ON TCP-FRIENDLY VIDEO TRANSFER WITH CONSIDERATION ON APPLICATION-LEVEL QOS

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 sessions co-exist in the current Internet environment, the performance of TCP sessions easily deteriorate because of congestion incurred by UDP sessions of real-time multimedia applications.

In this paper, we extend the TCP-friendly rate control protocol which originally pursuits the fair-share of link bandwidth among TCP and non-TCP sessions. With our proposed method, the achievable application-level QoS, such as perceived video quality and file transfer delay, becomes the same among TCP and non-TCP which traverse the same path. Through simulation experiments, we show that the high quality video transfer can be performed with our proposed method while satisfying the TCP-friendliness with regard to application-level QoS.

1. 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`, `telnet` employ TCP which accomplishes the loss-free data transfer by means of window-based flow control and retransmission mechanism. On the other hand, to avoid the unacceptable delay introduced by the retransmission, real-time multimedia applications such as video conferencing prefer UDP.

Since the multimedia application, especially when video is employed, emits considerable amounts of data, the network is easily driven to congestion by greedy UDP packets without any appropriate control mechanism. When the congestion occurs, TCP sessions shrink their congestion window `cwnd` and decrease data emission rate to recover from the congestion. This means that the performance of TCP sessions is heavily affected by UDP.

One way to achieve fairness between TCP and UDP is for intermediate nodes to employ the appropriate packet scheduling algorithm such as WFQ (Weighted Fair Queueing) and CBQ (Class-Based Queueing). However, it is required that all nodes employ the same algorithm with the same parameters. An alternative and easier way is to apply the rate control algorithm to non-TCP sessions. In recent years, numbers of researches have been devoted to achieving the fairness among TCP and non-TCP by introducing the concept of *TCP-friendliness* [1-7].

In those works, "TCP-friendliness" 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 session does. Even though the fairness among TCP and UDP sessions can be achieved by careful parameter determination in those TCP-friendly protocols, they do not take into account characteristics of video applications. For example, the rate control interval should be long enough to avoid video quality fluctuation. On the contrary, infrequent control leads to the unfairness because the video applications cannot catch up with changes of network condition.

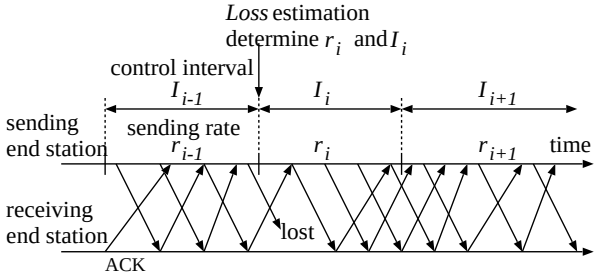

Figure 1: TCP-friendly rate control protocol

Another problem of the TCP-friendly protocols is that they regard the fair-share of link bandwidth as the inter-protocol fairness. From the standpoint of users, the fair-share of bandwidth is not necessarily identical to the fairness in terms of application-level QoS. For example, if we consider the file transfer delay as the application-level QoS of data applications and the perceived video quality as that of video applications, the achievable QoS with the same bandwidth differs among applications. In other words, the same rate given to TCP and UDP sessions does not mean the same QoS level. In this paper, we propose the rate control algorithm for real-time video applications to achieve the fairness with TCP sessions with consideration on application-level QoS.

This paper is organized as follows. In Section 2, we consider TFRCP (Tcp-Friendly Rate Control Protocol) which is suitable for video applications. In Section 3 we then introduce Q-TFRCP (Qos-based TFRCP) where the fairness with regard to the application-level QoS can be achieved and show some results obtained from simulation experiments. Finally, we summarize our paper and outline our future work in Section 4.

2. TCP-FRIENDLY RATE CONTROL PROTOCOL

In this section, we consider the TFRCP (Tcp-Friendly Rate Control Protocol) which is suitable for real-time video applications. TFRCPs proposed in [1-7] behave as illustrated in Fig.1; at the end of the control interval i-1, the server (1) estimates the network condition from information gathered within the interval, (2) then derives the throughput of a TCP session, r_{TCP}, which traverses the same path, (3) and finally adjusts its emission rate r_i for the next interval i to the estimated TCP throughput r_{TCP}.

