# 國 立 交 通 大 學

# 電信工程學系


## 碩 士 論 文

# 在以封包為基底的傳輸系統下<br>對影像串流之平順演算法

# Smoothing Algorithm for Video Streaming in Packet Based Transmission System


研究生　：　林旃偉

指導教授　：　張文鐘　博士



中　華　民　國　九十四　年　八　月

# 在以封包為單位的傳輸系統下
# 對影像串流之平順演算法
# Smoothing Algorithm for Video Streaming in Packet Based Transmission System

研 究 生：林旃偉　　Student：Chan-Wei Lin
指導教授：張文鐘　　Advisor：Wen-Thong Chang

國 立 交 通 大 學
電 信 工 程 系
碩 士 論 文

A Thesis
Submitted to Department of Communication Engineering
College of Electrical Engineering and Computer Science
National Chiao Tung University
in partial Fulfillment of the Requirements
for the Degree of
Master
in

Communication Engineering

August 2005

Hsinchu, Taiwan, Republic of China

中華民國九十四年八月

# 在以封包為單位的傳輸系統下
# 對影像串流之平順演算法

研究生: 林旃偉　指導教授: 張文鐘 博士
國立交通大學電信工程學系碩士班

## 中文摘要

電腦網路技術的提升，使得整合性應用的服務增加。影像串流被認為未來網路影用的主要部份。網路必須分送不同的影像資料以滿足不同的應用，例如，包含聲音的即時影像要求、多媒體短片與電影。這類本身含有大量突衝以及速率變動的資料流，加重網路的負載，同時降低了網路的使用效率，也使得網路的管理更加困難。因此，為克服這些問題，發展了能降低資料流突衝的平順演算法。然而，傳統的平順演算法，並無法與影像壓縮標準完美地結合。在此我們提出了兩種以封包為基底的平順演算法。這兩個以封包為基底的平順演算法，能夠近似傳統平順演算法，同時能在不對影像壓縮標準作修正的情形下被實現。在這篇論文中，我們介紹了我們的以封包為基底的平順演算法，展示他的模擬結果以及演算法的效能。

# Smoothing Algorithm for Video Streaming in Packet Based Transmission System

Student: Chan-Wei Lin
Advisor: Dr. Wen-Thong Chang
The Department of Communication Engineering
National Chiao Tung University

## Abstract

The improvement of the computer network technologies result in the increasing of application of integrated services. The video stream is expected to be the major portion of the network application. Network has to deliver various numbers of video data traffic for several applications such as Video on Demand, which contains audio, multimedia clip and movies. This kind of traffic, which has much burst and variable rate inherently, increases the load of network service, decreases the efficiency of network utilization and also makes the network management difficult. Therefore, to overcome these difficulties, the smoothing algorithms, which reduce the burst of traffic, are developed. However, the conventional smoothing algorithm cannot combine with the video compression standards perfectly. We proposed two packet-based smoothing algorithms. These two algorithms approximate to the conventional smoothing algorithm, and can be implemented without modifying the video compression algorithm. We introduce our packet-based smoothing algorithms, show the simulation result and also demonstrate the performance in this thesis.

# Acknowledgement

三年的碩士班生涯，即將在這裡告一個段落。從一開始對未來的迷惘，到認清事實，然後逐步走上軌道，這段過程都將成為我人生中很重要的一個階段吧。

必須感謝我的父母，在這段時間完全放手的讓我去尋找自己的道路，也因為有你們的支持，我才能無後顧之憂。感謝指導教授張文鐘教授，口試委員范國清教授、余孝先主任和李安謙教授，你們的指導，讓我的論文更趨完整。實驗室的夥伴們、學長姐們，從你們那裡學到了許多，也得到了很多的照顧，不成熟的我，如果能在未來有些許的成長，是多虧了與你們的相處。

高中時期的好朋友們，在我對將來感到迷惘的這個時期、在碩士生涯的無數個低潮，有你們的陪伴和鼓勵，那些接近無厘頭的相處和對談，讓我能完全的放鬆，用更開闊的心胸面對生活上的挑戰。傅思維、豬、狗、五百、井長瑞，很感謝你們。最後這一兩個月，大概是這三年來最難熬的時候，修改論文上遇到的種種困難，讓自己很想在最後關頭放棄，之所以能重新站起來走到最後，猛男、糸花、蕃薯，多虧了有你們的鼓勵。

這段過程，看到了許許多多的人生百態，有無數需要感謝的人，不論

是好是壞，所有帶給我人生歷練的一切，都感謝你們。感謝上天，讓我在學生生涯的最後，遇見了蕙君，讓一切的努力都逐漸出現了目標，也讓未來變的更加清晰。

# Content

# List of Figures

# 1. Introduction

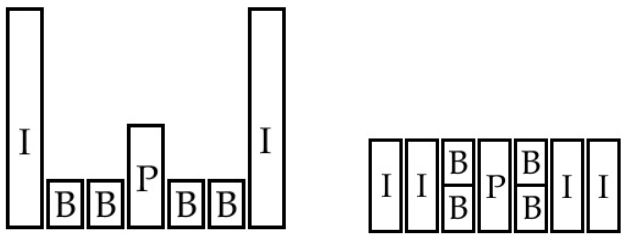
Due to the increasing of network bandwidth, multimedia applications, such as video casting, video-based entertainment, and video on demand, become more and more. These applications consume a substantial amount of network bandwidth. Although Video which compressed in constant bit rate (CBR) is easier to network service, the picture quality will vary with the content of video. If the complexity of the picture content or the motion of video is higher, the picture quality will reduce significantly. On the contrary, if the picture quality is maintained, the rate will vary with the content. Also, the compression technique which introduces the method, that compress I, P, B, three different frame types in different way, result in data burst and rate variation. This kind of variable bit rate (VBR) data waste the network resources and lower the bandwidth utilization efficiency, as it is transmitted through network directly. For overcoming this difficulty, smoothing algorithms are developed. In order to access network bandwidth efficiently and manage easily, the transmission of streams data tends to be transmitted in CBR. Thus the VBR data streams have to be modified into low burst and rate variation. To transmit VBR stream through network in CBR is the main function of smoothing algorithm.

In order to access network bandwidth efficiently and manage easily, the transmission of streams data tends to be transmitted in CBR. To transmit VBR stream through network in CBR is the main function of smoothing algorithm. Many smoothing algorithms are proposed to reduce the variation in transmission rate. But these algorithms can't combine with the video compression standards. Video streaming transmission is restricted to be transmitted packet by packet. The packet cutting mechanism is simultaneous with the video compression, and these packets cut prior to smoothing algorithm can't match the optimum transmission schedule generated by smoothing algorithm perfectly. If the video has to be transmitted according to the transmission schedule, the packets have to be segmented again by server, and be reassembled by client. The segmenting and reassembling mechanisms are not defined in video compression standards. It will limit the application of smoothing algorithm. We proposed two packet-based smoothing algorithms to approximate performance of the conventional smoothing algorithm with the method to control the transmitted number of packets at each frame slot.

In the following section, one buffer constrained smoothing algorithm, minimum variance bandwidth allocation (MVBA)[2][3][4][5], and one rate constrained smoothing algorithm, rate constrained bandwidth smoothing (RCBS) [7][14], is introduced and its performance is shown in section 2 and 3. Our packet based smoothing is related in section 4, and section 5 includes the simulation result and performance of our algorithm. Finally, a conclusion is presented in the last section.

## 2. Conventional Smooth Algorithm

Now, we describe the framework of video streaming. First step of the procedure of video stream transmission is that client sends a demand to the server, and the connection is established. Server accesses the video data from the storage device, and then sends the video data through the network to the client. The client will store up this data in the buffer, and extract them to send to the decoder according to the playback schedule. The traditional method is that server sends each frame data at each corresponding frame slot. A streaming server with smoothing algorithm can transmit video data at a more stable rate. Instead of transmitting each frame data at each corresponding frame slot, the server will reschedule the transmission data at each frame slot. The large frame, such as I frame, may be cut into several portions and transmitted ahead at the previous frame slot, and the small frame, such as P or B frame, may be transmitted together at one frame slot. For instance, an I frame may be cut into several part and be transmitted at the previous P or B frame slots. See the following Fig. 1 as an example. The large I frame data is cut into two segments, and one of them is transmitted at the B frame slot, and two B frames are transmitted at a frame slot.




(a) (b)

Figure 1: An example of transmission schedule. (a) is the transmission schedule before smoothing. (b) is the schedule after smoothing.

Because of the real time property of video streaming, transmission schedule is under the following restrictions. If the transmission rate is too low, the client buffer will be underflowing, and it will induce the playback delay. On the contrary, if the transmission rate is too high, the client buffer will overflow, and it will induce the loss of data and frame drop. Therefore, although the ultimate goal of smoothing algorithm is to find a CBR transmission schedule, but CBR schedule is usually infeasible because it may disobey the upper bound and lower bound. The rate of a feasible transmission schedule can't be too low or too great in order to play fluently at client. Thus a feasible transmission schedule has to obey some constraints, these will describe in the following sections.

## 2.1 Buffer Constrained Smoothing Algorithm

First, we discuss the buffer constrained smoothing algorithm. This kind of algorithm is to find a feasible schedule with a given fixed client buffer.

### 2.1.1 Overflow and Underflow Constrain

At first, we describe the restriction of the transmission schedule in buffer constrained smoothing algorithm. Because the client buffer is fixed, the transmitted data can't be too many to store up. However, the client has to receive enough data to play fluently, thus the transmitted data can't be too less. Therefore, the total transmitted data have to obey some upper bound and lower bound. The following is the mathematic representation of the upper bound and lower bound.

At first, we describe the notation of the parameter of a video. A compressed video stream consists of N frames, and the amount of data of frame i is $a_i$ bits (or bytes) to store. Without loss of generality, we assume that time is measured in unit of frame slots. In order to playback streaming video fluently at client terminal, the server always has to send enough data to the client to prevent buffer underflowing. Thus, the lower bound of transmission schedule,

$$L(k) = \sum_{i=0}^{k} a_i , \qquad \text{for all } 0<k<N \qquad (2\text{-}1)$$

The sum of $a_i$ for the duration of 0 to k represents the amount of data consumed at the client during k frame slots. The server has to transmitted at least L(k) to client in order to confirm that the client always has enough data to play.

Also, a client cannot receive too many data to store. The remainder data, which subtract the consumed data from the total transmitted data, can't be greater than the client buffer size to prevent buffer overflowing. Thus the upper bound of transmission schedule,

$$U(k) = L(k) + b , \qquad (2\text{-}2)$$

where $b$ is the client buffer size. The sum of L(k) and b represents that the total amount of data which is transmitted by server cannot exceed the total consumed data at client plus buffer size to prevent buffer overflow. The total transmitted data can't be above U(k), or the remainder data after the client consume the video data will be above b, and then the client will be overflowing.

