On TCP-friendly Video Transfer

Naoki Wakamiya, Masayuki Murata, Hideo Miyahara
Graduate School of Engineering Science, Osaka University
1-3 Machikaneyama, Toyonaka, Osaka 560-8531, JAPAN
wakamiya@ics.es.osaka-u.ac.jp

***Abstract*—When both TCP and UDP connections co-exist in the Internet environment, the performance of TCP connections is heavily affected by the behavior of "greedy" UDP connections of real-time multimedia applications. In this paper, we propose a new TCP-friendly rate control protocol for UDP connections to fairly share the link with TCP connections. To achieve fairness among TCP and UDP connections while performing high quality video transmission, we argue that (1) the interval of rate control must be appropriately determined, (2) the network condition must be accurately predicted, (3) the TCP throughput must be precisely estimated and (4) the video rate must be effectively adjusted. However, the existing proposals on the TCP-friendly rate control do not satisfy all of those conditions. Through simulation experiments using the MPEG-2 video application, we show that high quality video transfer can be performed with our proposed method and the network bandwidth is fairly shared among TCP and UDP connections.

I. Introduction

In the current Internet, each application chooses the preferable transport protocol to achieve the required QoS (Quality of Service) because the network does not provide QoS guarantee mechanisms. For example, traditional data applications such as `http`, `ftp` and `telnet` employ TCP which accomplishes the reliable data transfer by means of window-based flow control and retransmission mechanisms. On the other hand, to avoid the unacceptable delay introduced by the retransmission, real-time multimedia applications such as video conferencing prefer UDP.

The multimedia application emits a considerable amount of data especially when video streams are employed to achieve the impressive presentation. The network is easily driven to congestion by such greedy UDP packets without any appropriate control mechanism. When the congestion occurs, TCP connections shrink their congestion window `cwnd` to recover from the congestion. On the contrary, UDP connections are persistent with sending much data for the high quality video presentation. This means that the performance of TCP connections is heavily affected by UDP [1], [2].

One way to achieve fairness among TCP and UDP connections is for intermediate routers to employ the appropriate packet scheduling mechanisms such as CBQ (Class-Based Queueing) or WFQ (Weighted Fair Queueing) [2]–[4]. However, it is required that each intermediate router distinguishes the ill-behaved flows and gives them incentive to regulate the data emission rate [1]. Furthermore, all routers should employ the same scheduling mechanism with same control parameters to achieve the end-to-end fairness. An alternative and easier way is to apply the rate control algorithm to UDP connections. In recent years, the number of researches have been devoted to achieving the fairness among TCP and UDP connections by introducing the concept of *TCP-friendly* [5]–[11]. In those works, "TCP-friendly" is defined as "a non-TCP connection should receive the same share of bandwidth as a TCP connection if they traverse the same path". The TCP-friendly rate control protocol regulates its data emission rate according to the network condition to achieve the same throughput that a TCP connection would acquire on the same path. In that sense, their control mechanisms provide the TCP-friendliness, but the fairness among TCP and UDP connections can be achieved by using carefully chosen parameters. More importantly, they do not take into account characteristics of video applications. For example, because it is preferable to provide users with meaningful and effective video presentation, the rate control interval should be long enough to avoid frequent video quality fluctuation. However, infrequent control leads to the unfairness because the video applications cannot catch up with changes of network condition.

In this paper, we propose a new TCP-friendly rate control mechanism suitable to the video applications where the video emission rate can be adjusted to the appropriate level according to the estimated network condition. To achieve fairness among TCP and UDP connections while performing high quality video transmission, we will argue that (1) the interval of rate control must be appropriately determined, (2) the network condition must be accurately predicted, (3) the TCP throughput must be precisely estimated and (4) the video rate must be effectively adjusted. Through simulation experiments using the MPEG-2 video application, we show that high quality video transfer can be performed with our proposed method and the link bandwidth is fairly shared among TCP and UDP connections.

This paper is organized as follows. In Section 2, we propose TFRCP (Tcp-Friendly Rate Control Protocol) suitable for video applications. In Section 3, we then evaluate the performance and effectiveness our proposed TFRCP mechanism through simulation experiments. Further, in Section 4, we investigate the way to improve the TFRCP algorithm. Finally, we summarize our paper and outline our future work in Section 5.

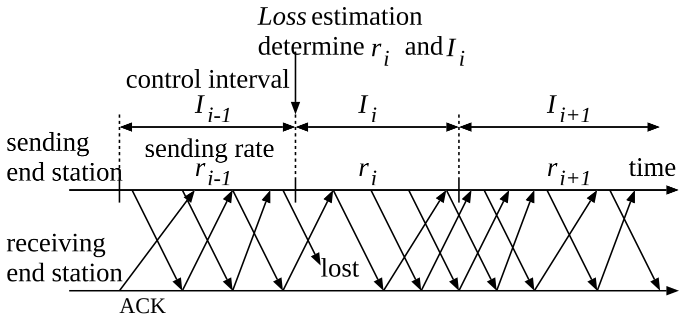

Fig. 1. TCP-friendly rate control protocol

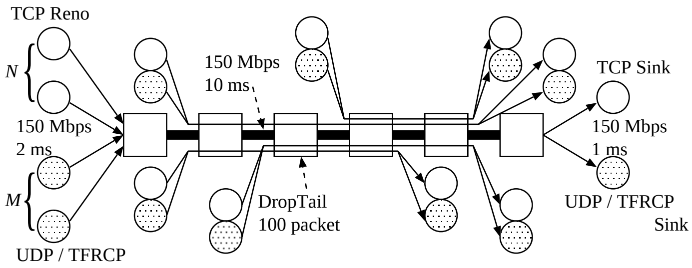

Fig. 2. Network model

II. TCP-FRIENDLY RATE CONTROL PROTOCOL

In this section, we consider new TFRCP (Tcp-Friendly Rate Control Protocol) suitable for real-time video applications. TFRCPs proposed in [5]–[11] behave as illustrated in Fig. 1; at the end of the control interval $i-1$ (with duration I_{i-1}), the sender estimates the network condition from information gathered within the interval. Then it estimates the throughput of the TCP connection which traverses the same path and finally adjusts its emission rate r_i for the next interval i to the estimated TCP throughput r_{TCP}.

