Journal of Zhejiang University SCIENCE A
ISSN 1009-3095 (Print); ISSN 1862-1775 (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

On optimal receiver buffer size in adaptive Internet video streaming

OZBEK Nukhet†, TUNALI E. Turhan
(*International Computer Institute, Ege University, Bornova, Izmir 35100, Turkey*)
†E-mail: nukhet.ozbek@ege.edu.tr
Received Dec. 12, 2005; revision accepted Feb. 21, 2006

Abstract: The effect of receiver buffer size on perceived video quality of an Internet video streamer application was examined in this work. Several network conditions and several versions of the application are used to gain understanding of the response to varying buffer sizes. Among these conditions local area versus wide area, bandwidth estimation based versus non-bandwidth estimation based cases are examined in detail. A total of 1000 min of video is streamed over Internet and statistics are collected. It was observed that when bandwidth estimation is possible, choosing larger buffer size for higher available bandwidth yields quality increase in perceived video.

Key words: Video streaming, Receiver buffer size
doi:10.1631/jzus.2006.AS0112 **Document code:** A **CLC number:** TN919.8

INTRODUCTION

Internet video streaming has lately become a popular application. However, continuously varying network conditions leading to variations in bandwidth, delay and loss of packets degrade the perceived video quality. On the other hand, when massive video data is transmitted over relatively limited bandwidth channels, streaming itself causes congestion on the channel. This latter point has been studied (Tunali *et al*., 2005). It was pointed out by many researchers that video applications should be fair to other applications that have comparably low data rates (Balk *et al*., 2003). This topic is often phrased as "TCP friendliness". All of these points have lead to the development of "adaptive video streaming" applications that feature "rate control" in the response to changing network conditions. In a typical algorithm, spatial and temporal resolution parameters are seamlessly modified by the sender to change transmitted data rate (Rejaie *et al*., 2000).

On the other hand, at the receiver side, the perceived video quality varies due to several reasons. First, any decrease in data rate causes decrease in spatial resolution, or temporal resolution, or both. Second, even if the video quality is good, frequent changes in video parameters also degrade the perceived video quality. Third, jitter may cause serious problems. Last but not least, drain of receiver buffer causes interruption of the decoding and display process. Even though there are no well defined performance measures for the streamed video data. The above three reasons are considered to be widely accepted (Wu *et al*., 2001).

With all the above facts in mind, adaptive video streaming problem is formulated as determination of an algorithm for modifying the video parameters under changing network conditions so that the perceived video quality is maximized in some sense. At this point, it is important to understand what we can use as a measure of changing network conditions. Typical network parameters are available bandwidth, delay and loss rate of packets. Among these parameters, loss rate can be measured in a relatively short time interval. In other words, it can be easily integrated into an adaptation algorithm within an acceptable parameter modification period. On the other hand, delay and bandwidth are more difficult to measure. In particular, precise estimation of available bandwidth in a short time interval may not be feasible.

An adaptive algorithm should respond to congestion while trying to maintain an acceptable receiver buffer level. It should absorb jitter by buffered data. When congestion is over, it should try to increase the video quality. Furthermore, it should avoid fluctuations in video parameters. Considering these issues, an algorithm is developed in (Tunali *et al.*, 2005) and its performance is measured in (Tunali *et al.*, 2004). It has been stressed that the receiver buffer policy plays a crucial role in the overall performance of streamers. In this study, the effects of changing receiver buffer size on the algorithm in (Tunali *et al.*, 2004) are examined. In addition to the three perceived quality measures given above, initial waiting time can be considered as an additional measure because the person at the receiver side does not want to wait a long time before the display starts. It is natural to expect good perceived quality with streaming done on a high bandwidth network with relatively low initial buffer filling period and vice versa. However, it should be noted that in adaptive algorithms, high bandwidth causes encoding of high quality video. If any congestion starts right after the filling period, receiver buffer level decreases and the adaptation algorithm responds by decreasing the video quality. This, in turn, causes frequent quality switches, because high bandwidth does not necessarily provide stable receiver buffer level for the streaming application. We will address this issue in the following sections.

