CROSS-LAYER DESIGN OF SOURCE RATE CONTROL AND QOS-AWARE CONGESTION CONTROL FOR WIRELESS VIDEO STREAMING

Peng Zhu
Henan Key Lab of Infor. Elect. Apparatus
Tsinghua University
zhupeng00@mails.tsinghua.edu.cn

Wenjun Zeng
Dept. of Computer Science
Univ. of Missouri-Columbia
zengw@missouri.edu

Chunwen Li
Dept. of Automation
Tsinghua University
lcw@mail.tsinghua.edu.cn

ABSTRACT

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 (i.e. joint design of source rate control and congestion control for video streaming over the Internet) in [1] and propose a cross-layer design approach for wireless video streaming. By jointly designing the source rate control at the application layer and congestion control at the transport layer, and taking advantage of MAC layer information, our approach can avoid the throughput degradation caused by transmission error of the wireless channel, and better support the QoS requirements of the application. Simulation results show that the proposed mechanism can significantly improve the playback quality of the application, while maintaining good performance of the transport protocol.

1. INTRODUCTION

Multimedia streaming over the wireless channels has been a very challenging issue due to the dynamic uncertain nature (e.g., variable available bandwidth and random packet loss) of the wireless channels. 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.

Congestion control for streaming media has to take care of not only the fairness and responsiveness of the transport protocol, but also the rate smoothness to help the multimedia application achieve better playback quality[2]. A number of TCP-friendly congestion control schemes designed for wired channels have been proposed to provide smoother send rate. These include the window-based schemes[3] and the rate-based schemes which can be further classified into the probe-based[4] and equation-based schemes[5].

However these mechanisms can not be directly applied to the wireless channels. A common limitation of the above approaches is that they all assume that every packet loss is an indication of congestion, which is not true for the wireless channels, as in the wireless scenario, packet losses can also be attributed to link error. One way to overcome this problem is to distinguish packet losses due to link errors from those due to congestion, as proposed in [6]. However to obtain high accuracy in determining the actual source of packet loss is challenging. Akan et. al proposed an equation-based approach - the analytical rate control scheme (ARC) for multimedia traffic in wireless networks[7], which only needs the statistic information of wireless losses.

Source rate control at the application layer is to adaptively adjust the source rate based on the channel condition to achieve better video quality. Many mechanisms[8] have been proposed in the past. 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 mechanisms for streaming multimedia usually need to smooth their send rate variation 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. The end-to-end delay constraint of multimedia applications also imposes constraints on the send rate, because source rate control alone can not guarantee the end-to-end delay constraint due to the minimum bandwidth requirement and the quality smoothness constraint of the video source. However these constraints can not be considered by most congestion control mechanisms due to the separate design principle.

The cross-layer design approach, which allows layers to have more communication with each other, on the other hand, can achieve better overall system performance. Many cross-layer design solutions for streaming video over the wireless have been proposed [9]. However, most of them mainly concern about how to utilize the information from the MAC/physical layer. In [1], we propose a joint design algorithm of source rate control at the application layer and congestion control at the transport layer for streaming video over the Internet. However the congestion control mechanism is designed for the wired channels and thus can not be directly applied to the wireless scenario.

In this paper, we extend our work in [1] to the wireless scenario by incorporating ARC[7] into our joint design architecture. Simulation results show that our approach can significantly improve the video quality of the application while maintaining good performance of the transport layer.

The rest of this paper is organized as follows. In Section 2, we briefly introduce our joint design work in [1]. Section 3 describes how to extend our algorithm to the wireless scenario in detail. Simulation results are presented in Section 4. Section 5 gives the concluding remarks.

2. PREVIOUS WORK FOR THE WIRED SCENARIO

2.1. The system architecture

The system architecture is illustrated in Fig. 1. At the transport layer, we proposed a QoS-aware congestion control mechanism, named TFRCC (TCP-friendly rate control with compensation), based on the well-known work of TFRC[5]. 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 of the protocol can be preserved by introducing a rate compensation algorithm. At the application layer, the virtual network buffer management mechanism, denoted as VB, is used to translate the QoS

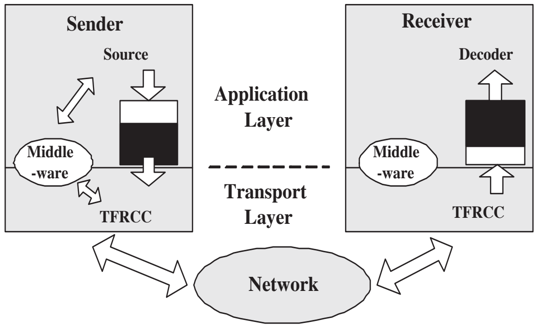


Fig. 1. The system architecture

requirements of the application to the constraint of the source rate and send rate. There is a middleware component located between the application layer and the transport layer. The joint decision of the source rate and the send rate is done within the middleware at the sender.

