An Adaptive Feedback Rate Control Algorithm for Unicast Video Transmission

주유*, 장경식*
*한국기술교육대학교 전자학과
e-mail : ruru605@163.com

ZhouRu*, Kyung-Sik Jang*
*Dept. of Computer Science, Korean University Technology and Education

Abstract

In this paper, we propose a novel probing mechanism for adaptive transmission of video data based on congestion control and client state. The proposed mechanism is friendly to the network dominant transport protocols (TCP) and can reduce fluctuations compared to the previous works.

1. Introduction

Due to the explosive growth of the Internet and increasing demand for video information on the web, streaming video data over the Internet has received tremendous attention. However, the current best-effort Internet does not offer any QoS guarantees to streaming video.

In this paper, we propose a novel adaptive algorithm for monitoring the network condition and client condition and then estimating the appropriate transmission rate for adapting conditions.

Our algorithm is based on the probing mechanism. Compared with other probing mechanisms, firstly, we propose to use packet loss rate as an important factor instead of packet loss rate. We believe that the LER can perform much more stable than packet loss rate in the case of transient period of severe congestion, as a result, we can get better sending rate performance in the aspect of fluctuations than before. Secondly, we add jitter as another symbol of congestion. Providing friendly behavior to other network is also an important factor. We chose TCP as the dominant protocol for testing friendliness.

The proposed mechanism is based on the Real-Time Transport Protocol (RTP). RTP provides RTCP reports mechanism which can transmit metrics for deciding network conditions. We calculate network metrics and client metrics out at the client side and the values are transmitted by RTCP, the server uses those values to estimate the network and client condition out. According to the network and client conditions, the server side can adjust sending rate.

The rest of this paper is organized as follows: we present some of the related work in Section 2. In Section 3, we present the architecture of the adaptive transmission mechanism. Section 4 presents our novel algorithm. In Finally, Section 5 concludes the paper and presents our future work.

2. Related Work

QoS control mechanism of video transmission is divided into two main kinds, one is based on network condition while the other one is based on end systems. The former one is mainly provided by routers such as doing not drop packets randomly while congestion, but according to some priority of important factors [1]. There are some examples such as RSVP[2], DiffServ, IntServ etc. However, these models can not be implemented widely nowadays because of the large cost. The latter has no requirement for network but adds control mechanism at end systems to get the best video quality as they can.

The end system-based mechanism consists of congestion control and error control. Congestion control is employed to prevent packet loss and reduce delay while error control on the other hand, is to improve video presentation quality in the presence of packet loss.

The congestion control must include rate control and rate adaptive encoding or rate shaping. The concept of rate control is to adapt the sending rate to the available bandwidth in the network. During video transmission, congestion can make large amount of packet loss and long delay which are disasters to video quality. On the other hand, if video transmission rate is lower than available bandwidth, we can not make good use of network resources fully. So the key issue of rate control is to estimate the available bandwidth accurately for video transmission while rate adaptive encoding or rate shaping is responsible for adjusting the sending rate to the target value.

Existing rate control can be classified into three categories: source-based, receiver-based, and hybrid-based control. The source-based rate control means that the sender is responsible for adapting the video transmission rate. It can be applied to both unicast and multicast. The receiver-based rate control means that the receiver regulates the receiving rate of video streams by adding/dropping channels while sender does not participate in rate control. Receiver-based is mainly involved in multicast.

We concern about source-based control. For unicast video, existing source-based rate-control mechanisms follow two approaches: probe-based and model-based. In the case of probe-based approach, the source probes for the available network bandwidth by adjusting the sending rate in a way that some network requirements are reached, such as the packet loss ratio below a certain threshold [3]. Existing ways to adjust the sending rate includes AIMD (Additive Increase Multiple Decrease) and MIMD (Multiple Increase Multiple Decrease). AIMD has been widely used to adjusting sending rate such as RAP[4]. In the case of model-based approach (also called equation-based), the throughput model of TCP

connection is used to determine the sending rate of the video stream. Thus the video connection could avoid congestion in a similar way to that of TCP and can be friendly with TCP flows. A more detailed and fully description of related work can be found in [5].

In this paper, we propose a novel probing mechanism for source-based rate control. Our adaptive feedback rate control algorithm adjust sending rate accordingly based on the network and client condition. It is TCP-friendly and can reduce fluctuations.

3. The architecture of adaptive transmission mechanism

The proposed architecture of real-time video transmission is showed in Figure 1.