Therefore, a feasible transmission schedule must stay within the area, which is enclosed by the upper bound and lower bound.

That is,

$$L(k) \le \sum_{i=0}^{k} s_i \le U(k), \qquad \text{for all } 0<k<N \qquad (2\text{-}3)$$

where $s_i$ is the transmission rate of the schedule of the smoothed video stream at frame slot i.

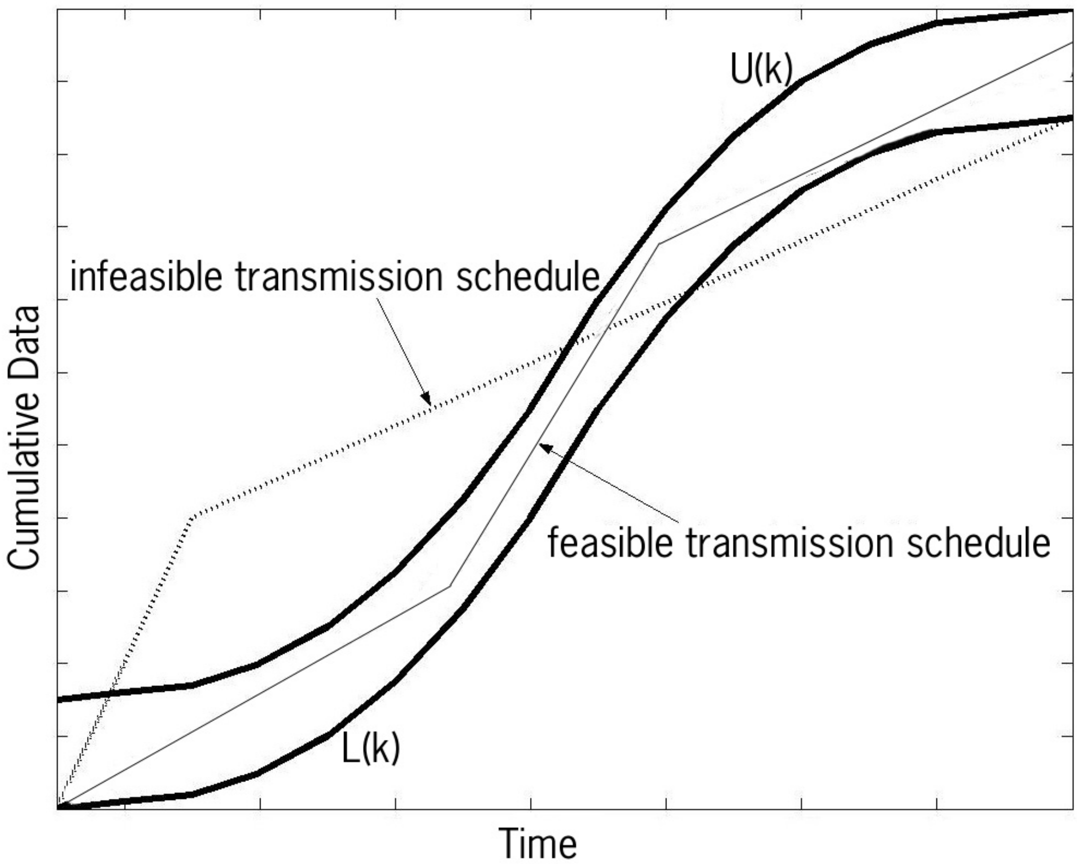



Figure.2: An example of feasible and infeasible transmission schedules.

A transmission schedule consists of several linear segments. Each segment of a feasible transmission schedule has to stay inside upper bound and lower bound, otherwise, the schedule is infeasible, and the slope of each segment indicates the transmission rate during each corresponding period. Fig.2 is an illustration of feasible and infeasible transmission schedule. Where the two bold solid lines represent the upper and lower bound, the fine solid line is a feasible transmission schedule, and dot line is an infeasible transmission schedule which result in client buffer overflowing at the prior segment and buffer underflowing at the later segment.

### 2.1.2 Minimum Variability Bandwidth Allocation (MVBA)

In this paper, we introduce one of the buffer constrained smoothing algorithms, MVBA. The criterion of this algorithm is to minimize the variance in transmission rate. The transmission schedule resulted by MVBA has the following properties.

1. The rate variation is minimized.

2. The peak rate is minimized.

3. The lowest rate is maximized.

MVBA will find a succession of feasible CBR segments. These segments connect with each other at the end points. Then the transmission schedule generated by MVBA consists of these CBR segments, and the slopes of each segments represents the constant transmission rate in the corresponding interval. MVBA will extend the interval of CBR segment and check if the segment will become infeasible or not. If MVBA find the CBR segment will result in buffer overflowing or underflowing, it will change the rate slightly to find the successive feasible CBR segment as early as possible. The new segment will start at the time that the client buffer is just overflowing or just underflowing when the process find that the CBR segment will be infeasible in the future, and the process will repeat until whole video is be segmented. The variance of rate of the transmission schedule generated by MVBA will be minimized. The proof of this property is described in [2].

**The Parameters of MVBA**

At first, we define the parameters of the MVBA. Let $s_{\max_{a,b}}$ segment represents the maximum transmission rate without making client buffer full in the interval [a,b]. The CBR segment starts with initial buffer level q, which is the remainder data stored in client buffer at $a_{th}$ frame slot when the process starts a new run from $a_{th}$ frame slot. That is

$$s_{\max_{a,b}} = \min_{a+1 \le t \le b} \frac{U(t)-(L(a)+q)}{t-a}, \qquad (2\text{-}4)$$

and $t_{U_{a,b}}$ is the latest frame slot at which client buffer is full when server transmit at the rate of $s_{\max_{a,b}}$ in interval [a,b]. That is

$$t_{U_{a,b}} = \max_{a+1 \le t \le b} \left\{ t : \frac{U(t)-(L(a)+q)}{t-a} = s_{\max} \right\}. \qquad (2\text{-}5)$$

For each interval [a,b], there exist only one $s_{\max_{a,b}}$ segment and one corresponding $t_{U_{a,b}}$. See Fig. 3 as an example. The bold line is the upper bound and lower bound of transmission schedule. $s_{\max_{a,b}}$ segment starts at $a_{th}$ frame slot with initial buffer level q. At the interval [a,a+4], [a,a+5], [a,a+6], [a,b], the $s_{\max_{a,b}}$ is invariant, and the corresponding $t_{U_{a,b}}$ is also the same.

Similarly, let $s_{\min_{a,b}}$ segment represents the minimum transmission rate without making client buffer starve in the interval [a,b], starting with initial buffer level q:

$$s_{\min_{a,b}} = \max_{a+1 \le t \le b} \frac{L(t) - (L(a) + q)}{t - a}, \qquad \text{(2-6)}$$

and $t_{L_{a,b}}$ is the latest frame slot at which client buffer is starve when server transmit in rate $s_{\min_{a,b}}$ in interval [a,b]. That is

$$t_{L_{a,b}} = \max_{a+1 \le t \le b} \left\{ t : \frac{L(t) - (L(a) + q)}{t - a} = s_{\min} \right\}. \qquad \text{(2-7)}$$

Similarly, see the $s_{\min_{a,b}}$ segment in Fig. 3 as an example. This segment start at $a_{th}$ frame slot with initial buffer level q, and the client buffer will starve at $t_{L_{a,b}}$.

There exist some feasible CBR transmission schedule in interval [a,b], if and only if $s_{\max} \ge s_{\min}$. See Fig. 4 as an example.

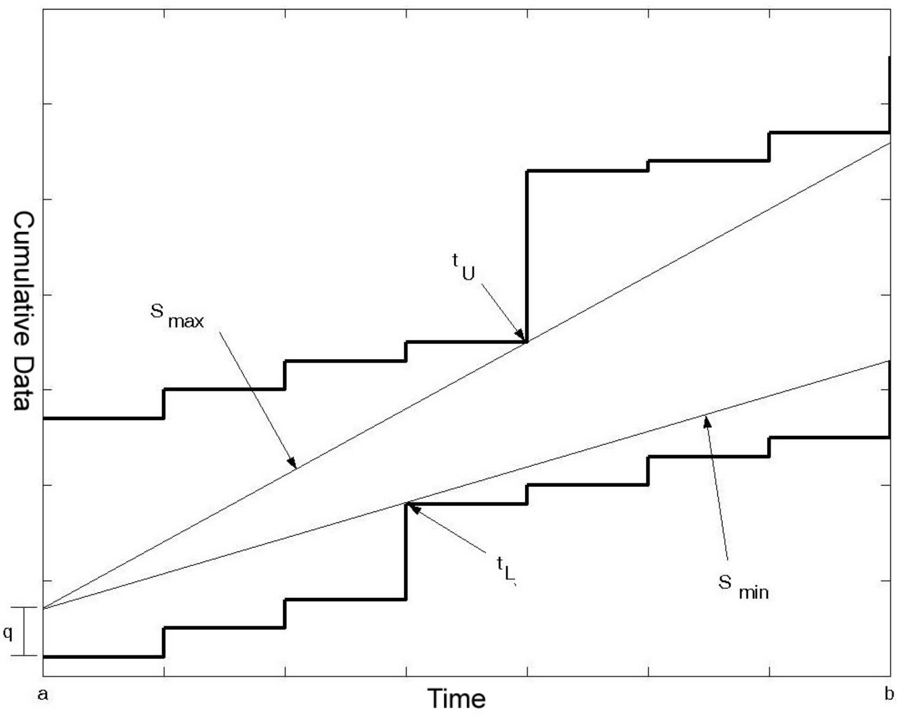



Figure 3: An example of $s_{\min_{a,b}}$ , $s_{\max_{a,b}}$ , and corresponding $t_{L_{a,b}}$ , $t_{U_{a,b}}$ .