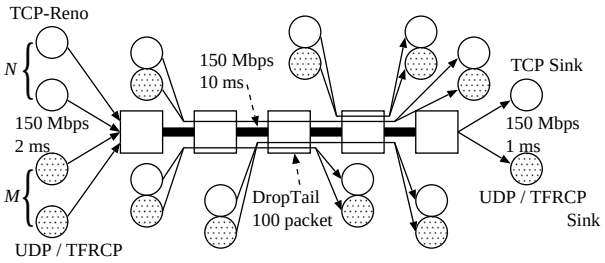

Figure 2: Network model

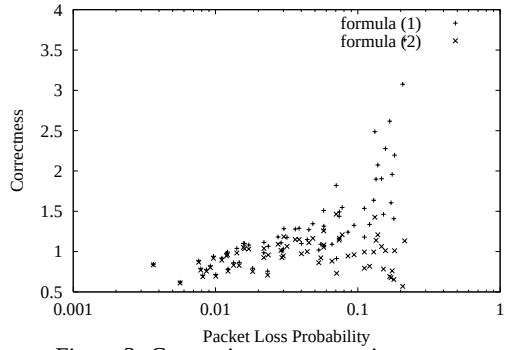

Figure 3: Comparison among estimators

2.1. TCP throughput estimation

In [1], they proposed a formula to derive the TCP throughput r_{TCP} from RTT, MTU and packet loss probability p;

$$r_{TCP} = C \times \frac{MTU}{RTT\sqrt{p}} \quad (1)$$

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. A formula proposed in [4] takes into account the timeout and the advertised window. The TCP throughput r_{TCP} is derived from the advertised window size W_{max}, MTU, RTT, RTO T_0 and packet loss probability p;

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

where b is 2 for delayed ACK and otherwise 1.

We evaluate the applicabilities of those two formulas in the network model shown in Fig.2 where TCP Reno sessions and UDP sessions coexist. Simulation experiments shown in this paper are performed with `ns version 2` [8]. The simulation results are depicted in Fig.3 where "Correctness" is given as the ratio of the estimated throughput to the actual throughput. From Fig.3, it is obvious that formula (2) is a better estimator than (1).

In this paper, we employ the formula (2) for the TCP throughput estimation. However, the formula is not applicable when no packet was lost within a control interval. Thus, according to [5], we determine the video emission rate r_i in the next interval i as;

$$\text{p>0} \quad r_i = r_{TCP} = \frac{MTU}{RTT\sqrt{\frac{2p}{3}}+T_0\,\min(1,3\sqrt{\frac{3p}{8}})p(1+32p^2)} \quad (3)$$

$$\text{p=0} \quad r_i = 2 \times r_{i-1} \quad (4)$$

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

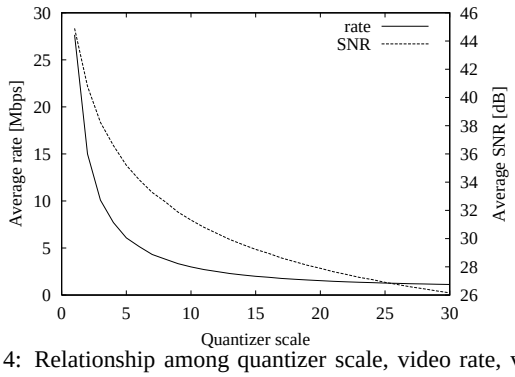

Figure 4: Relationship among quantizer scale, video rate, video quality

2.2. Estimation of network condition

In TFRCP, network condition is expressed by RTT and packet loss probability. To estimate the TCP throughput by formula (2), those values must be accurately determined from informations gathered within a control interval. RTT is obtained by observing timestamps added to packet headers in [1, 3-5, 7] or exchanging RTCP (Real-Time Control Protocol) packets [6]. Packet loss probability is derived using sequence numbers [6, 7].