The paper is organized as follows: In Section 2, the features of the previously developed adaptation algorithm will be reviewed. In Section 3, the design of experiments will be explained. In Section 4, experimental results will be given. Finally, in Section 5, concluding remarks will be made.

RATE ADAPTIVE VIDEO STREAMING ALGORITHM

In this section, our previously developed streaming algorithm (Tunali *et al.*, 2005) is briefly reviewed. Rate adaptation is achieved by video scaling in a seamless manner via frame dropping, switching to another encoding rate, or changing the packet interval. MPEG-1 video is encoded at multiple encoding rates and stored in the database. A metadata file is prepared by packetization module for every encoding rate of the video. Packetization module determines the number of packets to be sent per video file for each encoding rate and frame discard level pair. The transmission interval between consecutive packets is calculated as video duration divided by the number of packets for each pair. Packet interval values are also stored in the metafile.

Frame dropping is performed in levels. Considering the dependency among frame types, the adaptation module drops first B-frames then P-frames if necessary from the current GOP pattern. Each encoding rate (*er*) and frame discard level (*fdl*) pair corresponds to a different transmission rate in the network. Encoding rates have values as 1000, 500, 200, 100 kbps. Frame rates have values as 30, 20, 10, 5, 1 fps. A grid is formed by using *er* and *fdl* combinations. On this grid, depending on the receiver buffer and the network congestion status, the appropriate *er*-*fdl* pair is chosen for adaptation that follows an AIMD (Additive Increase Multiplicative Decrease) strategy to preserve TCP-friendliness.

The receiver periodically sends smoothed loss rate (*slr*), current stored video duration (time-to-play), denoted *ttp* and rate of change of *ttp* (*dttp*) to the sender. Examining these data and current values of the controlled variables *er*, *fdl* and packet interval (*pi*), the algorithm running at the sender decides on the tradeoff between the video quality and congestion.

The algorithm follows a conservative approach by allowing quality degradations after two adaptation requests and quality increases after 5 adaptation requests. We observed that a non-conservative approach that reacts to adaptation requests immediately resulted in frequent rate oscillations, displeasing the viewer. The conservative approach based on a hysteresis model preserved the prevailing quality until an indicator of persistent behaviour in congestion and buffer status is available, thereby eliminating the disturbance of the viewer.

The algorithm has a content-aware media scaling system. When adaptation is required, quality scaling is used by switching to a version at a lower encoding rate during the transmission of the video segments which contain high motion whereas temporal scaling (i.e., frame dropping) takes precedence over quality scaling during the transmission of the video portions with low motion content.

In addition to the observed variables above, the algorithm has a bandwidth estimator module. The estimated bandwidth value lets the algorithm choose the most appropriate encoding rate to start streaming. Combining available bandwidth estimation with rate adaptation has two important advantages. First, available bandwidth is used to decide on the quality of the initial video to be sent during initial buffer filling period. Hence, heavy load during initial buffer filling period is avoided. Second, when the quality is to be increased, available bandwidth is very useful for understanding whether the current channel capacity meets the new bandwidth requirement. By this way, the unnecessary quality increases in the adaptation algorithm, which may cause packet loss and congestion, are avoided.

Depending on the observed measures of *slr*, *ttp*, *dttp*, video dynamics and current values of *er-fdl-pi*, the algorithm determines new values for *er-fdl-pi*. If conditions impose a decrease in quality, than no further step is taken and new values of *er-fdl-pi* are applied to streaming. Since RTCP reports are at least as fast as our bandwidth estimator in informing the congestion to the sender, the estimated bandwidth value is not used when the quality is to be decreased. Hence, no advance information is available in this particular case. If new adaptation is not in the direction of decreasing data rate, then new proposed data rate (*put_bw*) is compared with estimated available data rate. If *put_bw* is less than the available bandwidth, then the chosen *er-fdl-pi* is applied to streaming. If *put_bw* is more than available bandwidth, the current *er-fdl-pi* values remain the same. This last case occurs when the streaming application is already using up a hefty part of bandwidth and there is no more bandwidth available for quality increase. A sketch of the algorithm is given in Fig.1 where the procedures *down_scale_video* and *up_scale_video* correspond to the movements along the grid paths given in (Tunali *et al*., 2005) for rate decrease and increase respectively.