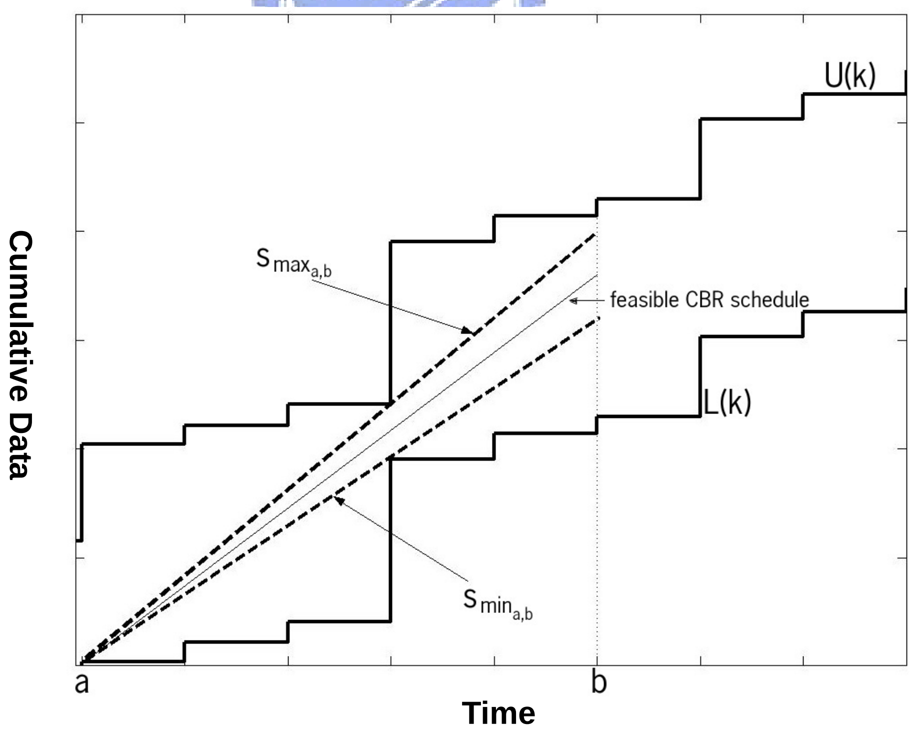



Figure 4: An example of a condition in which the feasible CBR transmission schedules exist.

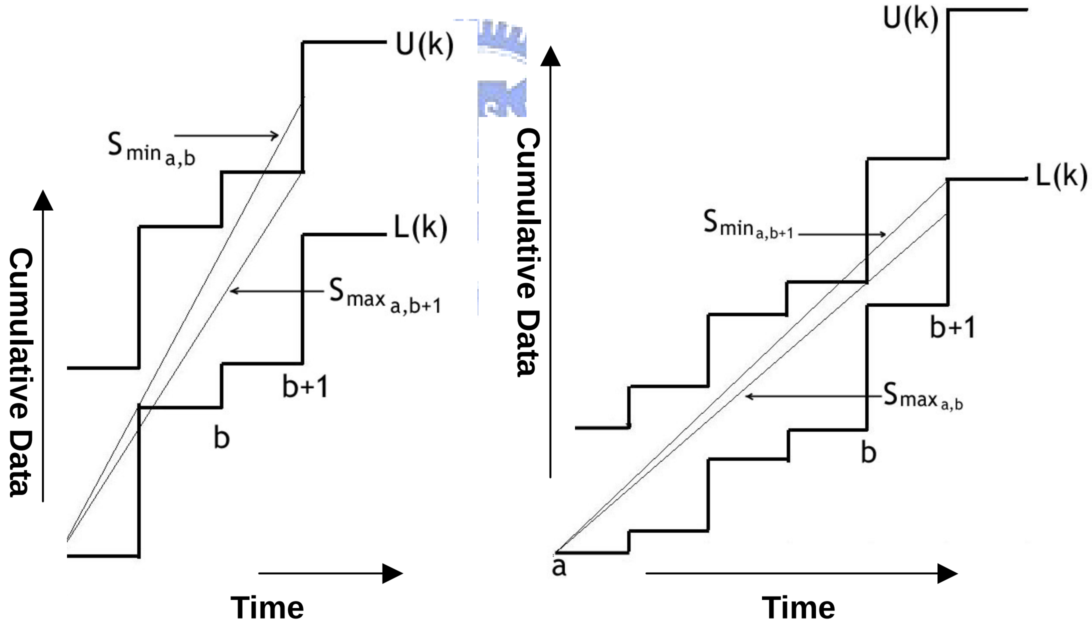
### The procedure of MVBA

The process starts in the interval [1,2], where a equals to 1 and b equals to 2, and calculates $s_{\min_{a,b}}$, $s_{\max_{a,b}}$, $t_{L_{a,b}}$, $t_{U_{a,b}}$ and $s_{\min_{a,b+1}}$, $s_{\max_{a,b+1}}$. Then the process will check if the following two conditions occur.

1) If the computed $s_{\max_{a,b+1}}$ at [a,b+1] is smaller than $s_{\min_{a,b}}$, it's means that if server transmits data in rate $s_{\min_{a,b}}$ at [a,b+1], the client buffer will overflow at b+$1_{th}$ frame slot. See Fig. 5(a) as an example. The slope of $s_{\min_{a,b}}$ is greater than $s_{\max_{a,b+1}}$, thus $s_{\min_{a,b}}$ will reach the upper bound at b+$1_{th}$ frame slot. This phenomenon represents that the client buffer is overflowing at b+$1_{th}$ frame slot. Thus the transmission rate must slow down before b+1th frame slot, if the server transmits in rate $s_{\min_{a,b}}$ from ath frame slot. MVBA algorithm determines to transmit in rate $s_{\min_{a,b}}$ at $[a, t_{L_{a,b}}]$, and the process starts a new computation procedure from $t_{L_{a,b}}$, then a is set to be $t_{L_{a,b}}$ and b is set to be $t_{L_{a,b}} + 1$. Then the initial buffer level q is set to 0.

2) If the computed $s_{\min_{a,b+1}}$ at [a,b+1] is greater than $s_{\max_{a,b}}$, it's means that if server transmits data in rate $s_{\max_{a,b}}$ at [a,b+1], the client buffer will underflow at b+$1_{th}$ frame slot. See Fig. 5(b) as an example. The slope of

$s_{\max_{a,b}}$ is smaller than $s_{\min_{a,b+1}}$, thus $s_{\max_{a,b}}$ will reach the lower bound at b+1$_{th}$ frame slot. This phenomenon represents that the client buffer is underflowing at b+1$_{th}$ frame slot. Thus the transmission rate must speed up before b+1$_{th}$ frame slot, if the server transmits in rate $s_{\max_{a,b}}$ from a$_{th}$ frame slot. MVBA algorithm determines to transmit in rate $s_{\max_{a,b}}$ at $[a, t_{U_{a,b}}]$, and the process starts a new computation procedure from $t_{U_{a,b}}$, then a is set to be $t_{U_{a,b}}$ and b is set to be $t_{L_{a,b}}+1$. Then the initial buffer level q is set to b, the client buffer size.




(a) (b)

Figure 5: (a) represents that when $s_{\min_{a,b}} \geq s_{\max_{a,b+1}}$, client buffer will overflow at b+1$_{th}$ frame slot. (b) represents that when $s_{\max_{a,b}} \leq s_{\min_{a,b+1}}$, client buffer will underflow at b+1$_{th}$ frame slot.

Otherwise there exists a feasible CBR segment over [a, b+1], then the extension of the interval continues, b is updated to b+1, and $s_{max}$, $s_{min}$ and $t_U$, $t_L$ keep on being updated. The process continues until the entire video is segmented. As b reaches to the end of the video, then b equals to N, and the last segment is just the mean rate in interval [a, N], which equals to $\frac{L(N)-L(a)+q}{N-a}$. The generated transmission schedule is identified as a series of CBR transmission segments, each ending with a change in transmission rate.

## 2.2 Rate Constrained Smoothing Algorithm

Now, we consider the rate constrained smoothing algorithm. In this case, the max transmission rate of the network link between server and client is constrained to be under r. Thus, the client buffer have to be large enough to store the data which is transmitted ahead to prevent the playback discontinuously, and the playback startup delay has to be long enough to confirm that the server can transmitted enough data ahead for the first peak of the video data with a rate that is lower than r. See Fig. 6 as an example of a feasible rate constrained transmission schedule. The minimum buffer requirement is the greatest distance between the transmission schedule and the consumed data, L(k). The server has to transmit enough data before the client starts to play the video to prevent the total transmitted data may be below L(k) in the future, and the rate for this duration has to be also lower than r. Thus the startup delay has to be long enough to confirm that the server can transmitted enough data at a rate which is lower than r before the client starts to play the video.

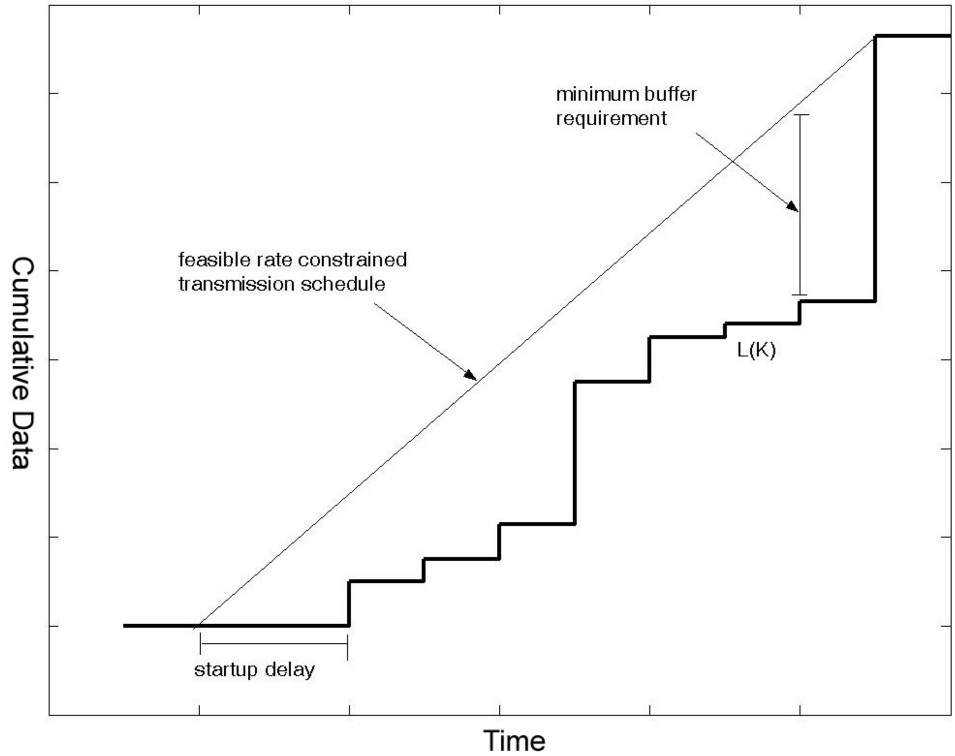



Figure 6: An example of feasible rate constrained transmission schedule. The slopes of each segment have to be under r, and the cumulative transmission data have to be greater than the L(k). There exist on minimum buffer requirement and one minimum startup delay for each feasible transmission schedule, and the two parameters will be determined after the schedule is generated.