A. TCP throughput estimation

In [5], the authors proposed a formula to derive the TCP throughput r_{TCP} from RTT, MTU and packet loss probability p;

$$r_{TCP} = C \times \frac{MTU}{RTT\sqrt{p}} \tag{1}$$

0.87 is given for C when the client employs the delayed ACK mechanism, and 1.22 otherwise. This formula is attractive for its simplicity, but they assume that packet loss occurs randomly and timeout does not happen. Another formula proposed in [8] takes into account the timeout and the advertised window. TCP throughput r_{TCP} is derived from the advertised window size W_{max}, MTU, RTT, RTO T_0 and packet loss probability p;

$$r_{TCP} \approx \min(MTU\frac{W_{max}}{RTT}, \frac{MTU}{RTT\sqrt{\frac{2bp}{3}}+T_0\min(1,3\sqrt{\frac{3bp}{8}})p(1+32p^2)}) \tag{2}$$

where b is 2 for delayed ACK and otherwise 1.

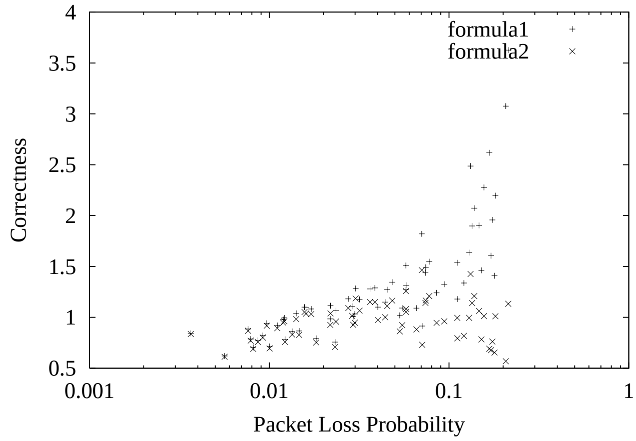

Fig. 3. Comparison between two estimators

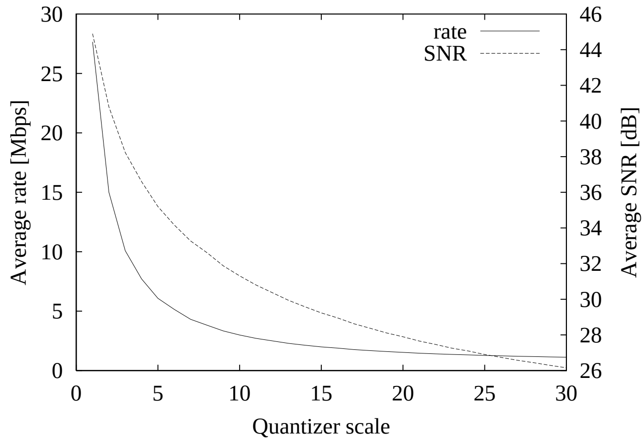

Fig. 4. Relationship among quantizer scale, video rate, video quality

We evaluate the applicabilities of those two formulas in the network model shown in Fig. 2 where TCP Reno and UDP connections coexist. Simulation experiments shown in this paper are performed with `ns version 2` [12]. The simulation results are depicted in Fig. 3 where "Correctness" is defined as the ratio of the estimated throughput to the actual throughput. From Fig. 3, it is obvious that Eq. (2) is a better estimator than Eq. (1). In this paper, we employ Eq. (2) for the TCP throughput estimation in TFRCP. However, the formula is not applicable when no packet is lost within a preceding control interval. Thus, according to [9], we determine the video emission rate r_i in the next interval i as;

$$r_i = \begin{cases} r_{TCP} = \dfrac{MTU}{RTT\sqrt{\frac{2p}{3}}+T_0\min(1,3\sqrt{\frac{3p}{8}})p(1+32p^2)}, & \text{if } p>0 \quad (3) \\ 2\times r_{i-1}, & \text{if } p=0 \quad (4) \end{cases}$$

In our experiments, b in Eq. (2) is set to 1 because we do not employ the delayed ACK mechanism and the advertised window W_{max} is large enough not to affect the video sender's behavior.

B. Estimation of network condition

In TFRCP, network condition is expressed by RTT and packet loss probability. To estimate the TCP throughput by Eq. (3), those values must be accurately determined from information gathered within each control interval. RTT can be obtained by observing timestamps added to packet headers [5], [7]–[9], [11] or exchanging RTCP (Real-Time Control Protocol) packets [10]. Packet loss probability can be derived using sequence numbers [10], [11].

In this paper, we estimate RTT by timestamp-based approach and packet loss probability by observing sequence numbers of ACKed packets. In our TFRCP, all packets contains the timestamp and sequence number in their headers, and the receiver sends back ACK for every received packet. RTT and RTO are smoothed by Jacobson's algorithm [13] and packet loss probability p is given as the ratio of the number of lost packets to the number of emitted packets within each control interval.

C. Video rate control

Once the target rate r_i is successfully determined from Eqs. (3) and (4), the video sender should adjust its video emission rate to the estimated throughput by regulating the video coding rate. In this paper, we consider MPEG-2 [14] as the video coding algorithm. In MPEG-2, each captured picture is first discrete-cosine-transformed and each DCT coefficient is quantized with specified quantizer scale. Thus, the coded video quality and the amount of data can be regulated by choosing the appropriate quantizer scale.

In Fig. 4, we depict an example of the relationship among the quantizer scale, the average video coding rate and the video quality in terms of SNR (Signal to Noise Ratio). The video sequence was coded by the MPEG-2 VBR coding algorithm where a static quantizer scale is applied to all pictures in the sequence [15]. The video sequence consists of 150 pictures of 704×480 pixels and the frame rate is 29.97 fps. By applying the method proposed in [15], we can easily determine the appropriate quantizer scale to adjust the video rate to the estimated TCP throughput.

In our TFRCP mechanism, the sender is allowed to emit the video data as far as the sum of data injected into the network is smaller than the multiple of the target rate r_i and the length of control interval I_i. When the network is heavily congested, considerable numbers of packets are lost and the target rate determined from Eq. (3) becomes far below the minimum video rate, which corresponds to the largest quantizer scale. In such cases, the video application first applies the largest quantizer scale to make the data emission rate as small as possible and sends the video data as much as possible.