In this paper, we estimate RTT by timestamp-based approach and packet loss probability by observing sequence numbers of ACKed packet. RTT and RTO are smoothed by Jacobson's algorithm [9] and packet loss probability is given as the ratio of number of lost packet to the number of emitted packet within a control interval.

2.3. Video rate control

Once the TCP throughput is successfully estimated, the video server should adjust its video emission rate to the estimated throughput by regulating video coding rate. In this paper, we consider MPEG-2 [10] as a video coding algorithm. In MPEG-2, each captured picture is first discrete cosine transformed and each DCT coefficient is then quantized with specified quantizer scale. Thus, the coded video quality and amount of data can be regulated by controlling the quantizer scale.

In Fig.4, we depict an example of the relationship among the quantizer scale, the average video rate and the video quality in terms of SNR (Signal to Noise Ratio) when a video sequence is coded by MPEG-2 VBR coding algorithm where a static quantizer scale is applied to all pictures in a sequence [11]. 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 [11], we can easily determine the appropriate quantizer scale to adjust the video rate to the estimated TCP throughput.

2.4. Control interval appropriate for video application

As described in Section 1, the duration of each control interval I_i (Fig.1) must be carefully determined to accomplish the effective rate control. It is expected that the fairness can be achieved by frequent rate control (ideally in the same order as TCP), but the server cannot gather enough feedback information to estimate TCP throughput within a short interval and the frequent video rate control results in the unacceptable video quality fluctuation. On the other hand, if the control interval is too long, the video emission rate does not reflect the dynamics of network condition and the fair-share of bandwidth cannot be achieved.

In this paper, we employ the duration of 32 times as long as smoothed RTT [3]. With consideration on characteristics of MPEG-2 video traffic which consists of the repetition of sequence of three

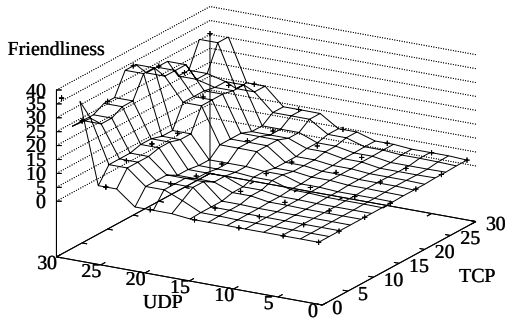

Figure 5: Friendliness (UDP)

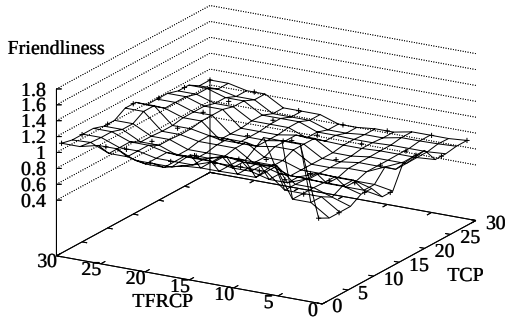

Figure 6: Friendliness (TFRCP)

types of coded pictures (called GoP, Group of Pictures), the control interval is rounded to the GoP time unit. GoP time unit is given as the multiple of the number of pictures in a GoP, 30 in this paper, and the frame rate, 29.97 fps, that is, 30/29.97=1.00 sec.

In Figs.5 and 6, the friendliness variation is depicted against combinations of variety numbers of TCP and video sessions in the network (Fig.2) where video applications employ UDP and TFRCP, respectively. In simulation experiments, M sessions of video applications produce the video traffic whose characteristics are shown in Fig.4. CBR traffic of 27 Mbps which is identical to the maximum rate of video traffic as shown in Fig.4 for comparison purpose is injected into N TCP sessions. External TCP and video sessions emits coded video traffic whose averaged rate is about 5.4 Mbps. The clock granularity of TCP Reno is set at 10 msec. Packet size is 1000 Bytes and identical among all sessions.

