A TCP-Friendly Congestion Control Scheme for Real-time Packet Video Using Prediction

Yeali S. Sun [1], F-M Tsou [2], Meng Chang Chen [3] and Zsehong Tsai [2]

Dept. of Information Management [1]
National Taiwan University
Taipei, Taiwan
sunny@im.ntu.edu.tw

Dept. of Electrical Engineering [2]
National Taiwan University
Taipei, Taiwan
{fmtsou,ztsai}@cc.ee.ntu.edu.tw

Institute of Information Science [3]
Academia Sinica
Taipei, Taiwan
mcc@iis.sinica.edu.tw

Abstract

In order to cope with time-varying conditions in networks with no or limited QoS support like the current Internet, schemes have been proposed for real-time applications to dynamically adjust the traffic source's transmission rate. However, employing adaptive rate control may not be sufficient to prevent or handle network congestion. As most of the real-time applications are based on RTP/UDP protocols, an issue of possibly unfair sharing of bandwidth between TCP and UDP applications has been raised. In this paper, we propose a scheme called R^3CP^+ that integrates several control mechanisms to maximize the delivery performance of real-time continuous media over networks without QoS support. Simulation results show that Recursive Least Square (RLS)-based prediction makes a good use of the past measurement in forecasting the future condition that can effectively avoid and cope with network congestion. It also shows that the scheme achieves reasonably friendly resource sharing with TCP connections.

1. Introduction

As there are many on-going research activities about the guaranteed end-to-end QoS provided in the next generation Internet [1][2][3][4], it will take some time to have such networks and services available. In the meantime, there are a lot of interests in providing multimedia communication services, including real-time audio/video clips embedded in WWW, electronic commerce, IP-telephone and Web TV over the existing Internet. How to maximize the QoS of these streaming applications in such a network becomes an important issue.

In this paper, we consider real-time transport control of sto red multimedia applications, specifically MPEG video. Wo rks on real-time stored video transport in the past have foc used on how to send variable-bit-rate video stream over a c onstant-bit-rate communication channel, e.g., [5][6][7][8]. These works all assume resource reservation and QoS supp ort are provided. Recently, there are studies on stored varia ble-bit-rate video over best-effort networks[9][10][11][12] [13][14]. In our previous work[11], we have shown that int 13][14]. In our previous work[11], we have shown that inte grating rate control with "in-time" packet retransmission ca n significantly improve overall performance. In this paper, we will take a different approach than [12][13][14] in flow/ congestion control.

In order to cope with time-varying network loads, these schemes propose to dynamically adjust the transmission rate of real-time traffic to prevent from sending excessive traffic into the network. In many of these schemes, rate adjustment is based on the feedback information from the receiver(s). The differences between them are mainly in two aspects: *a)* the characterization of the network conditions so that rate adjustment decision can be reliably made; and *b)* the effects of such adjustment on the flow itself as well as the efficiency of network bandwidth usage.

Employing adaptive rate control for real-time continuous media transmission may not be sufficient. Congestion control becomes an important issue in computer networks due to the unfair resources sharing caused by the very different natures of the protocols used by real-time and non-real time traffic. In general, real-time applications are transported using RTP and UDP protocols, whereas most non-real time traffic is based on the TCP protocol. The former implements no congestion control, while the latter does. Research results have shown that this may lead to unfair sharing of bandwidth among the two types of traffic[15]. Therefore, it is essential that the UDP-based real-time transport protocol align with TCP congestion control in the presence of network congestion.

In this paper, we propose a protocol called R^3CP^+ in which several control mechanisms are used and *integrated* to maximize the delivery performance of real-time continuous media over networks that provides no QoS support. They include *a)* the use of a network state predictor based on a Recursive-Least-Square (RLS)-based prediction algorithm to explore the correlation of network conditions in forecasting available bandwidth; and *b)* a sophisticated dynamic rate adjustment scheme with the considerations of congestion avoidance and handling. A

0-7803-5796-5/99/$10.00

main challenge in the design of the rate control scheme for a real-time application is to ensure, during the congestion, there are sufficient packets at the receiver for continuous, no-interrupt playback, while trying to be a good "network citizen" consuming only its fair share of resources at the bottleneck link.

