-W.

our cross-layer mechanism can achieve good performance only when the additional sent data beyond the TCP-friendly value (i.e., $RI - B$) can go through the network and be received by the receiver. However, when the network is heavily congested, its effective throughput may not afford such additional data going through the network, which cannot prevent the decoder buffer from underflow. On the other hand, such additional data will deteriorate network congestion. We use *ns*-2 simulation to evaluate the above analysis. The simulation environment is the same as in Section 5.2. The capacity R of the three links (R1-R2, R2-R3, and R3-R4) is set to 15 Mbps, and the Drop Tail queue is used. Ten GM-ARC or VB-TFRCC-W flows compete the bottleneck link with N FTP flows (N increases from 25 to 300), and we record the underflow percentage of decoder buffer and the overall packet loss ratio in the wired links, respectively, when GM-ARC flows run and when VB-TFRCC-W flows run. The simulation results are depicted in Figure 11(a), and note that every point in the figure is the average value of three runs. We can find that when the network is slightly or moderately congested (i.e., N is not very large), our cross-layer mechanism can maintain the underflow percentage of the decoder buffer at a low level, as opposed to GM-ARC. But with the increasing of N, the network becomes extremely congested and cannot meet the minimum bandwidth requirement of multimedia flows. Thus the decoder buffer underflow percent can no longer be maintained at an acceptable level even with our mechanism. Meanwhile, additional data (i.e., $RI - B$) sent by our mechanism will deteriorate network congestion and lead to the significant increasing of the overall packet loss ratio in the network.

The above analysis and simulation results show that our cross-layer design is not suitable for such extreme congestion scenario. To solve this problem, we add an "intelligent switching" function, which can switch between the cross-layer design mode and layered design mode (i.e., the TCP-friendly state) according to the network congestion level. Here a good metric to evaluate the network congestion level is the underflow percentage of the decoder buffer, which actually indicates how much the network can meet the bandwidth requirement of the multimedia application. If the recent decoder buffer underflow percentage is larger than P_H,[7] it means that the network might be too congested to support the minimum bandwidth requirement of the application, and the sender will be switched from the cross-layer mode to the layered design mode (i.e., setting the sending rate to the TCP-friendly value). Then if the recent decoder buffer underflow percentage is below P_L (set to 0.1 in the simulations), it means that the network has recovered from the serious congestion, and the sender will return back to the cross-layer design mode. We repeat the simulation by using the enhanced version of VB-TFRCC-W, and the results are shown in Figure 11(b). We can find that the enhanced version can make sure the network performance (e.g., the overall packet loss ratio in the wired links) is not degraded by intelligent switch to the TCP-friendly state during the serious network congestion.

[7] Note that P_H indicates the allowed maximum value of the decoder buffer underflow percentage, which is determined by the minimum acceptable video quality of the application. In the simulations, P_H is set to 0.4.

6. CONCLUSION AND FUTURE WORK

This paper proposes cross-layer design of source rate control and congestion control for wireless video streaming. With a joint decision of the source rate and sending rate by taking into account the information from the application layer, the transport layer, and the MAC layer, the proposed cross-layer design approach can effectively avoid throughput degradation caused by wireless link error, and help the multimedia application achieve better playback quality, while maintaining good performance of the transport protocol.

In this paper, we mainly use the simulation to evaluate the performance of the proposed mechanism. Next we will try to implement our mechanism in the real wireless network to evaluate its practical performance. Although we incorporate ARC into our framework for extension to wireless, it should be noted that other rate control schemes for wireless can also be incorporated, for example, MULTFRC [17] and AIO-TFRC [18], which belong to pure end-to-end approaches and do not need the cross-layer information. It is a very interesting topic to study how to combine MULTFRC or AIO-TFRC, and our proposed framework.

REFERENCES

[1] S. Jacobs and A. Eleftheriadis, "Streaming video using TCP flow control and dynamic rate shaping," *Journal of Visual Communication and Image Representation*, vol. 9, no. 3, pp. 211–222, 1998.

[2] Ö. B. Akan and I. F. Akyildiz, "ARC: the analytical rate control scheme for real-time traffic in wireless networks," *IEEE/ACM Transactions on Networking*, vol. 12, no. 4, pp. 634–644, 2004.

[3] M. van der Schaar and S. Shankar, "Cross-layer wireless multimedia transmission: challenges, principles, and new paradigms," *IEEE Wireless Communications*, vol. 12, no. 4, pp. 50–58, 2005.

[4] P. Zhu, W. Zeng, and C. Li, "Joint design of source rate control and QoS-aware congestion control for video streaming over the Internet," *IEEE Transactions on Multimedia*, vol. 9, no. 2, pp. 366–376, 2007.

[5] J. Widmer, R. Denda, and M. Mauve, "A survey on TCP-friendly congestion control," *IEEE Network*, vol. 15, no. 3, pp. 28–37, 2001.

[6] D. Bansal and H. Balakrishnan, "Binomial congestion control algorithms," in *Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '01)*, vol. 2, pp. 631–640, Anchorage, Alaska, USA, April 2001.