D. Control interval appropriate for video application

As described in Section I., the duration of each control interval I_i (Fig. 1) must be carefully determined to accomplish the effective control. It is expected that the fair share of network bandwidth can be achieved by frequent rate control. Ideally, it should be in the same order as TCP. However, the amount of feedback information gathered by the sender becomes insufficient to accurately estimate TCP throughput within a short interval. More importantly, the frequent video rate control results in the unacceptable video quality fluctuation because the rate adjustment is realized by a means of video quality regulation as explained in Subsection C.. On the other hand, if the control interval is too long, the TFRCP rate control does not reflect the dynamics of network condition and the fair-share of bandwidth cannot be expected. The structure of MPEG-2 video stream consists of GoP (Group of Pictures), i.e., the repeated sequence of three types of coded pictures. In this paper, we employ the duration of 32 times as long as a smoothed RTT [7]. The control interval is then rounded to the GoP time unit by $\lceil \frac{32 \times RTT}{\text{GoPtime}} \rceil \times$ GoPtime. Here, a GoP time unit is given as the multiple of the number of pictures in a GoP and the frame interval, that is 30/29.97=1.001 sec in this paper.

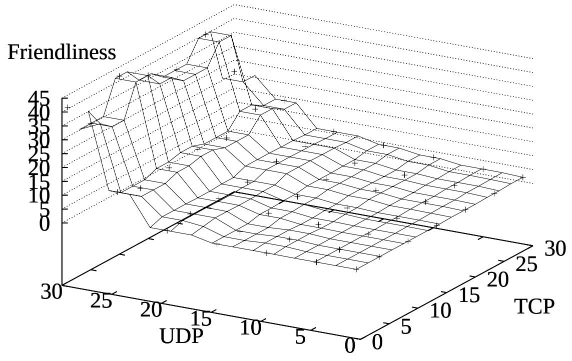

Fig. 5. Friendliness (UDP)

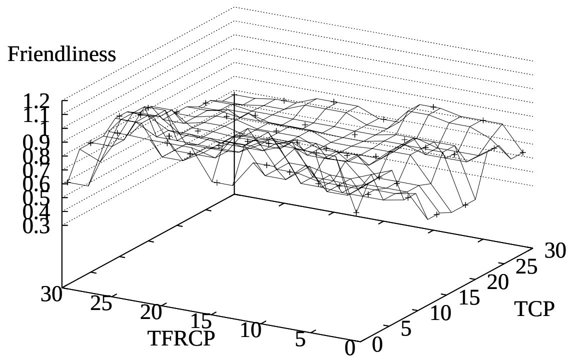

Fig. 6. Friendliness (TFRCP)

III. Simulation results

Figures 5 and 6 compare the friendliness variation against combinations of variety numbers of TCP and video connections in the network (Fig. 2). In obtaining Fig. 5, a simple UDP without any rate control is assumed for the video connections. Friendliness is defined as the ratio of the averaged throughput of non-TCP connections to that of TCP connections. The number N of TCP connections are persistent, that is, they always

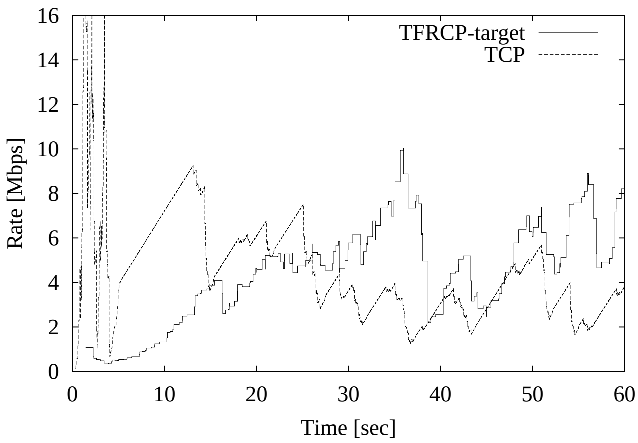

Fig. 7. Averaged rate variation (TFRCP)

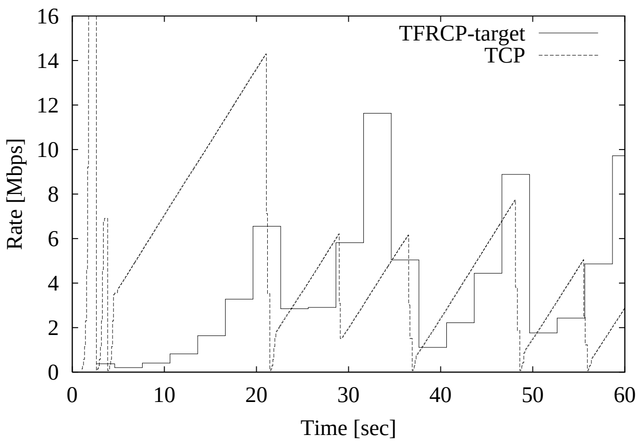

Fig. 8. Rate variation (TFRCP)

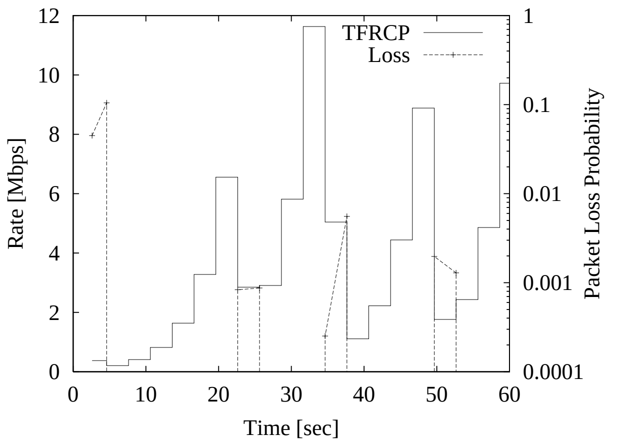

Fig. 9. Packet loss probability variation (TFRCP)

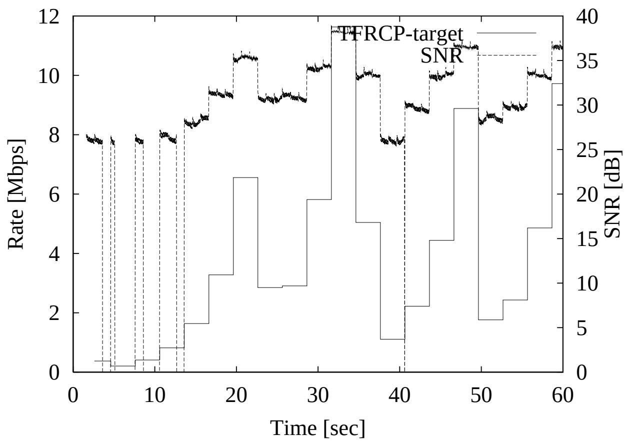

Fig. 10. Picture quality variation (TFRCP)

have data to send. The clock granularity of TCP Reno was set at 10 msec. Packet size is 1000 Bytes and identical among all connections. All connections start data emission at randomly selected time from 0 to 3.00 sec. The initial rate of TFRCP is 1.12 Mbps which corresponds to the largest quantizer scale, 30 (Fig. 4).