The algorithm uses three constant buffer threshold values: The first one is underflow threshold (*ttp-unfl*) that is an indication that the receiver buffer is about to drain. The second one is overflow threshold (*ttp-ovfl*) that is an indication that the buffer is about to be full. The scale is seconds of video stored in buffer (*ttp*). The algorithm tries to maintain a buffer

```
bandwidth_aware_scale( ){
  observe loss_rate, ttp, dttp, video_dynamics, available_bw;
  if (congestion)
       down_scale_video (loss_rate, ttp, dttp, video_dynamics);
  else
     compute put_bw;
     if (put_bw<available_bw)
          up_scale_video (ttp, dttp, video_dynamics);
     else
          no_scale; }
```

Fig.1 Bandwidth-aware video scaling algorithm

level that is in-between these two thresholds (*adj-thr*). Finally, *byte-ovfl* is a threshold in bytes to halt streaming for a short period to prevent physical buffer overflow. This latter threshold is necessary because previous thresholds introduced are all in seconds and physical memory is in bytes. Different quality of video stored may yield different corresponding byte values for *ttp-ovfl*. Initial buffer filling period (*ibfp*) is fixed and determined by simply postponing decode and display until the video level reaches (4/3)× *adj-thr* so that display process can start with a sufficient initially stored video. The quality of the video during the initial buffer filling phase is determined by the estimated available bandwidth.

DESIGN OF EXPERIMENTS

In the algorithm reviewed in Section 2, using a constant buffer size may not be the best choice under varying network conditions. The essential idea in buffering received video is to eliminate the variations of effective incoming data rate. Unfortunately, these variations are random in nature. Hence, constant threshold values may not respond to this randomness. In (Loguinov and Radha, 2001), extensive measurements are carried out to characterize statistical properties of video data. Tao and Gurein (2004) and Wang *et al*.(2002) developed strategies suitable for online estimation of loss rate and available bandwidth respectively. The choice of buffer size, however, is not a straightforward implication of loss rate and bandwidth. Quality factors such as video encoding rate, number of adaptations in unit time and interrupt of display indicate that the higher the buffer size the better. If buffer size is not sufficiently high, the receiver buffer may drain and display may be interrupted. On the other hand, the user is in general impatient and wishes to start watching the video in reasonable time. After a short observation period, the streamer has to set its parameters such as *ttp-unfl*, *adj-thr*, *ttp-ovfl*, *ibfp*, *er*, *fdl*, *pi* and start sending packets. There is not enough time to collect sufficient statistics for detailed analysis. Hence, the trade-off between low and high buffer size must be carefully tailored to the streaming application.

In this study, we carried out real streaming experiments to observe the effect of buffer size on the quality of the perceived video. We have examined both WAN and LAN conditions. Since we have no control on the actual network, we carried out each streaming experiment five times to obtain more reliable statistics. We considered the case when available bandwidth estimation is not possible. This latter situation may occur when a portion of probe packets of available bandwidth estimator are lost. To examine such a case, we turned off the estimator in a particular set of experiments. In fact, turning the estimator off does not perfectly create the congestion conditions, because the adaptation algorithm heuristics is due to the observed video packet loss rate whose value can only increase when packets are in fact lost. On the other hand, choosing packets manually may not simulate the actual conditions due to several reasons: First, in classical wired network, increase in loss rate usually comes with decrease in bandwidth and modifying loss rate manually does not affect bandwidth; second, if a bandwidth simulator is used, such a simulator delays packets itself, and in this case, our available bandwidth estimator module fails to succesfully estimate bandwidth.

PERFORMANCE RESULTS

The experiments mentioned in the previous section were carried out with the video streaming system that we summarized in Section 2. In the system, RTP was used for data transfer and RTCP was used to collect network statistics. Control messages are exchanged over UDP. Our streaming system has client-server architecture. The client requests video from the server. Server streams video to the client in a unicast manner. Both server and client software are multithreaded. The pipelined architecture of the client software further increases the performance of the whole system. Detailed explanation of our testbed can be found in (Tunali *et al*., 2004).