[7] N. R. Sastry and S. S. Lam, "CYRF: a theory of window-based unicast congestion control," *IEEE/ACM Transactions on Networking*, vol. 13, no. 2, pp. 330–342, 2005.

[8] R. Rejaie, M. Handley, and D. Estrin, "RAP: an end-to-end rate-based congestion control mechanism for realtime streams in the Internet," in *Proceedings of the 18th Annual Joint Conference of the IEEE Computer and Communications Societie (INFOCOM '99)*, vol. 3, pp. 1337–1345, New York, NY, USA, March 1999.

[9] D. Sisalem, *TCP-friendly congestion control for multimedia communication in the Internet*, Ph.D. thesis, Technical University of Berlin, Berlin, Germany, 2000.

[10] M. Handley, S. Floyd, J. Padhye, and J. Widmer, "TCP friendly rate control (TFRC): protocol specification," IETF RFC 3448, January 2003.

[11] Y.-G. Kim, J. Kim, and C.-C. Jay Kuo, "TCP-friendly Internet video with smooth and fast rate adaptation and network-aware error control," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 2, pp. 256–268, 2004.

[12] 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, pp. 703–717, 2003.

[13] F. Yang, Q. Zhang, W. Zhu, and Y.-Q. Zhang, "End-to-end TCP-friendly streaming protocol and bit allocation for scalable video over wireless Internet," *IEEE Journal on Selected Areas in Communications*, vol. 22, no. 4, pp. 777–790, 2004.

[14] G. Cheung and T. Yoshimura, "Streaming agent: a network proxy for media streaming in 3G wireless networks," in *IEEE International Packet Video Workshop*, Pittsburgh, Pa, USA, April 2002.

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

[16] D. Barman and I. Matta, "Effectiveness of loss labeling in improving TCP performance in wired/wireless network," in *Proceedings of the 10th IEEE International Conference Network Protocols (ICNP '02)*, pp. 2–11, Paris, France, November 2002.

[17] M. Chen and A. Zakhor, "Multiple TFRC connections based rate control for wireless networks," *IEEE Transactions on Multimedia*, vol. 8, no. 5, pp. 1045–1062, 2006.

[18] M. Chen and A. Zakhor, "AIO-TFRC: a light-weight rate control scheme for streaming over wireless," in *Proceedings of the International Conference on Wireless Networks, Communications and Mobile Computing (WirelessCom '05)*, vol. 2, pp. 1124–1129, Maui, Hawaii, USA, June 2005.

[19] M. Kalman, E. Steinbach, and B. Girod, "Adaptive media playout for low-delay video streaming over error-prone channels," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 6, pp. 841–851, 2004.

[20] E. G. Steinbach, N. Färber, and B. Girod, "Adaptive playout for low latency video streaming," in *Proceedings of IEEE International Conference on Image Processing (ICIP '01)*, vol. 1, pp. 962–965, Thessaloniki, Greece, October 2001.

[21] B. Xie and W. Zeng, "Rate-distortion optimized dynamic bitstream switching for scalable video streaming," in *IEEE International Conference on Multimedia and Expo (ICME '04)*, vol. 2, pp. 1327–1330, Taipei, Taiwan, June 2004.

[22] J. Viéron and C. Guillemot, "Real-time constrained TCP-compatible rate control for video over the Internet," *IEEE Transactions on Multimedia*, vol. 6, no. 4, pp. 634–646, 2004.

[23] C. E. Luna, Y. Eisenberg, R. Berry, T. N. Pappas, and A. K. Katsaggelos, "Joint source coding and data rate adaptation for energy efficient wireless video streaming," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 10, pp. 1710–1720, 2003.

[24] T. Ahmed, A. Mehaoua, R. Boutaba, and Y. Iraqi, "Adaptive packet video streaming over IP networks: a cross-layer approach," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 2, pp. 385–401, 2005.

[25] J. Yan, K. Katrinis, M. May, and B. Plattner, "Media- and TCP-friendly congestion control for scalable video streams," *IEEE Transactions on Multimedia*, vol. 8, no. 2, pp. 196–206, 2006.

[26] C. Chen, Z.-G. Li, and Y.-C. Soh, "TCP-friendly source adaptation for multimedia applications over the Internet," in *Proceedings of the 15th International Packet Video Workshop (PV '06)*, Hangzhou, China, April 2006.

[27] S. Floyd and S. McCanne, "Network Simulator, LBNL public domain software," http://www.isi.edu/nsnam/ns/.

[28] H. M. Radha, M. van der Schaar, and Y. Chen, "The MPEG-4 fine-grained scalable video coding method for multimedia streaming over IP," *IEEE Transactions on Multimedia*, vol. 3, no. 1, pp. 53–68, 2001.

[29] J. Padhye, *Towards a comprehensive congestion control framework for continuous media flows in best effort networks*, Ph.D. thesis, University of Massachusetts Amherst, Amherst, Mass, USA, 2000.