2.2. The Joint Design Algorithm

Next we will briefly introduce the joint design algorithm. One can refer to [1] 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 $Bv(k)$, 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 Eq. (1) by selecting appropriate source and send rates, the overflow and underflow of the encoder and decoder buffers can be avoided.

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

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

$$B_u = min(N * Ri(K) - Bv(K), BE) - M_1 * Ri(K) \quad (2)$$
$$B_l = max(0, N * Ri(K) - Bv(K) - BD) + M_2 * Ri(K)$$

where M_1 and M_2 are two non-negative safety margins. Note that the above constraints are derived by VB at the application layer.

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

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

A. Decision of the source rate and the send rate:

The actual send rate $Rs(k)$ is usually set to $Ri(K)$ for good TCP-friendliness. The source rate is determined so as to maintain the encoder buffer within the bounds of Eq.(2), while considering the video quality smoothness constraint, the minimum acceptable PSNR of the video source.

B. 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 send rate is updated as $Ri(K+1)$. Consequently, at times $k_1 - N$,...,$k_1 - 1$, the estimation of the future receive rates using $Ri(K)$ might not have been accurate and the constraints of Eq. (1) might not actually be met. So if necessary, the readjustment of the size of the encoded frame $k_1 - N$,...,$k_1 - 1$ (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,...,$k_1 + N - 1$ will not underflow and overflow. If this still can not prevent the decoder buffer from underflow or overflow, we will have to adjust the send 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 send 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 send rate will lead to un-TCP-friendliness, and the corresponding change of amount of send data which is caused by the temporal rate adjustment will be recorded and compensated later.

3. EXTENSION TO THE WIRELESS SCENARIO

TFRCC uses the same algorithm as TFRC to calculate the TCP-friendly send rate $B(K)$, so it can not be directly applied to the wireless channels. To overcome this problem, we incorporate ARC[7] into our architecture, i.e., we use the same algorithm as ARC to calculate the TCP-friendly rate $B(K)$. To differentiate the extended work from the original work of [1], we denote the modified congestion control mechanism as TFRCC-W.

3.1. ARC

ARC is an equation-based mechanism. It first models the ideal behaviors 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 gets the following throughput formula:

$$B = \frac{s}{4 * RTT}(3 + \sqrt{25 + 24(\frac{1-\omega}{\pi - \omega})}) \quad (3)$$

where B is the send rate in bytes/sec, s is the packet size, RTT is the round-trip time, ω is the wireless link packet loss ratio 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 Eq. (3) to avoid the un-necessary rate reduction due to wireless link error and achieve TCP-friendliness. 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 that the sender is not mobile station, the information regarding the wireless portion of the end-to-end path, i.e. the wireless loss ratio ω, should be conveyed to the sender through the feedback.

3.2. Details of TCP-friendly rate calculation

To make the send rate change smoothly, we need to perform a smooth measurement of the parameters used in Eq.(3), which is not discussed in [7]. 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:

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

and then compute the average "loss interval" $\hat{l}$ 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, 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 Eq.(4). In this paper, we set N to 8, and use the following weights: $m_1, .., m_4 = 1/6$;$m_5 = 2/15$;$m_6 = 0.1$;$m_7 = 1/15$;$m_8 = 1/30$. Then the TCP-friendly send rate B can be calculated according to Eq.(3) and $\hat{l}$.

We need to deal with the situation where the newest measurement ω^0 is no less than π^0. 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 can not directly use Eq. (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:

$$\omega_1 = \frac{\omega^0 * Num^0 + \omega_1 * Num_1}{Num^0 + Num_1}$$
$$\pi_1 = \frac{\pi^0 * Num^0 + \pi_1 * Num_1}{Num^0 + Num_1}$$

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

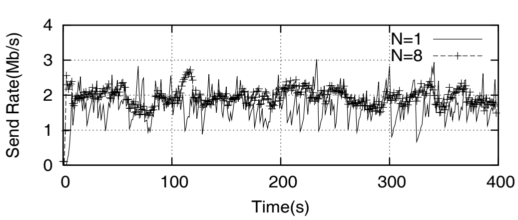


Fig. 2. The send rate of TFRCC-W

Figure 2 shows the send 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.