The rest of the paper is organized as follows. In Section 2, we describe an algorithm uses least mean square to forecast network state. In Section 3, the algorithms that computes the *desired sending rate* and *requested sending rate* are presented. In Section 4, the performance of the scheme is evaluated via simulation and the results are analyzed. Finally, Section 5 gives the conclusion.

2. Characterization and Prediction of Network State

In this paper, we use packet loss probability and round trip time to characterize the state of a transmission path. Due to the lack of space, detailed measurement mechanisms of packet loss probability and round trip time can be seen in our previous work[11]. In the end-to-end adaptive flow control, if the decisions are solely based on the observation of the current state, a traffic source may over-react to instantaneous or transient changes of network loads. This can easily cause instability of the system (oscillation of rate adjustment) and network load fluctuation, and possibly generates unnecessary congestion. It would not only lower network utilization, but also have negative impact on the delivery of real-time packets. To remedy the problem, it is important that the receiver catches the trend and notifies the sender to stop rate increasing at proper times to avoid driving the network operating towards congestion.

Here, we will show the feasibility of using *adaptive filter* to forecast network state based on the history as well as current state measurement. Methods such as moving average and exponential average have been widely used. In these schemes, the weighting factors of the current and past information are constant which limits the ability for systems to quickly adapt to network state changes while retaining network stability.

Different from [8], in this paper, we propose the use of an M-step adaptive linear predictor called Kalman filter[17] to forecast the network state. In RLS predictor, the weighting factor is corrected every time a new measurement was taken. Specifically, the algorithm uses the estimation error between the current measurement and its previous prediction to adjust itself. It can not only respond to network dynamics quickly but also remains stable.

2.1 Prediction of Packet Loss Probability

At the end of a rate control period, the receiver will compute the packet loss probability during the period, which is defined as follows:

$$p_n = 1 - \frac{\gamma_n}{\rho_{n-1}}, \quad n = 1,2,... \quad (1)$$

where γ_n is the packet receiving rate measured during the n^{th} control period, and ρ_{n-1} is the requested sending rate sent by the receiver at the beginning of the $(n-1)^{th}$ period.

Assume the measured packet loss probabilities form a random process. Let the size of the memory of the RLS predictor be fixed and denoted by M_{loss}. Let $\underline{P}(n)$ be a vector random variable defined as follows:

$$\underline{P}(n) = [p_n, p_{n-1}, ..., p_{n-M_{loss}+1}]^T \quad (2)$$

Then we use the formula derived in [17] to estimate the packet loss probability in the next rate control period, $\hat{p}_{n+1}$. Detailed operations can be found in [16].

2.2 Prediction of Round-Trip Time

Here, the rate control period is assumed to be one round-trip time. It is taken in order to use the same time scale as that is used in TCP for packet loss detection and congestion control. The estimate of round-trip time is used in three places: *a)* to control the duration of a rate control period; *b)* to compute the desired sending rate; and *c)* to be used by the sender in performing selective transmission. Similarly, the estimation of round-trip time $\{\hat{d}_n\}$ follows the same approach as in the prediction of packet loss probability[16].

3. Congestion-Conscious Adaptive Rate Control

In the Internet, one of the main issues in designing congestion control schemes for real-time packet video is how to minimize the effect of performance degradation when the sending rate is forced to reduce during the congestion. Two methods are proposed here. The first method is to let the sender transmit more data when the network is in an unloaded or lightly-loaded condition. It is different from the conventional methods, e.g., [19] in which rates are continuously increased until congestion occurs. Our idea is to make use of the unused buffer space for additional packet loading for the emergence, i.e. congestion. Second, if the available bandwidth is not sufficient, a *selective transmission* method is employed whereby the sender will give higher transmission priority to more important packets such as I frames, during

network congestion.