Under this constraint, a feasible transmission schedule must satisfy the following two terms:

$$\sum_{i=1}^{k} s_i \geq L(k) \text{, and} \qquad (2\text{-}8)$$

$$s_i \leq r \text{, for all i .} \qquad (2\text{-}9)$$

### 2.2.1 Rate Constrained Bandwidth Smoothing (RCBS)

RCBS, a rate constrained smoothing algorithm, is to find a transmission schedule, which transmits data as late as possible to reduce the requirement of client buffer, and also obey the rate constraint r simultaneously. After a feasible transmission schedule is computed, a corresponding group of minimum startup delay and minimum client buffer requirement is determined.

The procedure of RCBS is simple. The concept of this algorithm is that when the amount of data which must be transmitted at $i_{th}$ frame slot is greater than r, the server can only transmits in rate of r at $i_{th}$ frame slot, and the remainder data that can't be transmitted at this frame slot have to be transmitted at the earlier frame slot for preventing client buffer underflowing. Therefore, the amount of data, which has to be transmitted at $i\text{-}1_{th}$ frame slot, becomes the data of $i\text{-}1_{th}$ frame plus the remainder data of $i_{th}$ frame slot. The transmission schedule is as following:

$$s_i = \begin{cases} r & \text{if} \quad a_i + rem_i > r \\ a_i + rem_i & \text{if} \quad a_i + rem_i < r \end{cases}. \tag{2-10}$$

where $a_i$ is the amount of data of $i_{th}$ frame, and $rem_i$ represents remainder data that have to be transmitted at $i+1_{th}$ frame slot originally, but the server cannot transmit these data at $i+1_{th}$ frame slot because of the rate constraint. Therefore $a_i + rem_i$ represents the data have to be transmitted at $i_{th}$ frame slot. If $a_i + rem_i$ is greater than r, it means that the data, which have to be transmitted before $i_{th}$ frame slot, is great than the rate constraint, thus the server can only transmits in rate of r at this frame slot and the remainder data, which can't be transmitted at this frame slot, will remain to be transmitted at previous frame slot. The amount of this remainder data, $rem_{i-1}$, equals to $a_i + rem_i - s_i$.

**The Procedure of RCBS**

The initial setting of RCBS is that i equals to N, the latest frame number, and the initial $rem_i$ equals to 0, it also means that $rem_N$ equals to 0. Then, according to equation 2-10, $s_N$ equals to $a_N$ and $rem_{N-1}$ equals to 0 if $a_N + rem_N$ is smaller than r. On the other hand, if $a_N + rem_N$ is greater than r, $s_N$ will be set to r and $rem_{N-1}$ will be set to $a_N + rem_N - r$. Then, the next step is to check if $a_{N-1} + rem_{N-1}$ is greater than r, and determine $s_{N-1}$ according to equation 2-10. This process will repeat until i reaches 0. Then the total transmission schedule is generated, and the minimum buffer requirement and the minimum start up delay will be determined.

Fig. 7 illustrates this procedure, where the bold line represents L(k), and the fine line is the CBR segment of transmission schedule, of which the slope equals to r. The remainder data at $i_{th}$ frame slot equals to 0, therefore the data have to be transmitted at $i_{th}$ frame slot equals to the data of $i_{th}$ frame. Because the amount of data of $i_{th}$ frame is greater than the constrained rate, the server can only transmit in rate of r at $i_{th}$ frame slot, and the data, which can't be transmitted at $i_{th}$ frame slot, have to be transmitted at previous frame slot. These remainder data and the data of $i\text{-}1_{th}$ frame have to be transmitted together at $i\text{-}1_{th}$ frame slot. Then the process checks if the sum of the remainder data and the data of $i\text{-}1_{th}$ frame is greater than r. If the sum is smaller than r, the server can transmitted all of these data at this frame slot. Otherwise, like Fig. 7, the server can only transmitted in rate of r at $i\text{-}1_{th}$ frame slot, and the remainder data will be transmitted at $i\text{-}2_{th}$ frame slot with the data of $i\text{-}2_{th}$ frame slot, and repeat the same check process. This process will continuous until the whole video frame is checked.

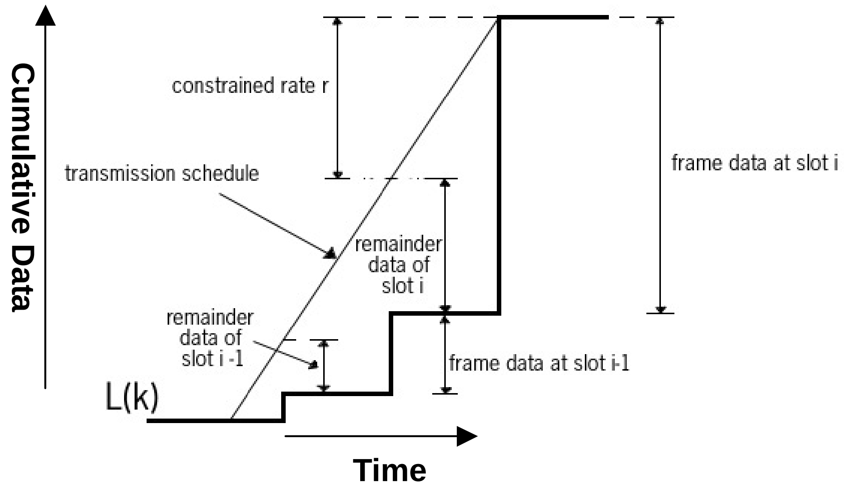



Figure 7: An example of RCBS procedure. The data exceeds the constrained rate is transmitted at the prior frame slot.

After the transmission schedule is determined, the maximum difference between transmission schedule and L(k) is the minimum required client buffer requirement. That is

$$b = \max_{1 \le t \le N} \left\{ \sum_{i=1}^{t} s_i - L(t) \right\}. \tag{2-11}$$

And the elongate part in front of the first frame slot is the startup delay, that is

$$w_t = \frac{rem_0}{r}, \tag{2-12}$$

where, $w_t$ means the start up delay, and $rem_0$ is the data which have to be transmitted before $0_{th}$ frame slot. The minimum startup delay equals to the data, which have to be transmitted before the video stars to be play, divide by the constrained rate, r.

# 3. Simulation and Performance Evaluation of MVBA

In this chapter, we demonstrate some examples of MVBA, and analyze and compare their performance. Also, we measure some important factors for video streaming service, peak rate, rate variation, startup delay and corresponding client buffer requirement.

The peak rate and the variability of transmission rate affect the network efficiency significantly. The lower these factors are, the higher the network efficiency is. The startup delay and buffer requirement can be considered as the criteria of admission control, for instance, a client informs the server of the maximum available buffer and the maximum waiting time, then the server execute these smoothing procedures and decides whether the demand is admitted or not according to these factor.

## 3.1 Simulation Setting

We experiment on several video trace with MVBA smoothing algorithm. These video trace is from [1], in QCIF format, 30 frames per second, and compressed with MPEG4. The total length is an hour, 108,000 frames. First we use the video trace of Disney animation "Aladdin" for testing. This video trace is compressed with I-P-B quantization level 10-14-16. The compression ratio is 43.4793. The total amount of frame data is 94,429,534 bytes.

In general application, clients permit a certain startup delay. In our experimentation, the startup delay is 30 seconds, 900 frame slots to lower the transmission rate of first run.

Finally, we experiment on three different video traces, “Jurassic Park One”, “Terminator One” and “Baseball Game”. They are all in QCIF format, compressed with MPEG4 with I,P,B quantization level 10-14-16.

## 3.2 Performance of MVBA

The following Fig. 8 is an illustration of the transmission schedules generated by MVBA with different client buffer size. We perform MVBA with 100K, 300K, 1M, 10M, 30M bits client buffer respectively. As the client buffer increase, the variation of transmission schedule and the peak rate lower, and the schedule is more like CBR.

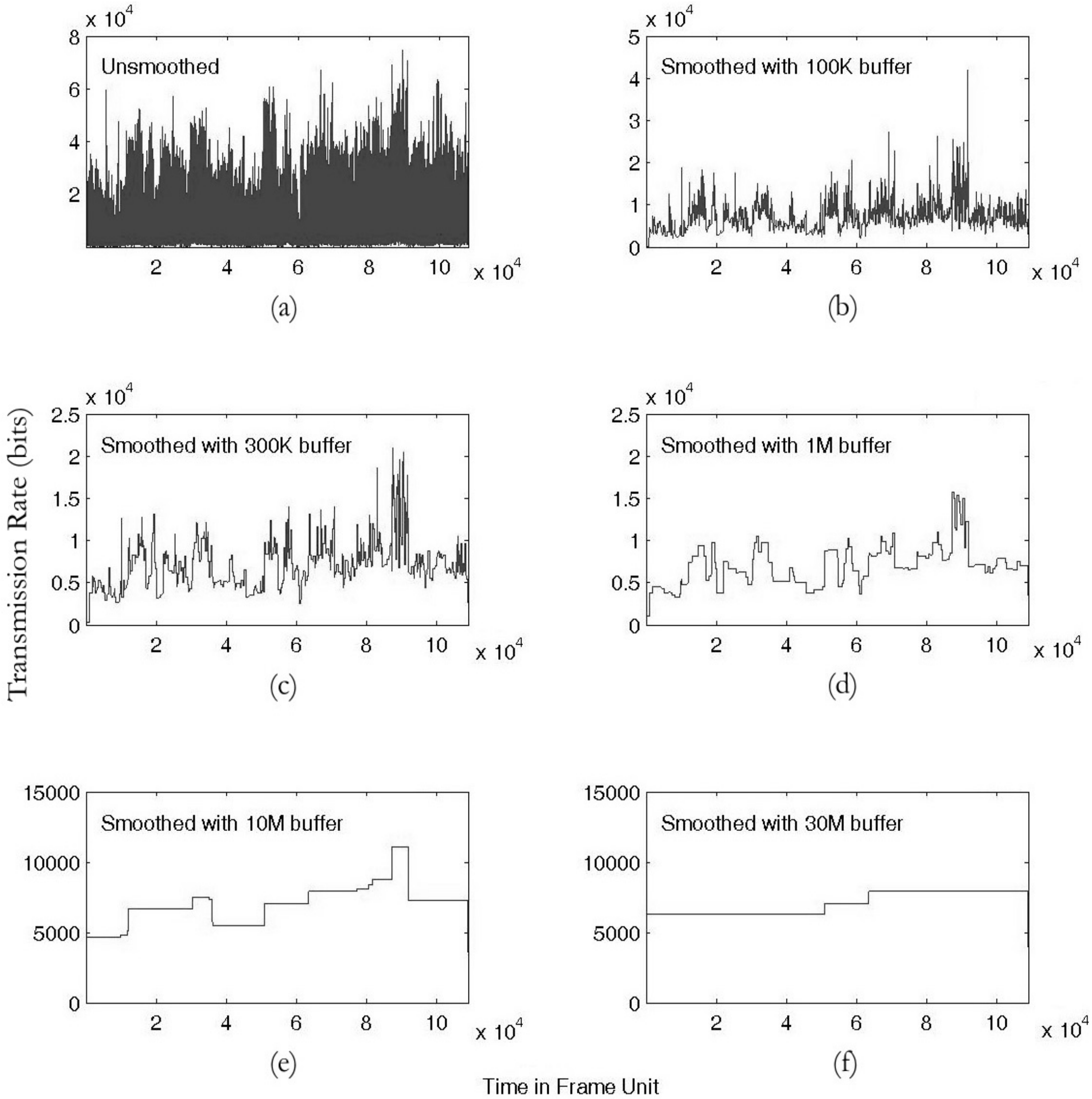



Figure 8: The transmission schedule yielded by MVBA with different client buffer. (a) is the original video trace. (b) (c) (d) (e) (f) is yielded by MVBA with 100K, 300K, 1M, 10M, 30M client buffer respectively.