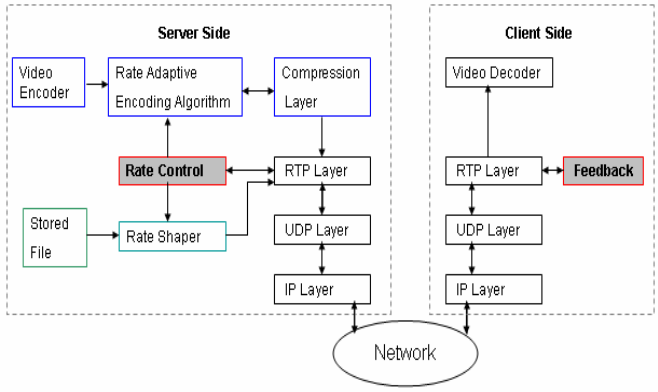

Figure.1 Architecture for transporting real-time video

The details of each module are described as follows:
Server Side:

●***Rate-adaptive Encoding Algorithm***: This module's target is to maximize the perceptual quality under a given encoding rate got from rate control. It is purely on the compression approach. This module is used when transporting live video.

●***Rate Shaper***: This module is responsible for adapting the rate of compressed video bit streams to the target rate got from Rate Control module. It is an interface between the encoder and the network and is applicable to any video coding scheme and is applicable to both live and stored video.

●***Compression Layer***: This module compresses the live video based on a rate-adaptive encoding algorithm.

●***Rate Control***: This module is responsible for the analysis of feedback information from receiver report. Based on the algorithm we propose in this paper, it calculates the proposed transmit rate value out and pass that value to Rate Adaptive Encoding Algorithm part or Rate Shaper part. We focus on this module in the section 4 in detail.

Client Side:

●***Feedback***: This module monitors the calculated metrics which decide the network and client conditions. Then this module passes the values to the RTP Layer which is responsible for transmitting both the raw video data and the feedback. The details of this module will be described in section 4.1.

●***RTP Layer***: The standard of this module is [6]. It includes data transformation (RTP) and control protocol (RTCP). The RTP data transmission is responsible for transmitting the video data while the RTCP is for the QoS information in Feedback module.

●***Video Decoder***: This module reads the received data and decodes them and sends them to player to play them out.

4. The Rate Control

We use RTP for the data transmission, and RTCP is used for this feedback module. Lots of related works use packet lost rate in receiver report as the important or the only factor for deciding internet condition and use AIMD (Additive Increase Multiple Decrease) rate control as follows:

If (p<=Pth)
R = min{(r+AIR) , MaxR}
Else
R = max{(a*r) , MinR}

Where p is the packet loss rate, Pth is the threshold for the packet loss ratio, r is the sending rate at the source, AIR is the additive increase rate, MaxR and MinR are the maximum and minimum thresholds.

An example of source rate behavior under the AIMD rate control is illustrated in Figure 2. The performance is full of fluctuation and our target of this section is to reduce fluctuations.

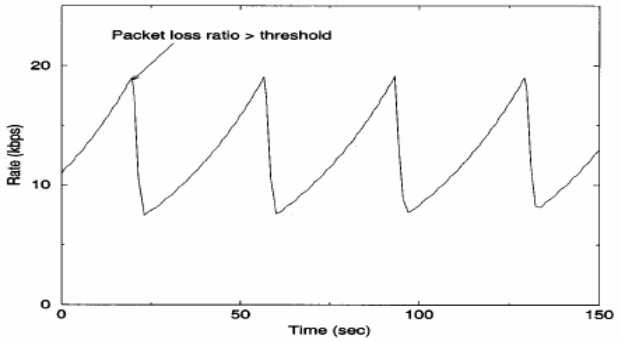

Figure 2 AIMD performance

The architecture of this module is structured in Figure 3 and described in the sections below.

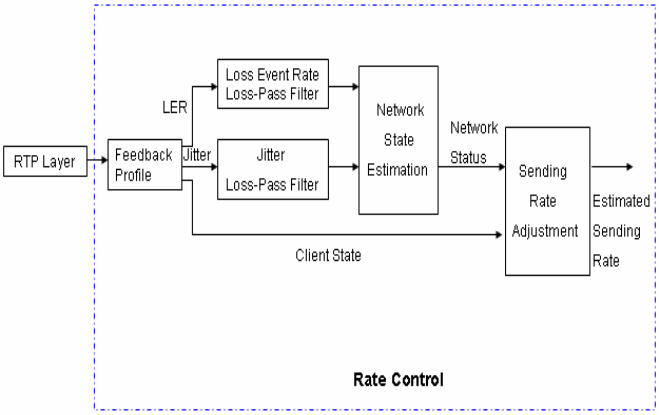

Figure 3 Rate Control

4.1 Feedback Profile

Feedback profile is calculated and added into RTCP profile at the client side and transmitted to the server side. Our feedback profile has three metrics including Loss Event Rate (LER), jitter and Client State. We describe these metrics in detail in this section.