The rate control and congestion control in R^3CP^+ are integrated and consists of three steps. First, the receiver makes an attempt to predict the condition that the session is likely to encounter according to its current observation of the network and the knowledge of the past history. In the second step, based on the forecast of the network state, the receiver calculates the *desired sending rate*, denoted by $\hat{\lambda}_n$, based upon the amount of packets in the buffer. Desired sending rate represents the rate that the receiver would like the sender to send so to assure continuous playback of frames. Lastly, depending on different conditions, the receiver determines the *requested sending rate* denoted by ρ_n, differently. The detailed algorithm will be described in Section 3.2.

Once the requested sending rate is determined, the receiver sends the sender a rate control message specifying the requested sending rate, the desired sending rate and the estimate of round-trip time. At the sender, based on the values of these three parameters it decides which frames should be transmitted in the next rate control period. If the requested rate is less than the desired rate, selective transmission is performed. Note that the requested rate includes the transmission of retransmitted packets.

3.1 Calculating Queue Length-based Desired Sending Rate

The goal of computing the desired sending rate is to maximize the playback quality while maintaining a minimum buffer usage. Due to the lack of space, detailed derivation of the algorithm can be found in [16]. To support effective retransmission of lost packets at least once, the *target queue length* at the n^{th} control point q_n^* is set to the minimum value, i.e. one round trip time plus I distance

$$q_n^* = P_{\hat{d}_n} + P_{I\_distance} \quad (3)$$

The desired sending rate is

$$\hat{\lambda}_n = \frac{P_{I\_distance} + 3 \cdot \mu \hat{d}_n - q_n^v - \tilde{\zeta}_n}{(1-\hat{p}_n)\hat{d}_n} - \rho_{n-1} \quad (4)$$

where μ is the average packet playback rate, $\tilde{\zeta}_n$ is the amount of packet skipped and q_n^v is the virtual queue length

3.2 Congestion Control/Avoidance

Note that desired sending rate only represents the receiver's wish to ensure continuous playback of frames However, the actual sending rate requested is determined according to the following algorithm. First, the network is assumed to be in one of the three states: "Unloaded" if $p_n < p_{low}$, "Loaded" if $p_{low} < p_n < p_{high}$ and "Congested" if $p_{high} < p_n$. The algorithm is shown in Table 1. Note that if both the current measurement and the forecast are in the "Unloaded" state, receiver will make more aggressive attempt to increase the rate with a larger amount. The rationale is to store as many data as possible in the receiving buffer (note that it is upper bounded by the maximum buffer space) to reduce the probability of buffer underflow during the congestion. Otherwise, it would take a more cautious step by a smaller amount of increase. In all cases, the receiver always takes the minimum of the current receiving throughput and the previously requested rate plus the increase. If the current condition of the transmission path is in the "Loaded" state, no matter what the forecast says, the receiver will be conservative and take the strategy of retaining the status in quo. If the current measurement indicates the transmission condition of the path is in the "Congested" state, the current observation is taken as a sign of possible network congestion. The receiver will then request the sender to reduce its sending rate. Depending on the different forecasts, different degrees of multiplicative decrease of the currently measured throughput are taken.

Current state/ Forcast	"Unloaded"	"Loaded"	"Congested"
"Unloaded"	$\min(\hat{\lambda}_{max}, \rho_{n-1} + \Delta_{inc})$	$\min(\hat{\lambda}_{max}, \rho_{n-1} + \delta_{inc})$	$\min(\hat{\lambda}_n, \rho_{n-1} + \delta_{inc})$
"Loaded"	$\min(\hat{\lambda}_n, \rho_{n-1})$	$\min(\hat{\lambda}_n, \rho_{n-1})$	$\min(\hat{\lambda}_n, \rho_{n-1})$
"Congested"	$\min(\hat{\lambda}_n \times \delta_{dec}, \rho_{n-1})$	$\min(\hat{\lambda}_n \times \delta_{dec}, \rho_{n-1})$	$\min(\hat{\lambda}_n \times \Delta_{dec}, \rho_{n-1})$