As we mentioned in Section 3, we conducted WAN and LAN experiments in which we considered the cases where bandwidth estimator is turned on and off. WAN experiments were performed in the actual Internet environment between two Sun Ultra 5 workstations. The workstation working as the streaming server is in Koc University Campus in Istanbul, Turkey. The client workstation is in Ege University Campus in Izmir, Turkey. Traceroute command shows that the number of hops between the two workstations is 9. Average available bandwidth measured was approximately 540 kbps throughout the WAN experiments. LAN experiments were carried out between two Sun Ultra 5 workstations located in our laboratory. Average available bandwidth measured was approximately 50 Mbps throughout the LAN experiments.

We have parameterized receiver buffer thresholds as follows: We set parameter *size*, into five different values as 1, 2, 3, 4 and 5. Then we set scale coefficients (*scale*) of *ttp-unfl*, *adj-thr*, *ibfp* and *ttp-ovfl* as 5, 9, 12 and 20 respectively. To determine a particular threshold value, we multiply the *scale* value of that threshold by *size*. For example, for *size*=2, *ttp-ovfl*=40 s. In the following, we will simply refer to a buffer size by its corresponding *size* value.

Each streaming session used the same MPEG-1 video which is 10 min long. Since we repeated each streaming session five times, four combinations of LAN-WAN versus available bandwidth estimator on and off and five values of *size* together with five repetitions add up to a total of 100 experiments each lasting 10 min add up to a total of 1000 min of video streamed over Internet.

Tables 1, 2, 3 and 4 (where "av #" means "average number of") give the statistics collected during the experiments. Column 1 gives the buffer size. Column 2 gives the average *er* value of the video session. The *er* values used are 0, 1, 2 and 3 for the rates of 1000, 500, 200 and 100 kbps respectively. The smaller the average *er* value, the higher the quality. Column 3 gives the total number of changes of *er* during the whole transmission. The smaller the total number of changes, the higher the perceived quality. Column 4 gives the average frame discard level. The values used are 0, 1, 2 and 3 for no frame discard, small, medium and high frame discard levels respectively. The smaller the frame discard level, the higher the quality. Column 5 gives the total number of frame discard level changes during the whole transmission. The lower the discard level changes, the higher the quality.

Table 1 WAN test results with bandwidth estimator

size	Average *er*	av # *er* switches	Average *fdl*	av # *fdl* switches
1	1.21	6.80	0.15	7.40
2	1.29	5.20	0.24	7.60
3	1.23	4.00	0.14	4.80
4	1.37	9.00	0.32	10.80
5	1.35	5.00	0.44	10.20

Table 2 WAN test results without bandwidth estimator

size	Average *er*	av # *er* switches	Average *fdl*	av # *fdl* switches
1	1.33	9.50	0.32	13.50
2	1.35	5.80	0.26	9.60
3	1.19	3.60	0.18	5.60
4	1.16	3.20	0.08	2.00
5	1.19	2.80	0.14	5.60

Table 3 LAN test results with bandwidth estimator

size	Average *er*	av # *er* switches	Average *fdl*	av # *fdl* switches
1	0.82	17.25	0.52	21.75
2	0.91	12.25	0.41	14.75
3	0.90	11.00	0.51	12.75
4	0.93	8.00	0.38	9.50
5	1.01	6.00	0.22	6.50

Table 4 LAN test results without bandwidth estimator

size	Average *er*	av # *er* switches	Average *fdl*	av # *fdl* switches
1	0.94	21.80	0.50	29.40
2	0.94	12.60	0.42	13.00
3	0.93	11.20	0.39	11.20
4	1.05	11.60	0.34	8.40
5	1.06	6.00	0.11	5.00

Having explained the type of statistics collected, we can now analyze the data. We will first compare WAN and LAN results when available bandwidth estimator is on. As buffer is increased, average encoding rate slightly decreases in both cases. This may be due to the fact that, the streaming algorithm rapidly decreases the quality if loss rate increases or buffer level decreases and reluctantly increases quality if buffer is not full. With larger buffer size, a good amount of time may be needed to fill the buffer above a certain level before quality is increased. On the other hand, as buffer size increases, even though there is no clear pattern of change in the number of *er* changes in WAN case, there is a considerable decrease in number of *er* changes in the LAN case. This latter point suggests that under LAN conditions, choice of large buffer should be preferred whereas under WAN conditions medium sized buffer may suffice. Note that as to buffer size, we are talking about duration of video stored rather than a physical data size. In high bandwidth case, the video streamed is of higher quality and has higher data rate. Hence, we can expect that the relation between allowed physical buffer size and available bandwidth is nonlinear in the LAN case.