When the video application employs UDP, the network bandwidth is easily occupied by the greedy UDP traffic. As a result, the performance of TCP deteriorates and the average throughput of UDP connections becomes about 41 times larger than that of TCP connections as shown in Fig. 5. On the other hand, when our TFRCP algorithm is employed by the video application, the TCP-friendly video transmission is achieved where the "friendliness" ranges from 0.374 to 1.125 (see Fig. 6).

To see the transient behavior of TFRCP precisely, we show the variation of throughput in Fig. 7 where ten TCP connections and ten TFRCP connections co-exist. The figure is magnified to have a closer look. In the case of TFRCP, the average of target rate r_i is also depicted. For TCP, the throughput of each TCP connection is derived by dividing the instantaneous congestion window size by RTT. In Fig. 7, it is shown that the network bandwidth is successfully shared among TCP and TFRCP connections except during the initial startup phase up to around 15 sec. We next pick up a pair of TCP and UDP connections and depict their transient behaviors in Figs. 8 through 10 for the throughput, the packet loss probability and the video quality, respectively. The TFRCP connection continues to double its data emission rate until it experiences the packet loss (see Fig. 9). Once it detects the packet loss in the preceding interval, the emission rate is decreased to the value estimated by Eq.(3). As a result of rate control, the transient behavior of TFRCP connection becomes similar to that of TCP as shown in Fig. 8.

However, rate variation shown in Fig. 8 and the resultant packet loss probability in Fig. 9 suggest that the aggressive rate increase of TFRCP by Eq.(4) sometimes causes the instantaneous and serious congestion. The sender is forced to decrease its own data emission rate when it detects the packet loss, and the decreased rate derived from Eq.(4) is usually much smaller than the data emission rate of the preceding interval. Then, the video presentation of stable quality can never be expected with such drastic rate fluctuation as shown in Fig. 10 where the video quality variation is depicted in terms of SNR (Signal to Noise Ratio). The SNR varies between 25.360 to 39.020 even after the initial startup phase, and such a quality fluctuation must be perceived by users at the receiver. In particular, no picture data is transmitted at 40.611 sec because the decreased data emission rate is so small that the sender is obliged to stop sending picture data according to the mechanism stated in Subsection C.. We should also note that the SNR values in Fig. 10 are obtained at the sender side, which means that the picture quality perceived at the receiver must be smaller than that in Fig. 10 because some packets could be lost in the network.

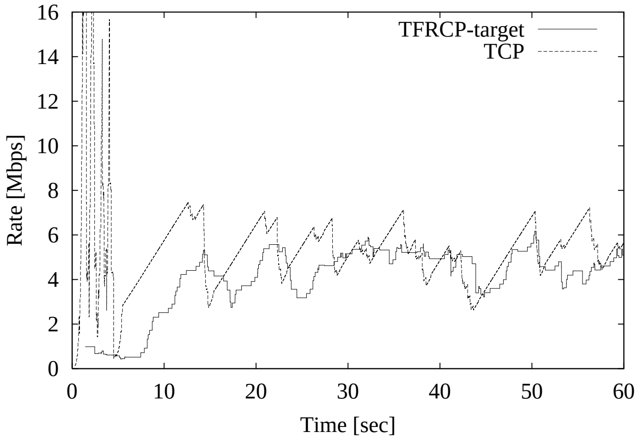

Fig. 11. Averaged rate variation (TFRCP-AI)

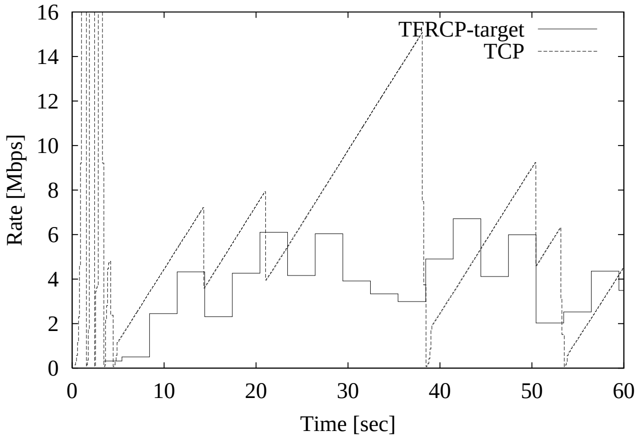

Fig. 12. Rate variation (TFRCP-AI)

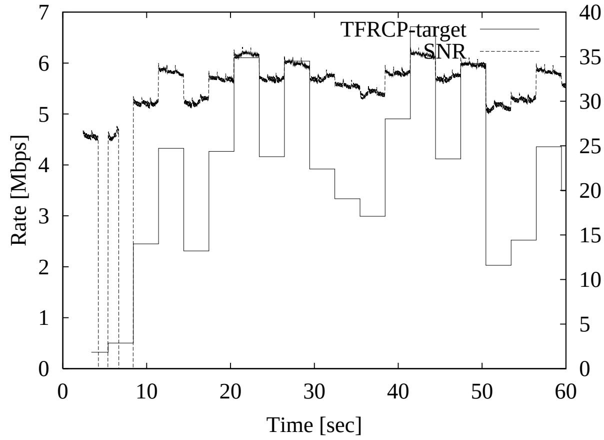

Fig. 13. Picture quality variation (TFRCP-AI)

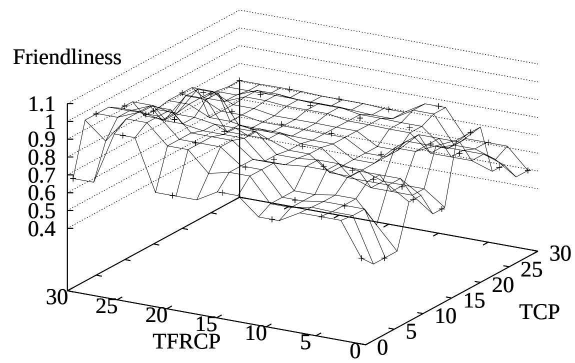

Fig. 14. Friendliness (TFRCP-AI)

IV. IMPROVING TFRCP PERFORMANCE

In this section, we describe several methods to improve the TFRCP mechanism for achieving the higher friendliness and the stable video quality.