4. SIMULATION RESULTS

We compare the performance of three source rate/congestion control algorithms using NS-2 simulation. One uses the global rate control model proposed in [8] with TFRC as the congestion control mechanism, denoted as GM-TFRC, and one uses the global model and ARC[7], 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.

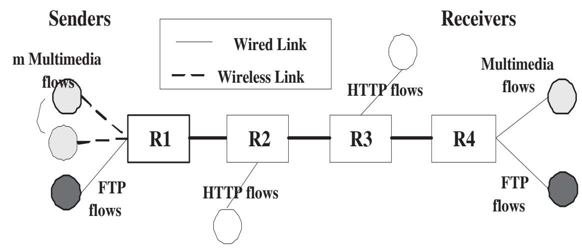


Fig. 3. The simulation topology

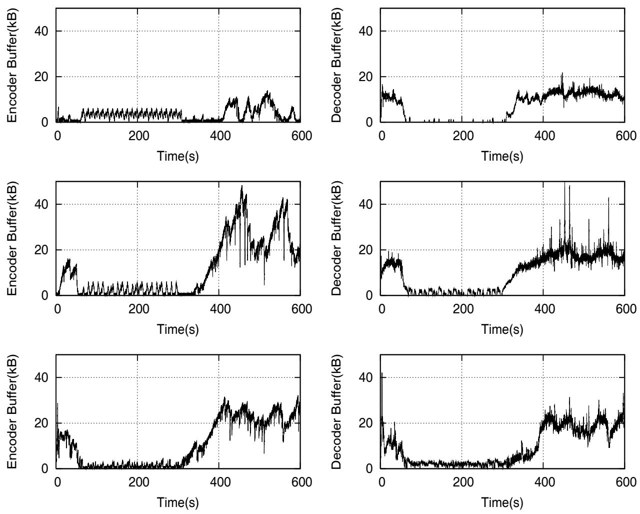
Table 1. The PSNR results (measured at the sender and receiver) and the network fairness (TCP-friendliness (TF) and internal fairness (IF))

	Average PSNR (dB)		Network fairness	
	Sender	Receiver	TF	IF
GM-TFRC	29.22	24.37	0.51	1.06
GM-ARC	31.41	29.14	0.88	1.08
VB-TFRCC-W	31.27	30.66	0.88	1.02

We use the simulation topology depicted in Fig. 3, where m multimedia flows 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 10Mb/s, a transmission delay of $40ms$, and a RED queue with the maximum threshold of 120 packets. All the wireless links have the same loss ratio of 0.5%. The standard video sequence of "foreman" (300 frames) in QCIF format is circularly used as the video source, and is encoded using a MPEG-4 Fine Granularity Scalable (FGS) coder. The encoder uses the interframe coding (with the GOP size of 10 and the frame type of I and P) and the quantization stepsize of 31 to generate the base layer, which provides the minimum video quality. Then the FGS coder generates the enhancement layer, which can be cut off at any point to adapt the source rate. 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. The maximum necessary PSNR is set to 40dB. The frame rate is set to 25 frames per second (fps). The start up delay is set to 15 frames. The encoder and decoder buffer sizes are both set to 100kB. We packetize the base layer and enhancement layer separately, and the MSS (Maximum Segment Size) value is set to 1000 bytes. For fair comparisons, all of the congestion control mechanisms have the same feedback interval of 1 second.

We adopt a dynamic scenario, which lasts 600s. 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, 70 FTP flows join at 50s and depart at 300s, and 5 WWW flows join at 0s and depart at 300s.

Because TFRC is mainly designed for the wired channels, it



(a) Encoder Buffer (b) Decoder Buffer

Fig. 4. The encoder and 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

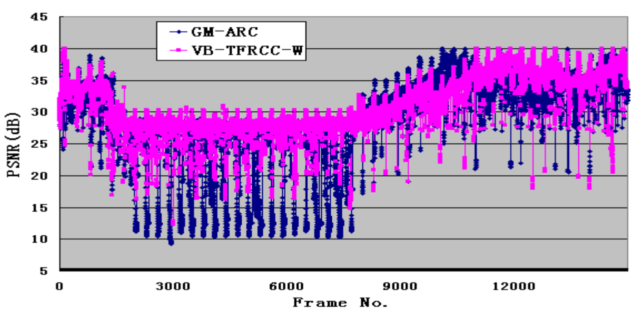


Fig. 5. The PSNR curve measured at the receiver of one GM-ARC flow and one VB-TFRCC-W flow

will decrease the send rate as long as there are packet losses. So it maintains low send rate, and shows poor TCP-friendliness when there exist wireless packet losses (see Table 1). Consequently the low throughput leads to poor video quality (see Table 1). ARC and TFRCC-W, on the other hand, can take into account the effect of wireless losses, and achieve high throughput.