When *fdl* and number of *fdl* changes are examined, similar argument applies. Figs.2 and 4 give *ttp* histograms for WAN and LAN cases respectively when the bandwidth estimator is on. Each bar represents the total number of times that the *ttp* value was in a particular 5 s interval. We note that we would not like to see high values towards 0 s because such cases are good representatives of display interrupts. In the WAN case, when *size*=3, the buffer level never goes below 10 s which can be considered as a pretty safe situation as far as display interrupts are concerned. In the LAN case, even though *size*=3 or *size*=4 could be chosen for display interrupt optimization, we recall that video quality is considerably low in these cases. As a result, we can suggest that, if bandwidth estimation is possible, *size* can be chosen as 5 for high bandwidth and 3 for low bandwidth.

If network is loaded and there are occasional packet losses, available bandwidth estimation may not be possible. Under such a situation, even though we examine both LAN and WAN cases, we will not be able to suggest different buffer sizes for different cases since the algorithm will have no chance to figure out the situation. As far as Tables 2 and 4 are concerned, there does not seem much difference in the quality of video in WAN case.

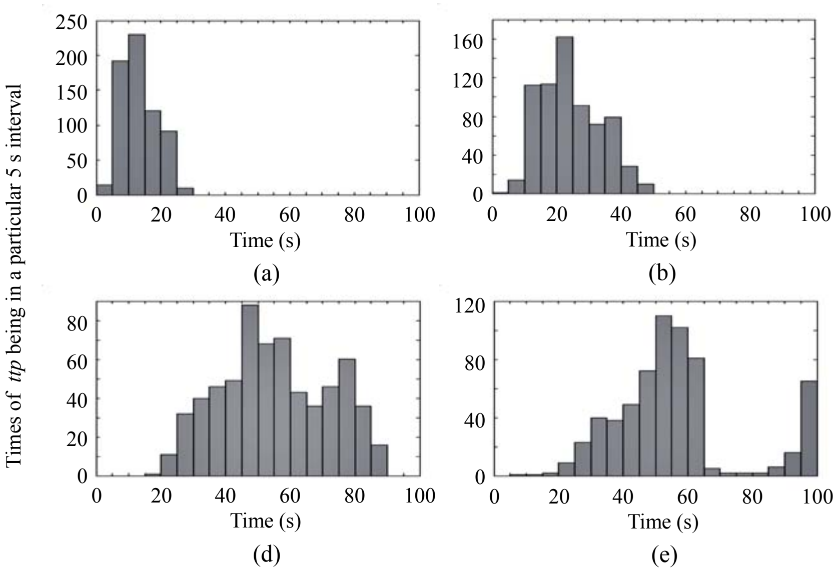

Fig.2 ***ttp*** **histograms of WAN w/** ***bw*** **estimator. (a)~(e) denote** ***size*****=1, 2, 3, 4, 5, respectively**