Table 1. Determination of the requested sending rate

4. Performance Evaluation

In this section, we evaluate the proposed prediction-based rate/congestion control scheme via simulation. All simulations were performed by using ns[18]. The network configuration is shown in Figure 1. We consider a single 1.5Mbps congested link shared by a number of TCP connections and UDP/RTP-based video flows. We use an actual video trace (the "Star War" movie[19]) as the video traffic source in the simulation. To provide some context, we compare the performance of R^3CP^+ with three other schemes: transmission without rate/congestion control, R^3CP [11] and the triple-feedback based congestion control (*TFCC*) scheme. In R^3CP, the sender transmits data at the

desired sending rate. It is used to represent the baseline approach that a typical UDP-based real-time connection takes - it grabs as much bandwidth as it could. *TFCC* employs loss-based congestion control and the popular additive increase/multiplicative decrease algorithm in the control of rate adjustment, i.e.

if $p_n < p_{low}$, *req_rate = max(curnt_thruput+INC, max_rate)*
else if $p_{low} < p_n < p_{high}$, *req_rate = curnt_thruput*
else req_rate = max(curnt_thruput/DEC, min_rate)

4.1 Playback Performance

In Figure 2, we show the overall *playback performance* and the performances of individual types of frames of R^3CP^+. The playback performance is defined as the percentage of the number of frames that are successfully played. In these experiments, an in-time completely received frame would not be played if its referenced frame(s) is not present due to the inter-frame dependence. Each video connection has an average rate of 378Kbps. From the figure, we can see R^3CP^+ outperforms the other schemes in all ranges. This is mainly due to the benefit of employing congestion control with prediction mechanism. Notably, the percentage of I frames that are successfully played remains at more than *80%* when the normalized offered load over .75. In R^3CP, no selective transmission is performed; I frames of larger size greatly suffer from packet loss during the congestion. In these experiments, only one packet retransmission attempt was made. The performance can be further improved if multiple attempts are made. For *TFCC*, although selective transmission is performed, its rate control function does not perform as well as that in R^3CP^+. This is because the latter takes more information in the assessment of the network states and thus better adapts sender's sending behavior to the actual network condition. It successfully avoids the possibility of congestion by not increasing the rate too quickly.

4.2 Fairness with TCP Traffic

In this section, we study the bandwidth shared between R^3CP^+ flows and TCP connections. A definition of TCP "friendliness" has been widely adopted in the congestion control of real-time applications[15][12][13][14]. In essence, if a UDP connection shares a bottleneck link with TCP connections of the same link, then the UDP connection should receive the same share of bandwidth (i.e. achieving the same throughput) as a TCP connection. A concern is raised with this approach. That is, real-time applications have more stringent performance requirements to meet than those of non-real time applications (mainly using TCP). We argue that equally sharing the bandwidth over a bottleneck link might sound fair at the first glance. We believe that relative fairness between UDP and TCP traffic is achievable if a congestion control algorithm for real-time applications can be shown to be "reasonably fair" to TCP. Under this philosophy, it would be more practical in terms of meeting the performance objectives of both types of applications and maximizing link utilization. Nevertheless, one still needs to be careful about how much more bandwidth should be given to a real-time connection which is still an open research issue[20]. In the following, we will show that the proposed real-time congestion control scheme only allows a real-time connection to obtain a slightly larger amount of resources than TCP during the congestion.