Fig. 8(a) is the original video trace. The variation is extremely large, and the peak appear at about 90,000th frame slot. The amount of frame data is 9,345 bytes.

Fig. 8(b) is the transmission schedule after MVBA smoothing with 100K client buffer size. 100K is about the 1.5 times as large as the peak rate. Fig. 9 (a) is the transmission rate from the 91000th frame slot to 92000th frame slot generated by MVBA with 100K bits client buffer. The peak rate is at about 91650th frame slot. Fig.9 (b) is the result generated in MVBA process. Where the two fine stair lines represent the transmission upper bound and lower bound and the bold line segments are the transmission schedule. At about 91650th frame slot, there is a succession of large data frame. It induces the rate increasing rapidly and results in the peak rate of transmission schedule.

Note that the peak rate of the transmission schedule is still at about 90,000th frame slot, but the amount is a little bit lower, and variation is reduced significantly. This phenomenon is more apparent as the client buffer increase. See Fig. 8 (c) and Fig. 8 (d) as examples, which are transmission schedules after MVBA smoothing with 300K and 1M client buffer respectively.

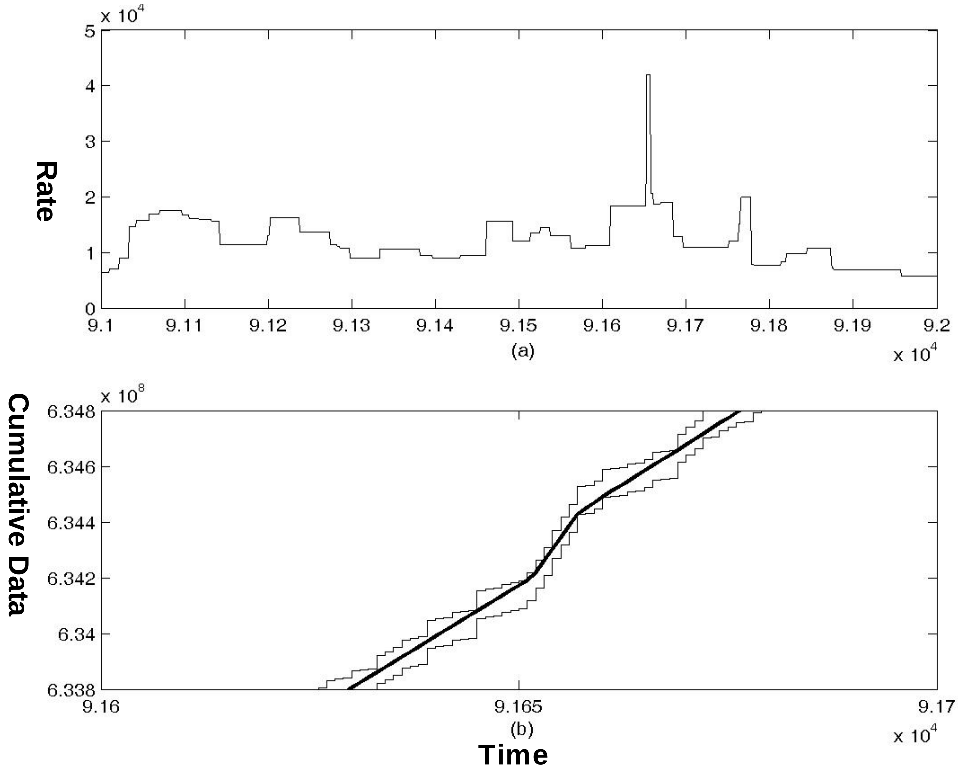



Figure 9: Peak rate of transmission schedule generated by MVBA with 100K bits client buffer size. (a) is the transmission rate. (b) is the transmission schedule and Upper and lower bound.

When the client buffer increases to 10M, the rate changes for less times but the peak rate appear at about 90,000th frame slot, and when the client buffer increases to 30M, it changes for only two times, and the transmission schedule is almost CBR.

The statistical data including maximum rate, minimum rate, average rate, mean to maximum ratio, and variance is shown in the following Table.1.

Tabel.1: The statistical data of transmission schedule of “Aladdin” smoothed with different client buffer.

| | Original Video Trace | Smoothed with 100K buffer | Smoothed with 300K buffer |
|---|---|---|---|
| Mean Rate (bits/frame) | 6,994.8 | 6,994.8 | 6,994.8 |
| Minimum Rate (bits/frame) | 72 | 111.0 | 333.0 |
| Maximum Rate (bits/frame) | 74,760 | 42,030 | 20,964 |
| Mean/Peak (BFF) | 0.094 | 0.166 | 0.334 |
| Variance ($bits^2$/frame) | 72,480,467 | 9,788,187 | 7,325,169 |
| Coefficient of Variance | 1.217 | 0.447 | 0.387 |
| | Smoothed with 1M buffer | Smoothed with 100K buffer | Smoothed with 300K buffer |
| Mean Rate (bits/frame) | 6,994.8 | 6,994.8 | 6,994.8 |
| Minimum Rate (bits/frame) | 1,109.9 | 4,651.8 | 6,336.4 |
| Maximum Rate (bits/frame) | 15,675 | 11,076 | 7,466.4 |
| Mean/Peak (BFF) | 0.446 | 0.632 | 0.880 |
| Variance ($bits^2$/frame) | 5,317,863 | 1,974,558 | 564,841 |
| Coefficient of Variance | 0.330 | 0.201 | 0.107 |

In table 1, the mean rate is defined as $\frac{total\ video\ data}{total\ number\ of\ frames}$, and the maximum rate and minimum rate are the maximum and minimum values of the generated transmission schedule. The variance is defined as $\frac{\sum_{i=1}^{N}(s_i - mean)^2}{N}$, and the coefficient of variance is defined as $\sqrt{\text{variance}}/\text{mean}$, which represents the relation between the standard deviation, which equals to the square root of variance, and mean rate. The coefficient of variance represents the rate variation to mean rate ratio. According to the statistical data of table.1, we can find that the smoothing algorithm won't affect the average data rate. But the minimum rate will increase, and the maximum rate, variation, and coefficient of variation will decrease as the client buffer increase. The coefficient of variance of the unsmoothed transmission schedule equals to 1.217, it means that the mean variation of transmission rate is about 1.2 times of mean rate. When the client buffer is 30Mbits, the value equals 0.107, it means that the variation of transmission rate is only about 10% of the mean rate, and the transmission rate is vary close to CBR.

The mean to peak ratio is so-called bandwidth fill factor (BFF), which represents the efficiency of CBR channel usage. In CBR transmission, the requirement of bandwidth is the same as the peak rate of the data stream. Thus, the total data could be transmitted in this channel is $Peak\_rate \times N$, where N is the total frame number, and actually, the total transmitted data is $\sum_{i=1}^{N} a_i$. Then, the efficiency of usage of this channel is

$$BFF = \frac{\sum_{i=1}^{N} a_i}{Peak\_rate \times N} = \frac{\frac{1}{N}\sum_{i=1}^{N} a_i}{Peak\_rate} = \frac{Average\_rate}{Peak\_rate}. \tag{3-1}$$

When this factor equal to 1, it means the data rate is CBR, and as the client buffer increasing, BFF will more approach 1.

Fig. 10 is the illustration of the transmission schedule of different video traces which are yielded by MVBA with 1Mbits client buffer. (a) is "Terminator One" (b) is "Jurassic Park One" (c) is a video of baseball game. The same, the rate variation of transmission schedules is lowered.

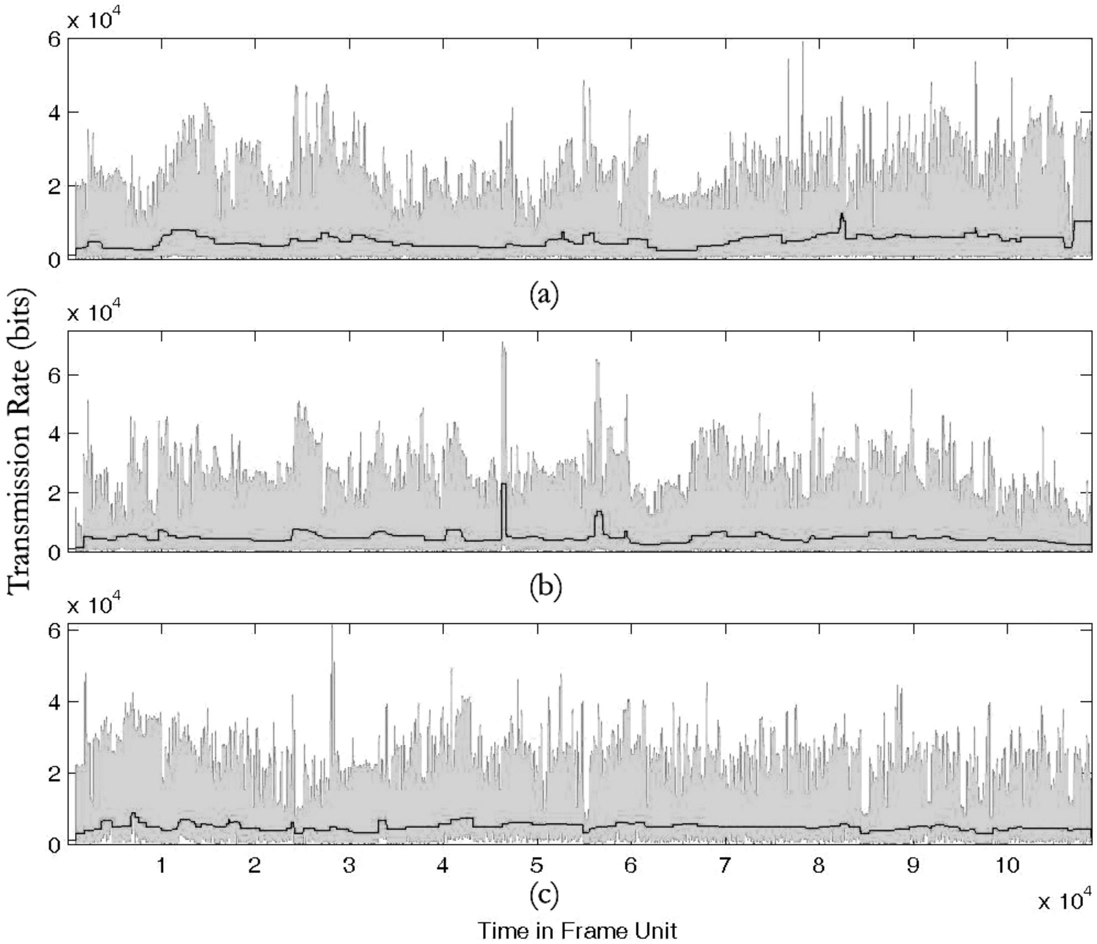



Figure 10: Three transmission schedules of (a) Terminator One, (b) Jurassic Park One, (c) Baseball game, which are yield by MVBA with 1Mbit client buffer. Where the gray part is the original video trace data and the black solid line is the smoothed transmission schedule.