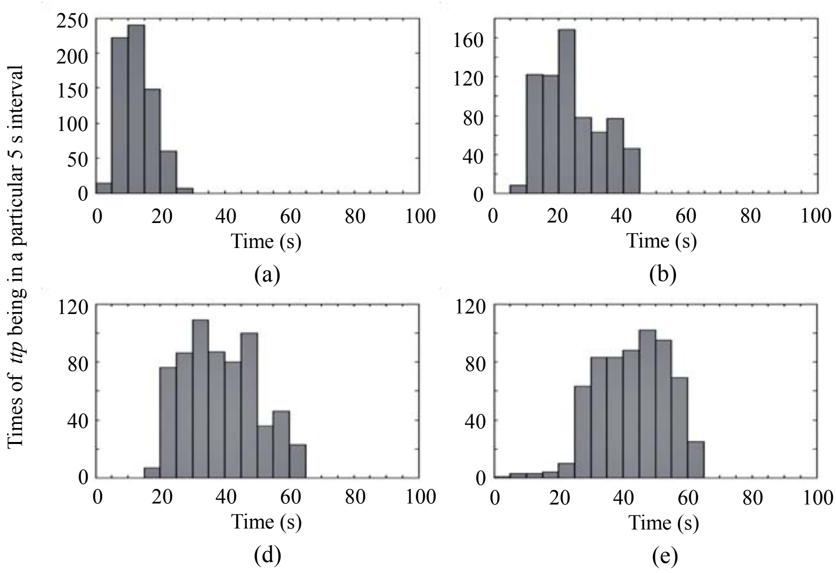

Fig.3 ***ttp*** **histograms of WAN w/o** ***bw*** **estimator. (a)~(e) denote** ***size*****=1, 2, 3, 4, 5, respectively**

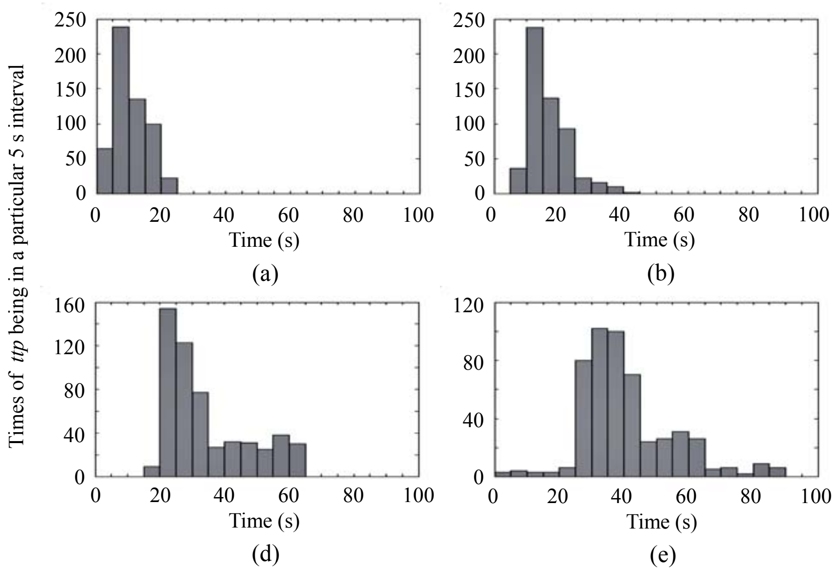

Fig.4 ***ttp*** **histograms of LAN w/** ***bw*** **estimator. (a)~(e) denote** ***size*****=1, 2, 3, 4, 5, respectively**