A. Consideration on TFRCP rate increase

In our TFRCP mechanism proposed in Section II., the data emission rate is doubled when there is no packet loss in the preceding control interval. Such sudden and aggressive rate increase leads to the instantaneous congestion as having been shown in Section III.. In this section, we introduce another way of increasing the TFRCP rate which is not aggressive, but still more similar to that of TCP.

During the congestion avoidance phase, TCP increases the congestion window size by 1/`cwnd` every time it receives ACK. Thus, the congestion window is increased by one segment size each RTT [13]. Our modified TFRCP imitates this mechanism by translating the window-based flow control to the rate-based one. The new algorithm becomes as follows:

$$r_i = \begin{cases} r_{TCP} = \dfrac{MTU}{RTT\sqrt{\frac{2p}{3}} + T_0 \min(1, 3\sqrt{\frac{3p}{8}})p(1+32p^2)}, & \text{if } p > 0 \quad (5) \\ r_i + \dfrac{MTU \times I_i}{RTT^2}, & \text{if } p = 0 \quad (6) \end{cases}$$

By introducing Eq. (6), the additive rate increase as in TCP can be performed in TFRCP.

We evaluate TFRCP with an additive increase mechanism (TFRCP-AI) under the same simulation environment as in Section III.. The simulation results are depicted in Figs. 11 through 14. Compared with TFRCP proposed in Section II., the rate variation becomes smoother as shown in Figs. 11 and 12. As a result, the video quality becomes stable and ranges from 28.570 to 36.080 (see Fig. 13). The averaged SNR of TFRCP-AI experiments is 30.537 dB, which is slightly higher than that of TFRCP, 29.535 dB. In addition, the variation of SNR is smaller than that of TFRCP.

It should be noted that the bandwidth occupied by TFRCP is close to that of TCP (see Fig. 11). It follows that the high TCP-friendliness is achieved and the network bandwidth is fairly shared among TCP and TFRCP connections. The friendliness

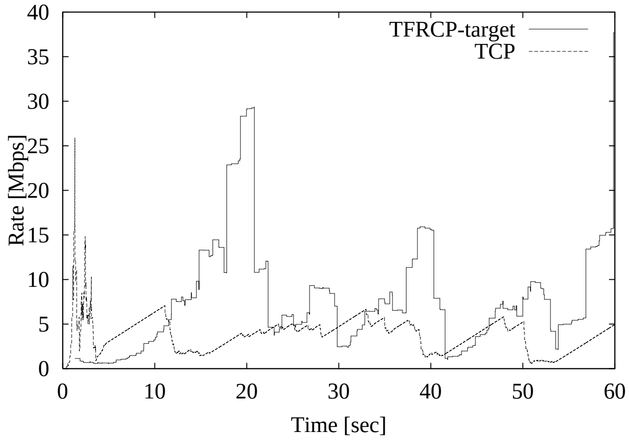

Fig. 15. Averaged rate variation (TFRCP-SP)

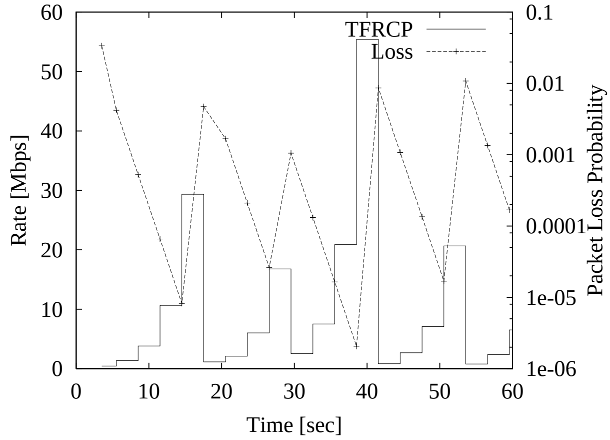

Fig. 17. Packet loss probability variation (TFRCP-SP)

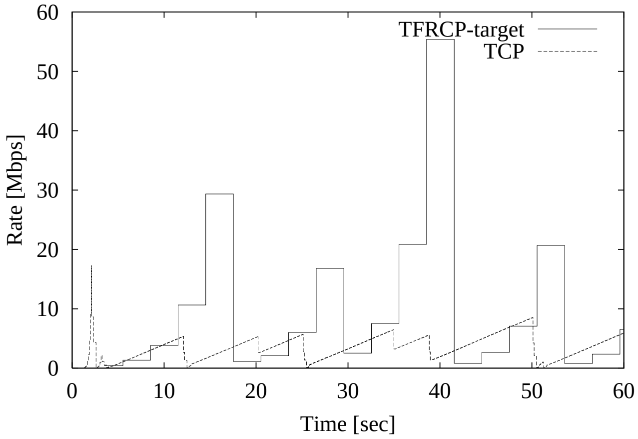

Fig. 16. Rate variation (TFRCP-SP)

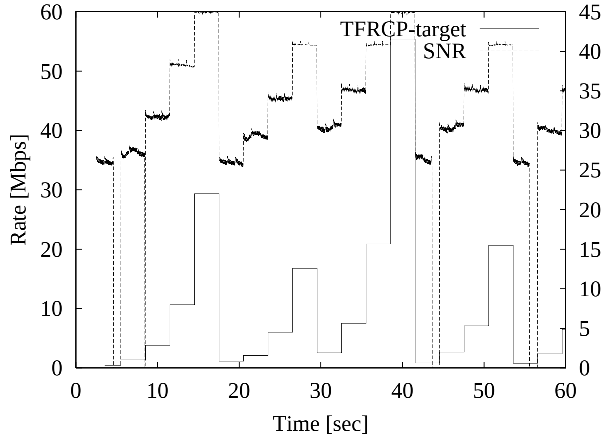

Fig. 18. Picture quality variation (TFRCP-SP)

variation in TFRCP-AI is depicted in Fig. 14, which ranges from 0.439 to 1.002.