Furthermore, our joint design approach can provide better support for the QoS requirement of the application than GM-ARC. With the joining of 70 FTP flows between 50s and 300s, the available bandwidth becomes so low that source rate control alone can not guarantee the end-to-end delay constraint being met because of the minimum bandwidth requirement and quality smoothness constraint of the application. So the decoder buffer underflow of GM-ARC occurs (see Fig. 4). However VB-TFRCC-W can meet the end-to-end delay constraint by making the send rate temporarily larger than the TCP-friendly value when necessary (see Fig. 6). So VB-TFRCC-W can almost avoid the decoder buffer underflow. As a result, VB-TFRCC-W can significantly decrease the video quality reduction due to lost/late packets between the sender and the receiver, and achieve better playback quality (higher average PSNR) than GM-ARC (see Table 1 and Fig. 5). Note that very low PSNR values (e.g. less than 25dB) in Fig. 5 typically indicate an effective loss of base layer

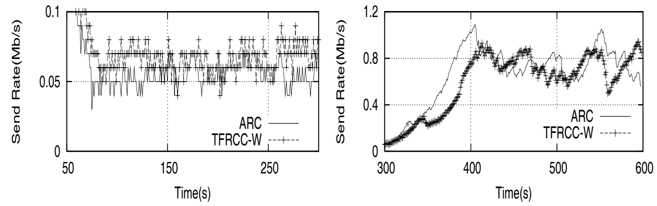


Fig. 6. The send rate of one VB-TFRCC-W flow and one GM-ARC flow

packet for a frame, which introduce significant quality loss for the lost frame and the subsequent frames. To achieve good long-term TCP-friendliness, VB-TFRCC-W maintains a little lower send rate than ARC after 300s to do rate compensation.

To evaluate the performance of the transport protocol, we adopt the same metrics (i.e. long-term TCP-friendliness and internal fairness) defined in Chapter 4 of [5], where a value close to 1 indicates a good TCP-friendliness or internal fairness. From Table 1, we can find that TFRCC-W can guarantee good performance of the transport protocol by using the rate compensation algorithm.

5. CONCLUSION

This paper proposes a joint design algorithm of source rate control and QoS-aware congestion control for wireless video streaming. With a joint decision of the source rate and send rate by taking into account the information from the application, the transport layer and the MAC layer, this cross-layer design approach can avoid suffering from wireless losses, and provide better QoS support for multimedia applications, while maintaining good performance of the transport protocol.

References

[1] P. Zhu, W. Zeng, and C. Li. Joint design of source rate control and qos-aware congestion control for video streaming over the Internet. In *Proc. IEEE Workshop on Multimedia Signal Processing*, Nov. 2005.

[2] J. Widmer, R. Denda, and M. Mauve. A survey on TCP-friendly congestion control. *IEEE NETWORK*, 15:28–37, 2001.

[3] Y. R. Yang and S. S. Lam. General AIMD congestion control. In *Proc. 8th IEEE Int. Conf. Network Protocols*, pages 187–198, Nov. 2000.

[4] D. Sisalem. *Tcp-friendly congestion control for multimedia communication in the Internet*. PhD thesis, Technical university of Berlin, 2000.

[5] J. Padhye. *Towards a comprehensive congestion control framework for continuous media flows in best effort networks*. PhD thesis, University of Massachusetts Amherst, 2000.

[6] S. Cen, P. C. Cosman, and G. M. Voelker. End-to-end differentiation of congestion and wireless losses. *IEEE/ACM Trans. Networking*, 11:703–717, Oct. 2003.

[7] O. B. Akan and I. F. Akyildiz. ARC: The analytical rate control scheme for real-time traffic in wireless networks. *IEEE/ACM Transactions on Networking*, 12:634–644, Aug. 2004.

[8] J. Vieron and C. Guillemot. Real-time constrained TCP-compatible rate control for video over the Internet. *IEEE Trans. Multimedia*, 6(4):634–646, August 2004.

[9] M. van der Schaar and D. Shankar. Cross-layer wireless multimedia transmission: Challenges, principles, and new paradigms. *IEEE Wireless Communications*, pages 50–58, Aug. 2005.