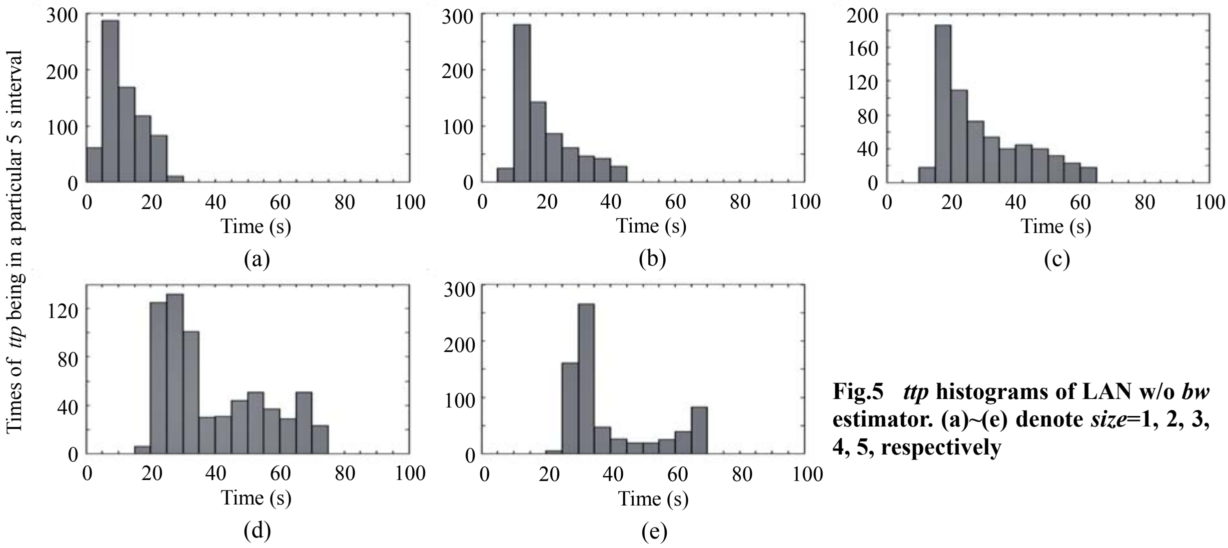

Fig.5 *ttp* histograms of LAN w/o *bw* estimator. (a)~(e) denote *size*=1, 2, 3, 4, 5, respectively

However, in the LAN case, we note that the number of switches is considerably reduced with increasing buffer. Examination of Fig.5 above showed that choosing larger buffer reduces display interrupt probability in the LAN case, and in particular, *size*=5 gives the best value. Fig.3 indicates that *size* values 3, 4 or 5 are all suitable for the WAN case. Even though 5 is not the optimal choice in the WAN case, very small values for the low buffer indicate that display interrupt probability is very low. As a result, we suggest that when no bandwidth estimation is possible, buffer with *size*=5 should be chosen.

REMARKS

In this study, we addressed the problem of choosing optimal receiver buffer size to maximize the perceived video quality in Internet video streaming. We collected actual statistics on a total of 1000 h of streamed video through Internet. We particularly examined LAN and WAN cases and observed that, unexpectedly, streaming through LAN requires larger buffer size. This is particularly necessary to minimize the number of quality switches. Since LAN has higher bandwidth, even though, larger buffer implies longer video stored in the buffer before the display starts, this does not necessarily mean that the viewer has to wait a longer time. Our streaming algorithm has an available bandwidth estimator module and if there is no congestion, the algorithm can determine available bandwidth in 5 s intervals periodically. After an initial estimate of bandwidth, the buffer size can be chosen as suggested in Section 4 to optimize the overall perceived quality.

References

Balk, A., Maggiorini, D., Gerla, M., Sanadidi, M.Y., 2003. Adaptive MPEG-4 Video Streaming with Bandwidth Estimation. Proceedings of the 2nd International Workshop on Quality of Service in Multiservice IP Networks.

Loguinov, D., Radha, H., 2001. Measurement Study of Low-bitrate Internet Video Streaming. Proceeding of ACM SIGCOMM Internet Measurement Workshop. San Fransisco.

Rejaie, R., Handley, M., Estrin, D., 2000. Layered quality adaptation for Internet video streaming. *IEEE Journal of Selected Areas of Communications*, **18**(12):2530-2543. [doi:10.1109/49.898735]

Tao, S., Guerin, R., 2004. On-line Estimation of Internet Path Performance: An Application Perspective. Proceedings of IEEE INFOCOM.

Tunali, T., Ozbek, N., Anar, K., Kantarci, A., 2004. Bandwidth Aware Scaling for Internet Video Streaming. Proceedings of the 19th International Symposium Computer & Information Sciences, **3280**:157-166.

Tunali, T., Kantarci, A., Ozbek, N., 2005. Robust quality adaptation for Internet video streaming. *Journal of Multimedia Tools and Applications, Kluwer Academics*, **27**: 341-357.

Wang, R., Valla, M., Sanadidi, M.Y., Gerla, M., 2002. Using Adaptive Rate Estimation to Provide Enhanced and Robust Transport over Heterogeneous Networks. ICNP. Paris, France.

Wu, D., Hou, Y.T., Zhu, W., Zhang, Y.K., Peha, J.M., 2001. Streaming video over the Internet: approaches and directions. *IEEE Trans. on Circuits and Systems for Video Tec.*, p.282-300.