4.1.1 Metric 1 --- Loss Event Rate

4.1.1.1 Loss Event Rate vs Loss Fraction

Here we use LER as a main factor of internet condition instead of packet loss rate in receiver report. The LER is not quite the same as the fraction of packets lost. The loss fraction may not be a sufficient measurement for two reasons: Firstly, TCP responds primarily to loss events, rather than to the actual number of lost packets, with most implementations halving their congestion window only once in response to any number of losses within a single round-trip time. Tahoe, NewReno, and Sack TCP implementations generally halve the congestion window once in response to several losses in a window, while Reno TCP typically reduces the congestion window twice in response to multiple losses in a window of data. A measure of loss events that treats multiple consecutive lost packets within a round-trip time as a single event, rather than counting individual lost packets should compete more equally with TCP. Secondly, the reported loss fraction during any particular interval is not necessarily a reflection of the underlying loss rate, because there can be sudden changes in the loss fraction as a result of unrepresentative bursts of loss. This is a problem because fluctuant behavior can result if a sender uses the loss fraction to compute its sending rate directly. For a very high loss environment and for a low loss environment, there will be little difference between the packet loss rate and the loss rate for a flow. However, for a moderate loss environment, there is some difference between the two. To some extent, such behavior is unavoidable – AIMD algorithms are inherently fluctuant – but the fluctuations should be reduced as much as possible by using LER.

4.1.1.2 Loss Event Rate in RTP/RTCP

RTP/RTCP protocol do not directly measure the loss event rate, but instead count the number of packets lost over each RTCP reporting interval and include that number in the RTCP reception report packets as a loss fraction. Here we calculate LER in the way the same as [7]. The process of calculating LER in the client part happen every interval and the value can be transported using a dedicated field of the RR packets.

Low-pass filter is used to smooth loss event rate as follows:

LERcur = a * LERpre + (1-a) * LERnet

▲LERcur : The current filtered value of loss event rate.
▲LERpre : The previous value of loss event rate.
▲LERnet : The current value loss event rate got from RR
▲a : The parameter to smooth the loss event rate, the value can determine how strong the current value of loss event rate effects. The range is [0,1].

If ECN (Explicit Congestion Notification) [8] is available, we recommend using ECN to calculate loss event rate as a sample of congestion directly. [7] also describes how to calculate loss event rate with ECN bits. Although ECN has no standard on UDP, some researches about rate control are using it as an important input such as [9].

4.1.2 Metric 2 --- Jitter

We also use jitter as an important factor to evaluate network status. Here we use the definition of jitter in RFC 3550[10]. The filter is also used to smooth as follows:

Jcur = b * Jpre + (1-b) * Jnet

▲Jcur : The filtered value of jitter.
▲Jpre : The previous value of jitter.
▲Jnet : The current value of jitter got from RR
▲b: The parameter to smooth the jitter, the value can determine how strong the current value of jitter effects. The range is [0,1].

4.1.3 Metric 3 --- Client State

The client state represents the state of the client, including CPU load, battery lifetime. Because the diversity of the kinds of the clients, the main parameters to determine the client state are different.

When the client is a PC, we consider CPU load:

If ($CPU_{load} < CPU_{th}$)
ClientState = idle
Else
ClientState = busy

▲CPU_{load}: The value of CPU load of the client
▲CPU_{th}: The threshold value of CPU load.

When the client is PDA which means that the network involves wireless network, we consider remaining battery lifetime:

If ($B < B_{th}$)
ClientState = busy
Else
ClientState = idle

▲B: The remaining battery lifetime of the client.
▲B_{th}: The threshold value of remaining battery lifetime of the client.

We choose to send less data to the client when the client is short of battery because some documents such as [12] shows transcoding a video stream can save lots of energy.

The parameters to determine client state depends greatly on the actual environments.

4.2 Network State Estimation

With the LER and Jitter low-pass filters, the network state estimation component characterizes the network on the following conditions:

● ***Condition congestion***: when the LER is high.
● ***Condition load***: when the LER is in affordable which means it might be larger than 0 but does not cause problems to the presentation of the video data,
● ***Condition upload***: when the LER is 0 or is very small.