In those figures, the friendliness is given by dividing the average throughput of video sessions by that of TCP. Comparing those figures it is obvious that our control algorithm provides the fair-share of bandwidth among TCP and video sessions. In addition, the perceived video quality with TFRCP is higher than that with UDP although not shown in the figure. We examined other control intervals and observed that TFRCP became aggressive with shorter interval such as 16×RTT and the friendliness decreased with longer interval.

3. TFRCP WITH CONSIDERATION ON APPLICATION-LEVEL QOS

The TFRCP proposed in the previous section provide the fair-share of bandwidth among TCP and video sessions as those in the previous studies. However, application-level QoS which users directly perceive are not necessarily identical among TCP and video sessions even if the occupied bandwidth is exactly the same. In this paper, we consider the file transfer delay and the video quality as

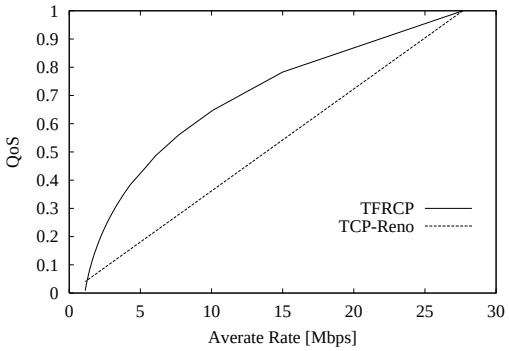

Figure 7: Relationship among application-level QoS

the application-level QoS of the data transfer and the video application, respectively. Doubled throughput results in the doubled service quality in the data transfer application because the file transfer delay becomes the half. On the contrary, the increase in the perceived video quality with additional throughput is not always twice as shown in Fig.4.

In this section, we propose the QoS-based TFRCP (Q-TFRCP) where the inter-protocol fairness with regard to the application-level QoS is achieved. To accomplish the application-level fairness among TCP and video sessions, Q-TFRCP should estimate the application-level QoS of TCP from the estimated TCP throughput, and then determine the video rate with which the same application-level QoS can be achieved in Q-TFRCP.

3.1. Estimation of application-level QoS

The application-level QoS in Q-TFRCP is given as the perceived video quality. The appropriate quantizer scale to achieve the specific QoS can be easily derived from the relationship depicted in Fig.4. Here, we employ SNR as the measure of the perceived video quality. To normalize SNR values comparable to the file transfer delay, we should have knowledge of the relationship among SNR and QoS, that is, how much SNR is regarded as QoS=1 where all users are satisfied with the perceived video quality and how much SNR means QoS=0. In this paper, we consider QoS becomes 1 when the video is coded with the smallest quantizer scale because no higher quality can be expected in MPEG-2 coding algorithm. QoS values for other quantizer scales are derived by dividing the achievable SNR by the maximum SNR (44.898 dB in Fig.4).

On the other hand, the application-level QoS in TCP is specified by the file transfer delay. For Q-TFRCP to estimate the TCP QoS from the estimated TCP throughput r_{TCP}, the video server must know the file size of data transfer application and the relationship among the delay and QoS. However, it is obviously difficult because there is no signaling protocol to exchange such informations among TCP and Q-TFRCP sessions. In this paper, we propose the mechanism where the video server can estimate the TCP QoS based on its local informations such as feedback from clients and video characteristics. It is assumed that the video server consider that the highest QoS is achieved when the throughput of a TCP session is identical to the maximum rate of video traffic (about 27 Mbps in our experiments). Thus, the estimated TCP QoS can be relatively derived by dividing the estimated TCP throughput by the maximum video rate. In Fig.7, we summarize the relationship among the application-level QoS of the data transfer and the video.