The following Fig. 11 illustrates the effect of MVBA on the factors, peak rate, variance in transmission rate and BFF. It shows these factors as a function of client buffer. When the buffer client is very small, the transmission schedule is close to the unsmoothed schedule. As the client buffer increasing, the variance of rate and peak rate will decrease, the BFF will increase, and the transmission rate is more close to CBR.

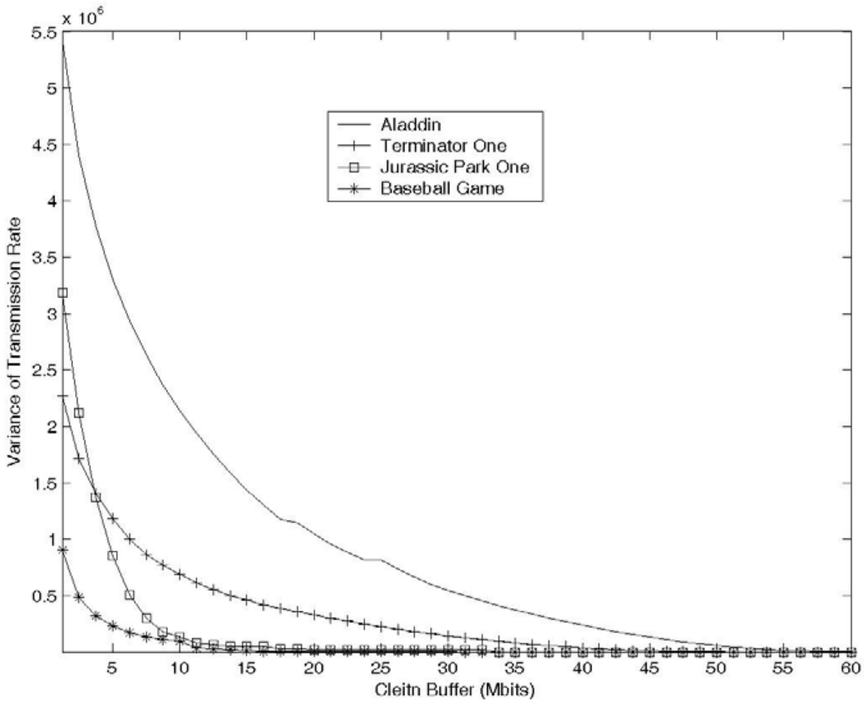



Figure 11 (a)

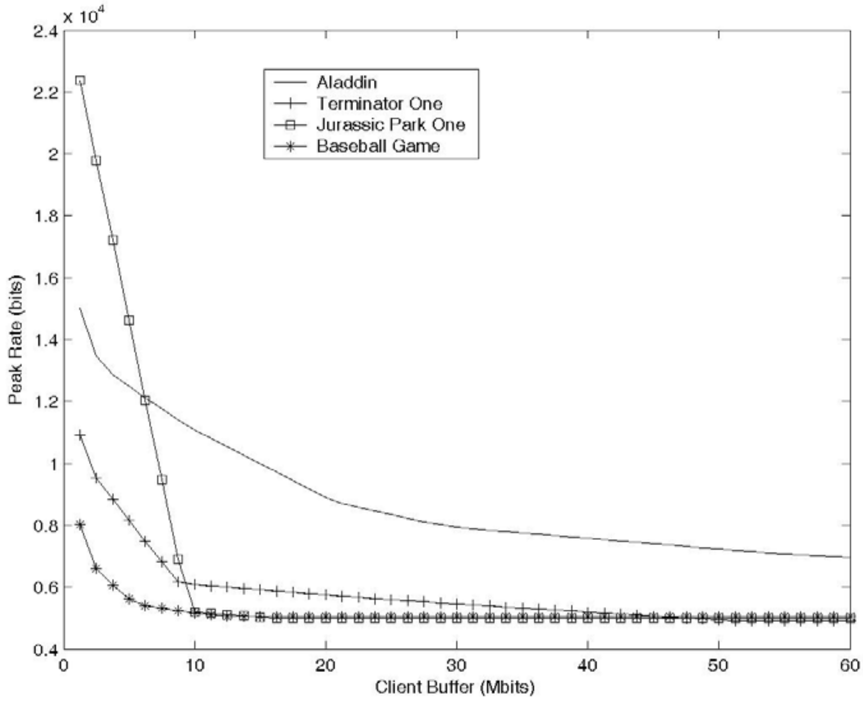



Figure 11 (b)

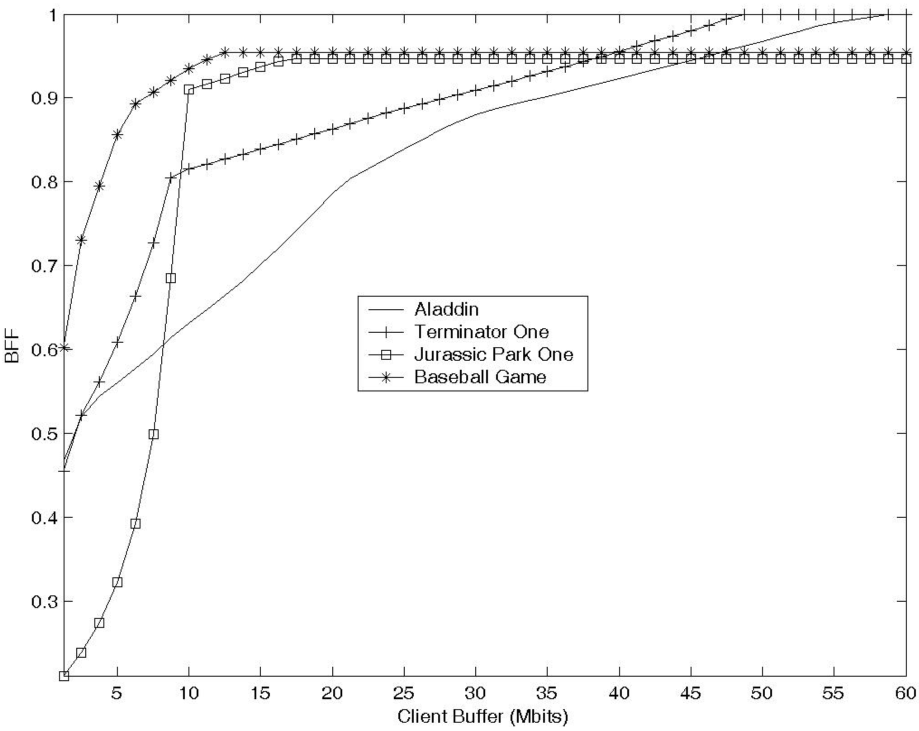



Figure 11 (c)

Figure 11: These graphs illustrate the (a) peak rate, (b) variance in transmission rate and (c) BFF as the function of client buffer.

# 4. Packet based Smoothing Algorithm

In most video streaming system framework, video data is packetized after being compressed, and then transmitted packet by packet in each frame slot. Therefore, transmission rate can only be controlled by transmitted number of packets in such system, but can't be controlled precisely according to MVBA schedule. See Fig. 12 as an example of this problem. In the example of Fig. 12, there are 3000 bits data waiting to be transmitted at 4 frame slots, these data are segmented into packets, which are in a size of 500 bits, and the rate of a CBR segment generated by certain conventional smoothing algorithm, such as MVBA, which we mentioned in Chapter 2, equals to 750 bits per frame slot. In Fig. 12, the black segment represents the CBR segment, but this CBR segment cannot be implemented in this case, because the server can only transmit data in a unit of packet, the transmission rate can only be the multiple of 500 bits. In this example, the server transmits two packets at t frame slot, one packet at t+1 and t+2 frame slots, and two packets at t+3 frame slot and the rate is 1000 bits per frame slot at t and t+3 frame slots and 500 bits per frame slot at t+1 and t+2 frame slot. The gray squares are the packet transmitted at the corresponding frame slot, and the gray segments represent the transmission schedule. This schedule can be implemented in the packet based transmission system but isn't an optimum one.

.

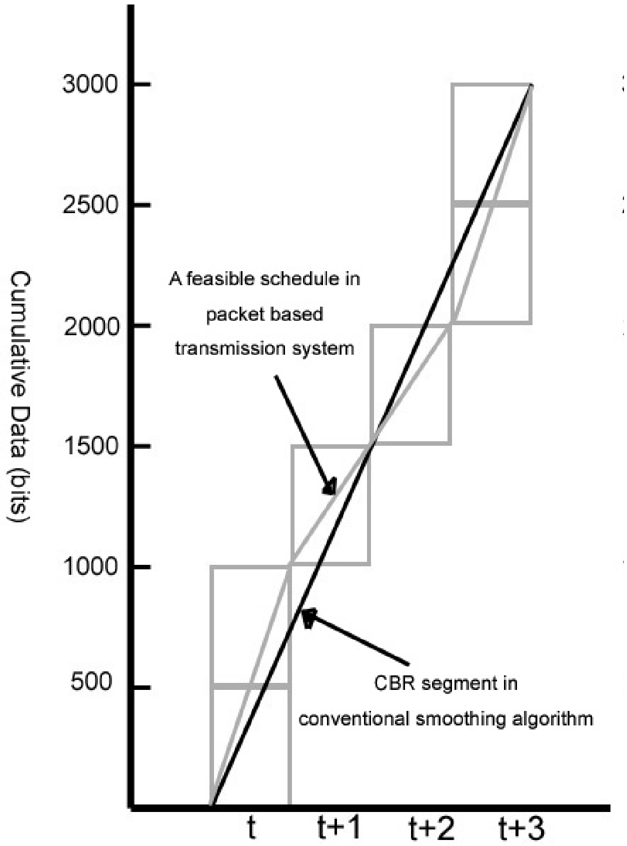



Figure 12: An illustration of that transmission rate is controlled by packet number control.