B. Consideration on estimation of packet loss probability

Next, we consider how the packet loss probability p should be estimated. In Section II., the packet loss probability is obtained by dividing the number of lost packets by the number of emitted packet within a control interval. This approach seems attractive for its simplicity but could be inappropriate because the estimated TCP throughput is easily affected by the short-term congestion. It is one of the reasons of rate fluctuation that we want to eliminate. Considering the fact that Eq. (2) is used to derive the "averaged" throughput of the TCP connection under the stable network condition, the packet loss probability used in Eq. (3) should be calculated over the long interval, or smoothed by applying the low pass filter function such as exponentially weighted moving averaging method against instantaneous values.

We apply the latter approach to the rate increase mechanism. We employ the low-pass filter which is similar to that for RTT estimation in TCP. That is, the smoothed packet loss probability p_i^s is determined as,

$$p_i^s = \frac{7p_{i-1}^s}{8} + \frac{p_{i-1}}{8} \tag{7}$$

It is substituted to Eq. (3) to derive the target rate r_i at the end of interval $i-1$. Here, p_{i-1} corresponds to the observed packet loss probability in the interval $i-1$. The simulation results of TFRCP with smoothed packet loss probability (TFRCP-SP) are shown in Figs. 15 through 18. As shown in Fig. 15, the average throughput of TFRCP connections becomes higher than that of TCP in most of cases because the TFRCP connection aggressively increases its data emission rate according to the estimated TCP throughput (Fig. 16). The reason for this greedy behavior can be observed in Fig. 17. With the algorithm in Eq. (7), the smoothed packet loss probability p_i^s for interval i becomes ten times smaller than p_{i-1}^s when there is no packet loss in the interval $i-1$. By applying the smoothed probability p_i^s to Eq. (3), the data emission rate of the interval i becomes $\sqrt{10}$ times larger than that of the preceding interval $i-1$. As a result, the rate increase of TFRCP-SP becomes mush higher than that of TFRCP-AI. Such a sudden increase causes the periodic and influential packet loss, and the video quality fluctuates very much as shown in Fig. 18. Thus, we have to employ the filter which provides more smoother changes than Eq. (7) in the smoothed packet loss probability. However, to achieve the effective control with such a filter, several numbers of parameters should be carefully chosen dependent on the network condition and the traffic characteristics.

From above observations, we conclude that the estimation of loss probability in our proposed TFRCP described in Subsec-

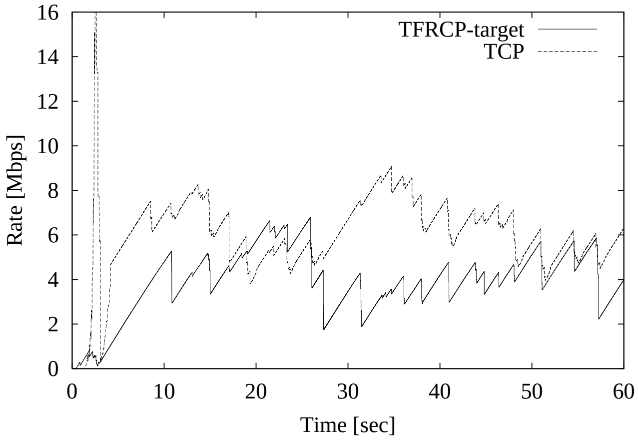

Fig. 19. Averaged rate variation (RTT)

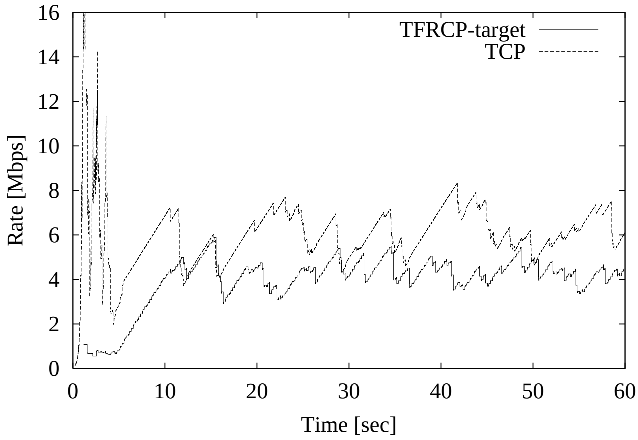

Fig. 21. Averaged rate variation (GoP)

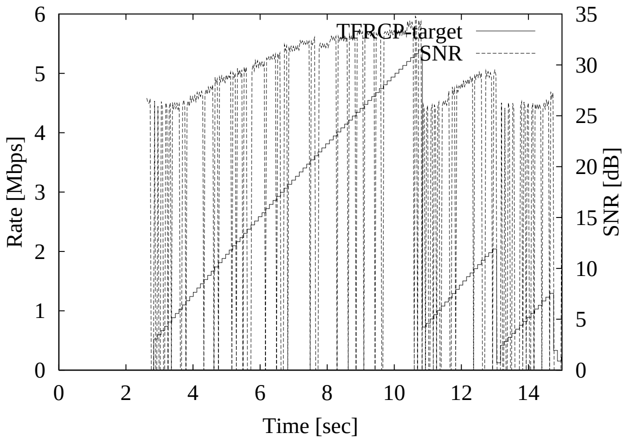

Fig. 20. Picture quality variation (RTT)

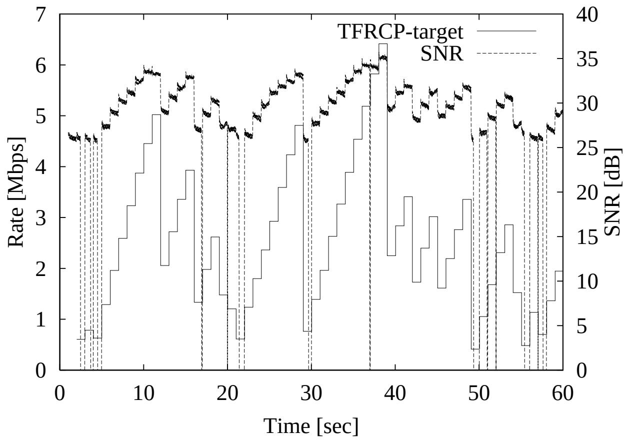

Fig. 22. Picture quality variation (GoP)

tion B. provides reasonable performance and is attractive for its simpleness.

C. *Consideration on appropriate control interval*

So far, the TFRCP algorithms proposed in this paper estimate the TCP throughput from the observed network condition and regulate the data emission rate at the periodic interval. In the rest of this section, we call this strategy as 32–GoP. As having been shown in Subsection A., TFRCP with 32–GoP interval and an additive increase mechanism provides good performance. However, one may expect the higher friendliness and an effective usage of network bandwidth by introducing the shorter interval. Or, the network condition could be kept stable with the longer interval while expecting the less fluctuation of the video quality.