Let us look at the average throughput a TCP connection can obtain when there are multiple TCPs transmitting over a shared link. In Figure 3(a), one can see that the average throughput continues to decline when more TCP connections join. Now, consider the join of a R^3CP^+ connection. When there is no congestion, the R^3CP^+ connection is very self-controlled by well controlling its rate without aggressively increasing the rate to cause congestion. It takes only what it needs and friendly shares bandwidth with TCP connections. With more TCP connections added to the link, the network becomes congested. The R^3CP^+ connection executes its congestion control and reduces the rate. There is no "TCP bandwidth starvation" situation. For TCP connections, the average throughput reduction is a bit less than that without R^3CP^+ connection. The effect on the throughput reduction for TCP connections is mainly from the competition among themselves with or without R^3CP^+ connection. In Figure 3(b), we repeat the same experiments using *TFCC*. One can see that the connection using *TFCC* obtains more bandwidth than the connection using R^3CP^+. Hence, the average throughput of TCP connections with R^3CP^+ connection is greater than that with *TFCC* connection.

5. Conclusion

In this paper, we have presented a prediction-based rate/congestion control protocol called R^3CP^+ for real-time packet video transfer over the Internet. The simulation results show that R^3CP^+ outperforms in all ranges. It is due to the use of more information in network state assessment in R^3CP^+ and finer control of rate adjustment. Thus, it can not only better adapt the packet sending rate to the actual condition but also avoid unnecessary response to transient changes. It indeed helps to maintain system stability. We also study the issue of resource sharing between R^3CP^+ flows and TCP connections. The results show that R^3CP^+

connection is very self-controlled and can share bandwidth with TCP connections in a "reasonably friendly" fashion. By "reasonably friendly", we mean that the real-time R^3CP^+ connection only takes a slightly larger amount of bandwidth than TCP throughput during the congestion. The source of the throughput reduction for TCP connections is mainly from the competition among themselves with or without R^3CP^+ connection. Therefore, there is no "TCP bandwidth starvation" situation when real-time connections using the flow/congestion control mechanisms of R^3CP^+.

References

[1] R. Braden et al., "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification," RFC 2205, September 1997.

[2] J. Wroclawski, "The Use of RSVP with IETF Integrated Services," RFC 2210, September 1997.

[3] J. Wroclawski, "Specification of the Controlled-Load Network Element Service," RFC 2211, September 1997.

[4] S. Shenker et al., "Specification of Guaranteed Quality of Service," RFC 2212, September 1997.

[5] S. Chong, S-Q Li and J. Ghosh, "Predictive Dynamic Bandwidth Allocation for Efficient transport of real-Time VBR Video over ATM," IEEE JSAC, January 1995.

[6] J. M. McManus and K. W. Ross, "Video-on-Demand Over ATM: Constant-Rate Transmission and Transport," IEEE JSAC, August 1996.

[7] M. Grossglauser, S. Keshav and D. N. C. Tse, "RCBR: A Simple and Efficient Service for Multiple Time-Scale Traffic," IEEE/ACM Trans. on Networking, December 1997.

[8] A. M. Adas, ``Using Adaptive Linear Prediction to Support Real-Time VBR Video Under RCBR Network Service Model," IEEE/ACM Trans. Networking, Oct. 1998.

[9] Z. Chen, S. Tan, R. Campbell and Y. Li, "Real Time Video and Audio in the World Wide Web," 4th International World Wide Web Conference, 1995.

[10] C. Papadopoulos and G. M. Parulkar, "Retransmission-Based Error Control for Continuous Media Applications," Proceedings of NOSSDAV'96, 1996.

[11] Y. Sun, F. M. Tsou and M. C. Chen, "A Buffer-Occupancy based Adaptive Flow Control Scheme with Packet Retransmission for Stored Video Transport over Internet," IEICE Trans. on Communications, Nov. 1998.

[12] D. Sisalem and H. Schulzrinne, "The Loss-Delay Based Adjustment Algorithm: A TCP-Friendly Adaptation Scheme," Proceedings of NOSSDAV '98, July 1998.

[13] R. Rejaie, M. Handley and D. Estrin , "RAP: An End-to-end Rate-based Congestion Control Mechanism for Realtime Streams in the Internet," INFOCOM '99, 1999.

[14] J. Padhye, J. Kurose, D. Towsley and R. Koodli, "A TCP-Friendly Rate Adjustment Protocol for Continuos Media Flows over Best Effort Networks," ACM Sigmetrics'99, May 1999.