Under this condition, the transmission schedule generated by conventional smoothing algorithm, such as we mentioned in chapter 2, cannot be used in the packet based transmission system, because the server can't control the transmission rate precisely according to this schedule. In this chapter, we propose two methods for smoothing in packet based transmission system. These two algorithms can find the sub-optimum transmission schedule, which can be implemented in packet based transmission system. One is to smooth the number of packets transmitted in each frame slot. Another one is to approximate the MVBA transmission schedule by packet mapping. In this chapter, we discuss these two algorithms in detail.

## 4.1 Number-of-Packets Based Smoothing Algorithm

In this section we describe the difference between Number-of-Packets Based Smoothing Algorithm and MVBA smoothing algorithm discussed previously, and demonstrate the relationship between transmitted packet number and transmission rate.

This algorithm aims to find a transmission schedule with a CBR packet rate. When the server transmitted the same number of packets at each frame slot, the transmission rate can be expected to be smooth. However, it's similar to the MVBA smoothing algorithm described above. The finite client buffer makes the CBR transmission schedule of number of packets infeasible, only some piecewise-CBR schedules can obey the upper bound and lower bound. Thus, the transmission schedule generated by this algorithm will be with a more stable rate in transmitted number of packets, and the actual transmission rate is the sum of all packets transmitted at the corresponding frame slot. We will discuss the difference between the number-of-packets based smoothing algorithm and MVBA.

### 4.1.1 Modification of Upper Bound in Packet Based Smoothing Algorithm

Because the server transmits video data in a unit of packet, the transmission rate can't be controlled arbitrarily. This fact induce that the client buffer can't be utilized completely, therefore the upper bound of the two packet based smoothing algorithm have to be modified.

In packet based smoothing algorithm, the transmission data is restricted in unit of packet. Therefore a server can only transmit the video data packet-by-packet, the data that can be transmitted ahead to client buffer will be restricted and it induces client buffer not to be utilized fully. See the following Fig. 13 as an example. In Fig. 13, at the first frame slot, the server only can transmit the first six packets. If the client receives the seventh packet at the first frame slot, its buffer will overflow, and so will the buffer at second and third frame slot. Therefore, the upper bound will be modified to the level of the sum of the first six packets data at the first three frame slots. The upper bound at each frame slot will be modified in this way. The lower bound is the same with MVBA, because even the frame data is cut into packet, the sum of the packets data of a certain frame equals to its frame data.

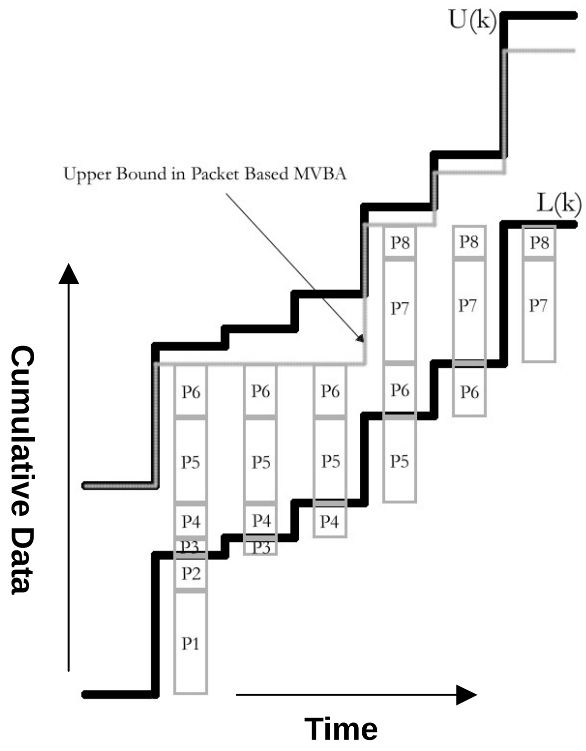



Figure 13: The difference of upper bound and lower bound between conventional smoothing algorithm and packet based smoothing algorithm. Where the black bold line represents the upper bound and lower bound in conventional smoothing algorithm, gray squares are data packets, and the gray line is the upper bound in packet based smoothing algorithm.

Next, the upper bound and lower bound are represented in packet number. See the following Fig.14.

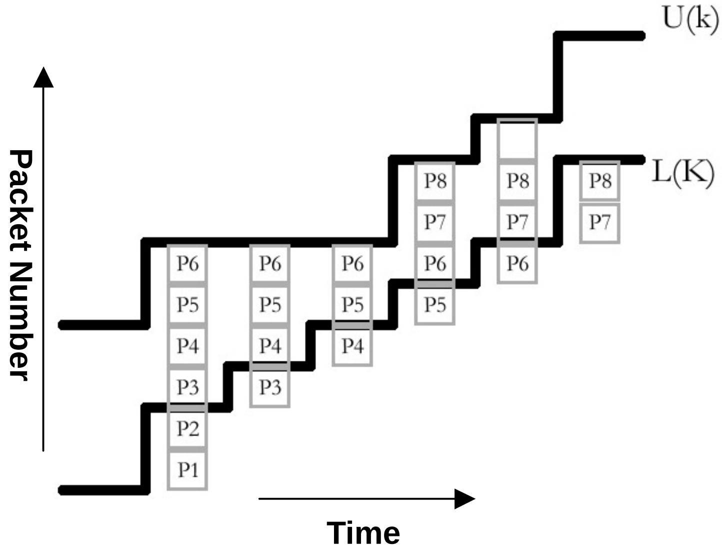



Figure 14: The upper bound and lower bound represented in number-of-packets form in packet based smoothing algorithm. We only consider the number of packets. $L(k)$ and $U(k)$ here is in unit of number of packets and represents the corresponding upper bound and lower bound in the previous figure.

In Fig. 14, the amount of data of each packet is ignored, and only the packet number is concerned, because we only care if the transmitted number of packets stays within the upper bound and lower bound. The server has to transmit until $2_{th}$ packet number at least and transmits until $6_{th}$ packet number at most at the first frame slot.

Thus, the lower bound in packet based smoothing algorithm is the total number of packets of previous frame slots,

$$L'(k) = \sum_{i=1}^{k} p_i \tag{4-1}$$

where $p_i$ is the number of packets of frame i, and upper bound equals to the lower bound plus the number of packets, which can be preserved in client buffer at $i^{th}$ frame slot.

$$U'(k) = L'(k) + b_i \text{，} \tag{4-2}$$

where $b_i$ represents the number of packets which can be transmitted ahead at $i_{th}$ frame slot. Take Fig.14 as an example, $b_1$ equals to 4, because at this frame slot, the client buffer can contain up to the sixth packet, and the client will consume the first two packets, and $b_2$ equals to 3, $b_3$ equals to 2, and etc.

Let $S'(k)$ to be a transmission schedule representing the number of packets transmitted during k frame slot.

$$S'(k) = \sum_{i=1}^{k} s'_i \text{，} \tag{4-3}$$

where $s'_i$ is the transmitted number of packets at $i_{th}$ frame slot. If the schedule is feasible, it has to be within upper bound and lower bound,

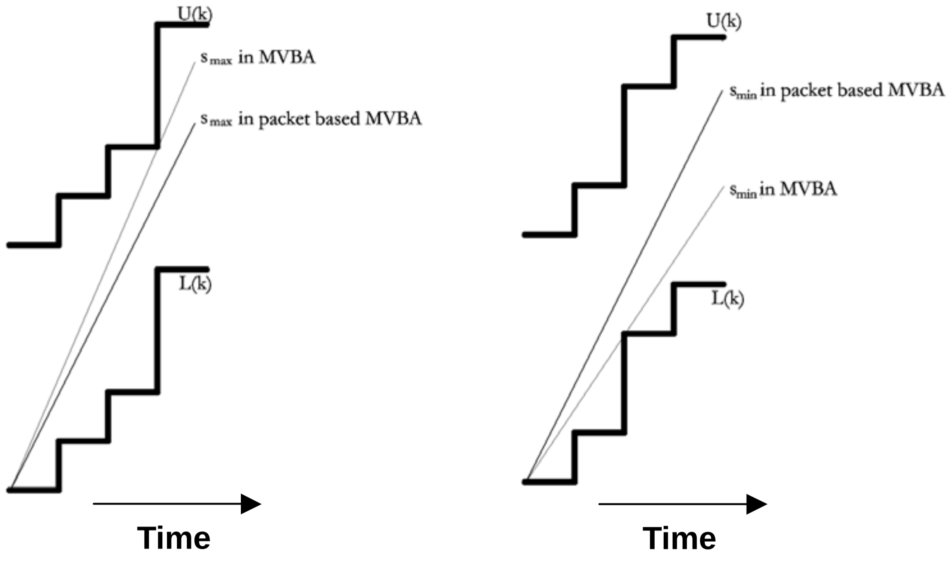
$$L'(k) \le S'(k) \le U'(k) \tag{4-4}$$

If we don't care the actual data of each packet in packet based transmission system, the modified upper bound and lower bound represented in amount of data form and upper bound and lower bound represented in number of packets form are equivalent. Thus the area enclosed by the upper bound and lower bound represented in amount of data form also equals to the area enclosed by upper bound and lower bound in number of packets form. Therefore, a transmission schedule, which stays within a set of upper bound and lower bound must stays within the other set of upper bound and lower bound as well. So a feasible schedule generated by packet based smoothing, which stays in the area enclosed by the upper bound and lower bound in number of packets form is also a feasible schedule and will stay in the area enclosed by the upper bound and lower bound in amount of data form.

### 4.1.2 Modification of integer transmission rate

The transmitted number of packets at each frame slot needs to be integer in a transmission schedule, which is generated by packet based smoothing algorithm. However, the rate generated by MVBA is unnecessarily to be integer. So we have to make the integer modification.

The transmission rate of an MVBA optimum schedule equals to $s_{\max_{a,b}}$ or $s_{\min_{a,b}}$ in MVBA procedure. Therefore, calculation of $s_{\max_{a,b}}$ and $s_{\min_{a,b}}$ in Number-of-Packets Based Smoothing Algorithm has to be modified as the following Fig. 15.