The network estimation module uses the following algorithm to determine the condition of network:

If (LERcur>=LERcon) → congestion
If (LERun<LERcur<LERcon) → load
If (LERcur<=LERun) → unload

▲LERcon : The parameter to determine whether the value of loss event rate is in the condition of congestion.
▲LERun : The parameter to describe whether the value of loss event rate is in the condition of unload.

As [10] described, the value of delay jitter depends on the transmission path and the cross-traffic in the transmission path, but abrupt increase of delay jitter may denote that the buffers in the queues on the transmission path had been overloaded and this may soon cause congestion to the network. The formula based on jitter is described as follows:

If (Jcur > c * Jpre) → congestion

▲c : The parameter to describe how abrupt the increase of delay jitter can cause the congestion. The range is [0,1].

4.3 Sending Rate Estimation Adjustment -- AIMD

Usually the transmission rate estimation component uses AIMD algorithm in order to estimate the new transmission rate. AIMD is used in congestion avoidance part of TCP rate control and many rate control algorithm of many applications of today's Internet. In order to make less fluctuation, we propose the adjustment of rate estimation as follows:

If (network = unload) → Rnew = Rold + Rinc
If (network = load) → Rnew = Rold
If (network = congestion) → Rnew = Rold * Rdec

▲Rnew : the estimated sending rate value
▲Rold : the last sending rate value
▲Rinc : the increase sending rate value when network is in unload condition
▲Rdec : the decrease sending rate value when network is in congestion condition.

However, client state is an important parameter for determining the estimating sending rate simply as below:

If (ClientState = busy) → Rnew = Rold * Rdec

In other sentences, when the application notices available bandwidth (unload condition) and the client state is not busy, it increases the transmission rate by adding a value to the transmission rate (Rinc), when congestion occurs or the client is busy, it decreases the transmission rate by multiplying the transmission rate with a value (Rdec, less than 1). These parameters depend on the network condition and can be changed when the requirements change.

5. Conclusion and future work

In this paper, we presented a novel algorithm for dynamically adjusting the sending rate of applications to the congestion level observed in the network and the status of client in the case of unicast video transmission. It uses some different metrics compared to other source-based probing algorithms and can have a better performance in reducing fluctuations caused by AIMD algorithm.

In order to simulate our algorithm, we decide to simulate our algorithm based on live555com library [11] which is a free source for standard RTP, RTCP and RTSP. However, liveMedia does not provide any congestion control. We add congestion control mechanism into liveMedia and then test our feedback rate control algorithm with only LER metric. Result shows that in the case of transient period of severe congestion, the sending rate is more stable compared with using loss fraction.

What we concern about in this paper is in the case of unicast, based on this paper, our future work concerns mainly about the adaptive multicast of multimedia data which is about accommodating clients with heterogeneous data reception capabilities.

Reference

[1] Huai-Rong Shao, "User and content-aware object-based video streaming over the Internet". Visual Communications and Image Processing 2000
[2] D.Hoffman G..Fernando "RSVP – Resource Reservation Protocol" RFC 2205 January 1998
[3] Dapeng Wu "On End-to-End" Architecture for Transporting MPEG-4 Video Over the Internet" IEEE transactions on circuits and systems for video technology VOL.10 NO.6 Sep 2000
[4] "RAP: An End-to-end Rate-based Congestion Control Mechanism for Realtime Streams in the Internet" 1998 1998
[5] DAPENG WU, YIWEI THOMAS, "Transporting Real-Time Video over the Internet: Challenges and Approaches" IEEE
[6] H.Schulzrinne,S.Casner,R.Frederick,andV.Jacobson."RT-P: A Transport Protocol for Real Time Application" RFC3550, July 2003
[7] M.Handley S.Floyd RFC 3448 "TCP Friendly Rate Control (TFRC) "
[8] K.Ramakrishnan and S.Floyd. "A proposal to add Explicit Congestion Notification (ECN) to IP" RFC 2481, January 1999
[9] Pavlos Antoniou "Adaptive Feedback Algorithm for Internet Video Streaming based on Fuzzy Rate Control" IEEE Symposium on Computers and Communications 2007
[10] Ch.Bouras A.Gkamas "Video transmission with adaptive QoS based on real-time protocols" International Journal Of Communication Systems 2003
[11] http://www.live555.com
[12] Qussam Layaida "Reconfiguration-based QoS Management in Multimedia Streaming Applications" IEEE 2004