In Figs. 19 through 26, we depict variations of the averaged rate and the picture quality for several settings of control interval, i.e., smoothed RTT, single GoP-time, 16$\times$RTT rounded by GoP-time and 64$\times$RTT rounded by GoP-time, respectively. The friendliness of those trials ranges from 0.587 to 0.749. From Figs. 19, 21, 23 and 25, we observe that the smaller control interval enables TFRCP to regulate its data emission rate similarly with TCP. However, the frequent control leads to the lower throughput of TFRCP because it causes frequent and bursty packet loss.

When the rate control is performed every RTT as in TCP window-based flow control, TFRCP behaves as pseudo-TCP and accomplishes the TCP-friendly video data transfer in an almost satisfactory way. However, Fig. 20 shows that the video quality seriously deteriorates with RTT-based rate control. The smoothed RTT in our simulation experiment is about 103 msec and is not a multiple of a picture time (1000/29.97=33.37 msec), which is still smaller than a GoP-time (30/29.97=1.001 sec). Thus, the sender of the TFRCP connection is forced to regulate its data emission rate by adjusting the quantizer scale while the picture is being coded. It is almost impossible to inter-picture rate adjustment in the MPEG-2 algorithm. It is impossible to predict the resultant picture size in such adjustment because the MPEG-2 employs the DCT transform, quantization and variable length coding. Then, the coded picture size often exceeds the allowable amount of bits and the sender abandons data emission in such a case. As a result, the picture quality fluctuates in RTT-based control as shown in Fig. 20.

The GoP-based rate control seems preferable for the MPEG-2 coding algorithm. We employed the MPEG-2 VBR coding algorithm where the quantizer scale is fixed during a control interval. However, the MPEG-2 CBR coding algorithm such as "Test Model 5" [16] might be useful if the control interval is a multiple of GoP-time and the rate control is performed on a GoP basis. The resultant picture quality variation is depicted in

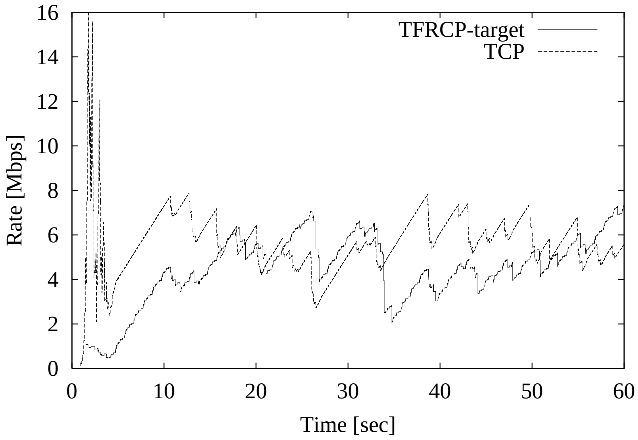

Fig. 23. Averaged rate variation (16–GoP)

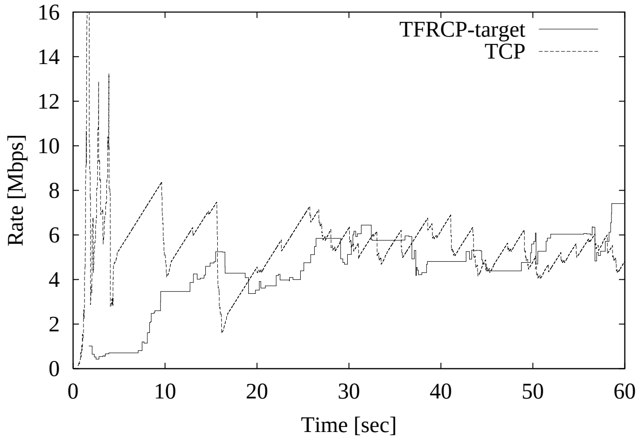

Fig. 25. Averaged rate variation (64–GoP)

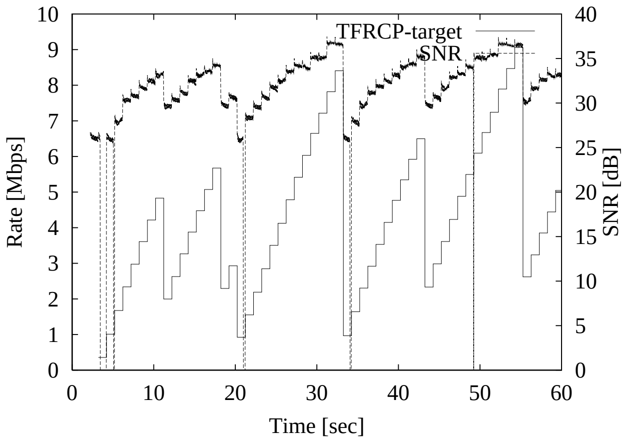

Fig. 24. Picture quality variation (16–GoP)

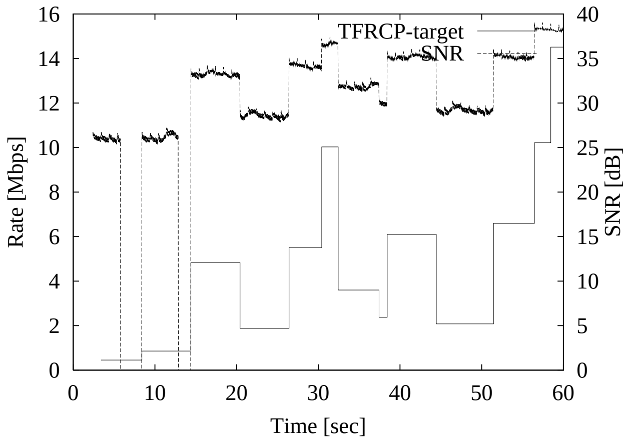

Fig. 26. Picture quality variation (64–GoP)

Fig. 22. Compared with Fig. 20 of RTT-based TFRCP, the picture quality is much improved but the periodic quality degradation is perceptible as shown in Fig. 22, while the average SNR is almost same as that of 32–GoP TFRCP in Fig. 13.