[15] S. Floyd and K. Fall, "Router Mechanisms to Support End-To-End Congestion Control," Technical Report, February 1997.

[16] Y. Sun, F. M. Tsou and M. C. Chen, "A TCP-Friendly Congestion Control Scheme for Real-time Packet Video Using Prediction," Technical Report, June 1999.

[17] S. Haykin, Adaptive Filter Theory, Prentice Hall, 1986.

[18] UCB/LBNL/VINT Network Simulator-ns (version 2), http://www-mash.cs.berkeley.edu/ns/.

[19] M. W. Garrett and A. Fernandez, "MPEG-1 Video Trace," Bellcore, ftp://thumper.bellcore.com/pub/vbr.video.trace/MPEG.data, 1992.

[20] F. Kelly, "Charging and Rate Control for Elastic Traffic," European Trans. on Telecommunications, volume 8, 1997.

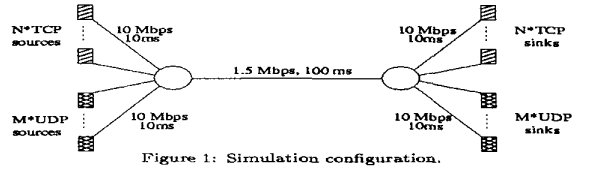
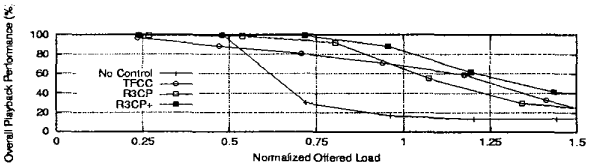
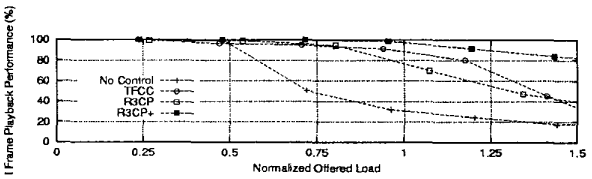


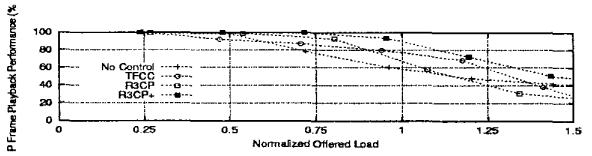
Figure 1: Simulation configuration.



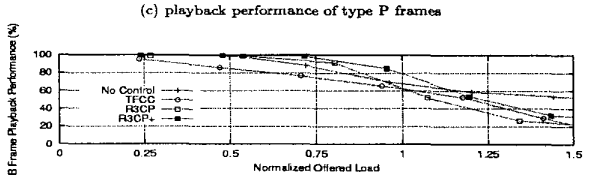
(a) overall playback performance



(b) playback performance of type I frames

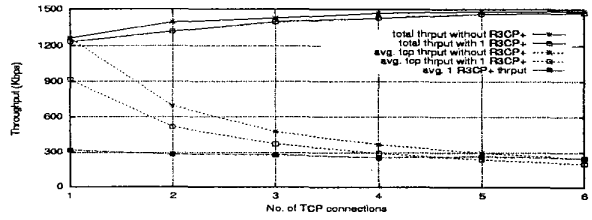


(c) playback performance of type P frames

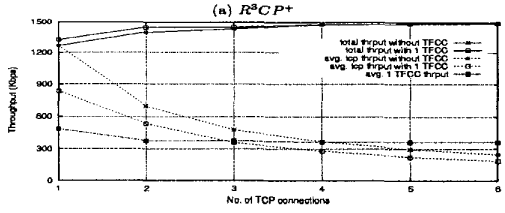


(d) playback performance of type B frames

Figure 2: Frame playback performance



(a) R^3CP^+



(b) $TFCC$

Figure 3 Bandwidth sharing between real-time video flow with TCP connections.