3.2. QoS-based TCP-friendly Rate Control Protocol

The QoS-based TCP-friendly rate control protocol (Q-TFRCP) behaves as follows; at the end of the control interval i-1, the server (1)

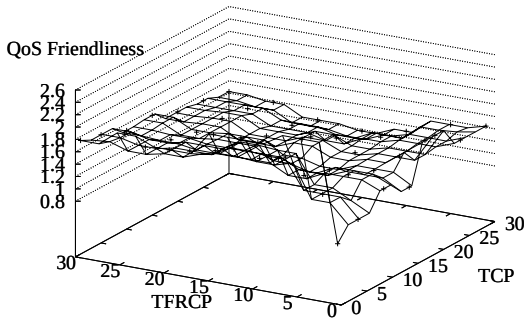

Figure 8: QoS friendliness (TFRCP)

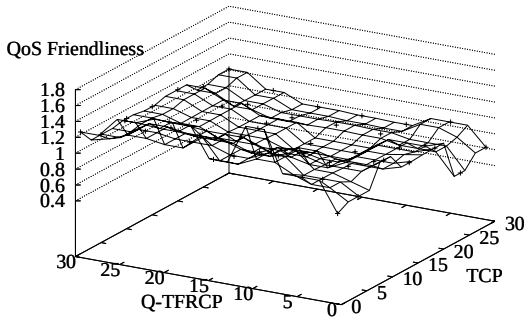

Figure 9: QoS friendliness (Q-TFRCP)

estimates the network condition from information gathered within the interval, (2) then derives the throughput of a TCP session r_{TCP} which traverses the same path, if there is no packet loss in the interval i-1, the estimated TCP throughput is twice as large as r_{TCP} of interval i-1, (3) derives the TCP QoS from r_{TCP}, (4) and finally adjusts its emission rate r_i with which it can achieve the same QoS as TCP from the relationship shown in Fig.7.

Fig.8 depicts the QoS friendliness in TFRCP for the simulation experiments in Fig.6. In the figure, the QoS friendliness for each set of parameters is derived by dividing the average QoS obtained in TFRCP by that in TCP. As shown in Fig.8, the fair-share of the bandwidth (Fig.6) does not necessarily satisfy the inter-protocol fairness with regard to the application-level QoS. By applying Q-TFRCP where the video applications regulate their emission rate with consideration on the application-level QoS-based fairness, the QoS friendliness is improved as shown in Fig.9. To further compare TFRCP and Q-TFRCP, we show the cumulative probability distribution of the QoS friendliness in Fig.10. As shown in the figure, the QoS-based fairness of Q-TFRCP is higher than that of TFRCP.

4. CONCLUSION

In this paper, we proposed the QoS-based TCP-friendly rate control protocol with which the inter-protocol fairness in terms of the application-level QoS is achieved. The proposed control is obtained by small modification to the TFRCP introduced in the paper where we investigate the control strategy which is preferable for video applications. We are currently trying to implement Q-TFRCP on RTP.

In this paper, we give $32 \times$RTT as a control interval, but the appropriate value should depend on the network model and conditions. It seems preferable that a TFRCP session bahaves similarly as a TCP session on the same path. We are investigating the dynamic interval adjustment strategy to achieve such control. Another problem in our proposed control is that the perceived video

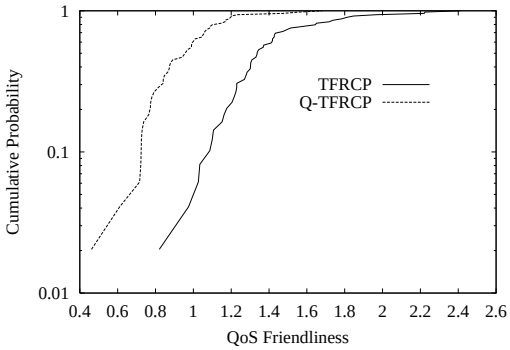

Figure 10: Cumulative probability distribution of QoS friendliness

quality could suddenly deteriorate because of the instantaneous congestion. When congestion occures, the video server degrades the video quality to regulate its emission rate according to the TCP throughput. The target rate is usullay relatively smaller than the previous one and we should employ some mechanism to avoid such rate collapse.

5. REFERENCES

[1] 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.

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

[3] 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.

[4] 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.

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

[6] 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.

[7] 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.

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

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

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

[11] 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.