In Figs. 23 and 24, we depict the simulation results of TFRCP with 16–GoP interval. The 16–GoP interval is identical to the GoP-time in this case because RTT in our simulation environment is around 106 msec and the GoP-time is about 1001 msec, i.e., the control interval is $\lceil \frac{16\times106}{1001} \rceil \times 1001$=1001 msec. However, we believe that the control interval should be determined based on the estimated RTT because the TCP's behavior is dominated by RTT.

Finally, we examine a longer control interval to achieve the stable control. As mentioned in Subsection B., estimation of TCP throughput by Eq. (3) is expected to be accurate when we observe the network condition for long duration. Figs. 25 and 26 correspond to the simulation results of the 64–GoP TFRCP. As expected, the rate variation of TFRCP becomes stable as shown in Fig. 25. It means that the aggregated TFRCP connections can be regarded as the CBR connection from TCP connections. Thus, the half of the network bandwidth is occupied by TFRCP and TCP connections compete for the rest in this case. It follows that the fair share of bandwidth can be accomplished by TFRCP without the bandwidth reservation mechanism. The video quality changes in the longer cycle as shown in Fig. 26 and it is preferable to the video application.

However, one may claim that 64–GoP TFRCP cannot catch up the sudden and drastic changes of the network condition. One possible and easy solution against this problem is to employ the shorter control interval, such as 32–GoP. Another is to adjust the control interval when the network condition changes. It is obviously difficult for an end system to identify the number and the characteristics of flows in the network. However, by considering the fact that those changes should involve the variation of RTT and packet loss probability, our TFRCP algorithm is expected to accomplish the up-to-date control to a large extent. The reason is that the control interval takes into account the smoothed RTT and the data emission rate is determined from the packet loss probability. To achieve higher performance, we are now considering on the regulation of control interval based on the packet loss probability from the standpoint that the high packet loss probability indicates the network congestion, which is caused by the newly started connection or the bursty data emission of ill-behaved flow.

V. Conclusion

In this paper, we proposed the TCP-friendly rate control algorithm with which the fair-share of network bandwidth among TCP and video connections is accomplished. We give considerations on some aspects of the TFRCP algorithm and investigate possible variants. Through simulation experiments, we show that the 32–GoP TFRCP with additive increase mechanism provides the best performance.

One interesting observation in our simulation results is that the network condition becomes relatively steady after the initial startup phase of about 15 sec. This implies that the TFRCP sender can perform the stable video transfer if it initially sends dummy data for several seconds and then emits the actual video data after the rate variations becomes small. The high quality video presentation can also be achieved by introducing the playout buffer at the receiver side and store a sufficient amount of data during the initial startup phase. After the initial phase, the receiver can expect the steady data arrival as far as the network condition does not drastically change.

In this paper, we only employ the single network model (Fig.2) where RTT is about 106 msec and there is no dynamic changes in the number of active flows. Although we have not examine the applicability of our TFRCP to the changes of network condition, we expect that the network becomes stable after some period.

We are currently working on the implementation issues of our TFRCP in the actual network. In our implementation, TFRCP is on the RTP protocol stack and effectively utilizes the mechanism of RTCP to obtain the feedback information required for the estimation of network condition. In RTP-based approach, the feedback information is obtained by exchanging RTCP packets. In such a case, we should employ some error recovery mechanism to avoid the inaccurate estimation from the insufficient information which would be caused by the loss of RTCP packets. One possible way is to exchange several numbers of RTCP packets in each interval. Another is for the RTCP packet to have the redundant information, where the RTCP packet contains the information the preceding RTCP packet had. We should also consider the variation of RTT. In the actual network, observed RTT considerably changes. Especially when the network is congested, RTT becomes large and it leads to the longer control interval. As shown in Subsection D., the infrequent control cannot catch up the changes of network condition.

REFERENCES

[1] S. Floyd and K. Fall, "Promoting the use of end-to-end congestion control in the Internet," *IEEE/ACM Transactions on Networking*, May 1999.

[2] S. Floyd and V. Jacobson, "Link-sharing and resource managament models for packet networks," *IEEE/ACM Transactions on Networking*, vol. 3, pp. 365–386, August 1995.

[3] J. C. R. Bennett and H. Zhang, "Hierachical packet fair queueing algorithms," *Proceedings of ACM SIGCOMM'96*, vol. 26, pp. 143–156, October 1996.

[4] D. Lin and R. Morris, "Dynamics of random early detection," *SIGCOMM Symposium on Communications Architectures and Protocols*, 1996.

[5] M. Mathis, J. Semke, J. Mahdavi, and T. Ott, "The macroscopic behavior of the TCP congestion avoidance algorithm," *Computer Communication Review*, vol. 27, pp. 67–82, July 1997.

[6] T. Turletti, S. F. Parisis, and J.-C. Bolot, "Experiments with a layered transmission scheme over the Internet," *INRIA Research Report 3296*, November 1997.

[7] W.-T. Tan and A. Zakhor, "Real-time Internet video using error resilient scalable compression and TCP-friendly transport protocol," *IEEE Transactions on Multimedia*, vol. 1, pp. 172–186, June 1999.

[8] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: A simple model and its empirical validation," *Proceedings of ACM SIGCOMM'98*, vol. 28, pp. 303–314, October 1998.

[9] J. Padhye, J. Kurose, D. Towsley, and R. Koodli, "A model based TCP-friendly rate control protocol," *UMASS-CMPSCI Technical Report*, October 1998.

[10] J.-C. Bolot and T. Turletti, "Experience with control mechanisms for packet video in the Internet," *ACM SIGCOMM Computer Communication Review*, vol. 28, pp. 4–15, January 1998.

[11] R. Rejaie, M. Handley, and D. Estrin, "RAP: An end-to-end rate-based congestion control mechanism for realtime streams in the Internet," *Proceedings of IEEE INFOCOM'99*, March 1999.

[12] The VINT Project, "UCB/LBNL/VINT network simulator - ns (version 2)," `http://www-mash.cs.berkeley.edu/ns/`, 1996.

[13] W. Stevens, *TCP/IP Illustrated*. Addison-Wesley, 1994.

[14] ISO/IEC DIS 13818-2, "MPEG-2 video," *ISO standard*, 1994.

[15] K. Fukuda, N. Wakamiya, M. Murata, and H. Miyahara, "QoS mapping between user's preference and bandwidth control for video transport," *Proceedings of Fifth IFIP International Workshop on Quality of Service*, pp. 291–302, May 1997.

[16] ISO/IEC/JTC1/SC29/WG11, "Test model 5," April 1994.