(a) (b)

Figure 15: The modification of integer transmission rate. (a) is a condition of $s_{\max_{a,b}}$. In order to prevent buffer overflowing, $s_{\max_{a,b}}'$ in Number-of-Packets Based Smoothing Algorithm takes the integer portion of $s_{\max_{a,b}}$, and truncates the decimal part. (b) is a condition of $s_{\min_{a,b}}$. In order to prevent buffer underflowing, $s_{\min_{a,b}}'$ rounds to the nearest next integer of $s_{\min_{a,b}}$.

When $s_{\max_{a,b}}$ is not an integer, $s_{\max_{a,b}}'$, the integer modified maximum rate in Number-of-Packets Based Smoothing Algorithm, must be smaller than the original $s_{\max_{a,b}}$, because $s_{\max_{a,b}}$ is the maximum feasible CBR transmission rate for all CBR segment in interval $[a,b]$. Thus the integer modification of $s_{\max_{a,b}}'$ is

$$s_{\max_{a,b}}' = \left\lfloor \min_{a+1 \le t \le b} \frac{U(t) - (L(a) + q)}{t - a} \right\rfloor , \qquad (4\text{-}5)$$

Similarly, the integer modified minimum rate in Number-of-Packets Based Smoothing Algorithm, $s_{\min_{a,b}}'$, must be larger than $s_{\min_{a,b}}$. That is

$$s_{\min_{a,b}}' = \left\lceil \max_{a+1 \le t \le b} \frac{L(t) - (L(a) + q)}{t - a} \right\rceil , \qquad (4\text{-}6)$$

where $\lfloor \ \rfloor$ is the floor function, $\lfloor x \rfloor$ returns the integer part of x, and $\lceil \ \rceil$ is the ceiling function, $\lceil x \rceil$ returns the next nearest integer of x.

Thus, $s_{\max_{a,b}}'$ and $s_{\min_{a,b}}'$ are guaranteed to be integer, and the transmission schedule must stay within the upper bound and lower bound. So, the transmission schedule generated by Number-of-Packets Based Smoothing Algorithm is feasible, and the packet rate is integer for each frame slot.

### 4.1.3 The procedure of Number-of-Packets based smoothing algorithm

The initial settings of this algorithm are the same as the MVBA, and the process starts from the beginning of the video. After the upper bound and lower bound is calculated, these upper bound and lower bound will be modified to the number-of-packet form, as Fig. 14. Then the process will calculate the $s_{\min_{a,b}}$, $s_{\max_{a,b}}$, $t_{L_{a,b}}$, $t_{U_{a,b}}$ and $s_{\min_{a,b+1}}$, $s_{\max_{a,b+1}}$, and perform the integer modification on $s_{\min_{a,b}}$, $s_{\max_{a,b}}$ and $s_{\min_{a,b+1}}$, $s_{\max_{a,b+1}}$. Then the process checks whether the overflowing condition or underflowing condition happens or not, which are described in section 2.1.2. Then the reaction is the same as MVBA. The process will repeat until the total video is be segmented. Finally, the sum of the data of the corresponding number of packets is the actual transmission rate.

## 4.2 Packet Mapped Smoothing Algorithm

This algorithm is to find a suboptimal transmission schedule, which approximate the optimum schedule generated by MVBA as well as possible, in packet based transmission system. We describe how transmission rate could be controlled to approximate to the optimal schedule generated by MVBA and stay within the feasible area by control of transmitted number of packets.

### 4.2.1 Upper and Lower Bound in Optimum Transmission Schedule

In MVBA algorithm, $s_{\max_{a,b}}$ is the maximum feasible transmission rate in interval [a,b]. If the total transmitted data is above line segment, $s_{\max_{a,b}}$, the client buffer may be overflowing within this interval. Therefore, when MVBA decided to transmit in rate $s_{\max_{a,b}}$ in interval [a,b], the total transmitted data can't be above line segment $s_{\max_{a,b}}$, as the following Fig.16(a). The two block bold lines represent the upper bound, U(k), and lower bound, L(k). The gray squares under L(k) is the data packet at the corresponding frame slot. Gray line segment is the optimum transmission rate decided by MVBA in this interval. Black fine line segments are the transmission rate controlled by transmitted number of packets, and the gray squares between upper bound and lower bound are the packet transmitted at the corresponding frame slot. In Fig.16(a), if server transmits the next packet, which is transmitted at 4th frame slot, at 3rd frame slot, the total transmitted data will be above line segment $s_{\max_{a,b}}$, and it results in buffer overflowing. Thus the sum of the transmitted data can’t be above $s_{\max_{a,b}}$, when MVBA decide to transmitted with rate $s_{\max_{a,b}}$ in interval [a,b].

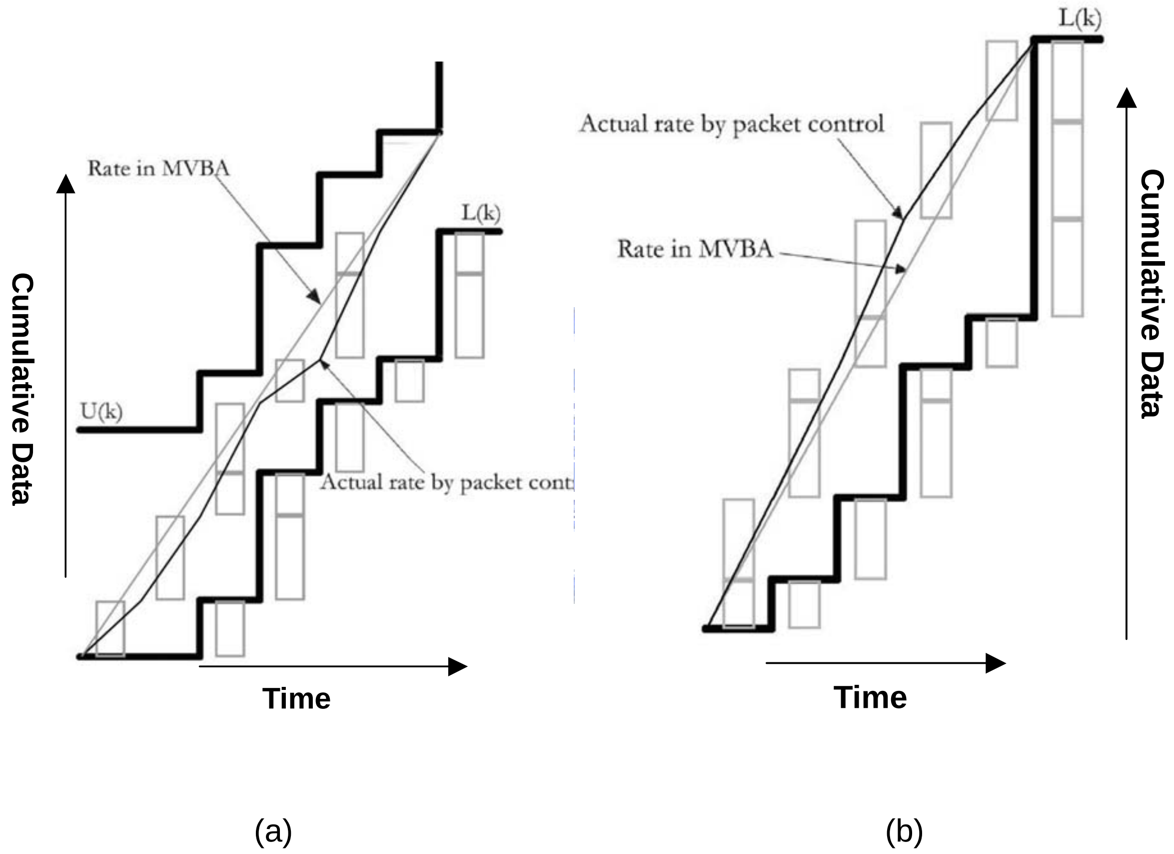



Figure 16: Number of packets controlled rate control. (a) represents the condition that the optimum schedule is $s_{\max_{a,b}}$. The total transmitted data can't be above the line segment $s_{\max_{a,b}}$ to prevent buffer overflowing. (b) represents the condition that the optimum schedule is $s_{\min_{a,b}}$. The total transmitted data can't be below the line segment $s_{\min_{a,b}}$ to prevent buffer underflowing.

Similarly, $s_{\min_{a,b}}$ is the minimum feasible rate in MVBA, if the total transmitted data is below line segment $s_{\min_{a,b}}$, the client buffer may be underflowing within this interval. Therefore, when MVBA decided to transmitted in rate $s_{\min_{a,b}}$ in interval [a,b], the total transmitted data can't be below line segment $s_{\min_{a,b}}$, as Fig.16(b).

### 4.2.2 The procedure of Packet Mapped smoothing algorithm

The packet mapped smoothing algorithm performs original MVBA with the modified upper bound and lower bound in amount of data form, and then calculate the packet number that the cumulative data from the end of the previous frame to this calculated packet number is nearest to the optimum transmission schedule, which is generated by MVBA, with obeying the two principles described above. By this way, the transmission schedule will approximate to optimum schedule of MVBA and be feasible.

## 4.3 Implementation of Packet Based Smoothing Algorithm

To transmit smoothly without modifying video stream standard is the greatest benefit of packet based smoothing algorithm. Take MPEG4 as an example, after encoder compresses the video data, the compressed data will be cut into packets and be transmitted on the network. The encoder will mark the quantization coefficients, the frame number and other parameters in the packet header. Then the client reassembles these packets and decodes frame data according to the header of packets. However, the optimum transmission schedule generated by MVBA won't match the data packets exactly. If the server would like to follow the optimum transmission schedule completely, the data packets have to be segmented again. Thus, there must be a mechanism, which reassembles these new packets in client. Otherwise, these packets can't be decoded correctly. But, this mechanism does not conform to the standard. So, the packet based smoothing algorithm can make the clients conform the video compression standard, moreover, it can make network to be utilize efficiently.

The token bucket algorithm can control the amount of data, which are inserted into the network. It’s a transmission mechanism that determines when the traffic can be transmitted, based on the presence of tokens, and the bucket leaks a variable number of token periodically. The server can control the period. Then the streamer will transmitted the same number of packets according to the leaked number of tokens. We can implement our algorithm easily by using this token bucket algorithm. In our system, we insert a token bucket between the encoder and network layer. After the video is compressed, the data will be packetized and this packet will be stored in a buffer in packet order. The token bucket leaks some tokens according to the transmission schedule generated by our packet based smoothing algorithm. Then the streamer releases the same amount packets. Therefore, our smoothing algorithm can control the transmission rate. Fig. 17 shows this mechanism.

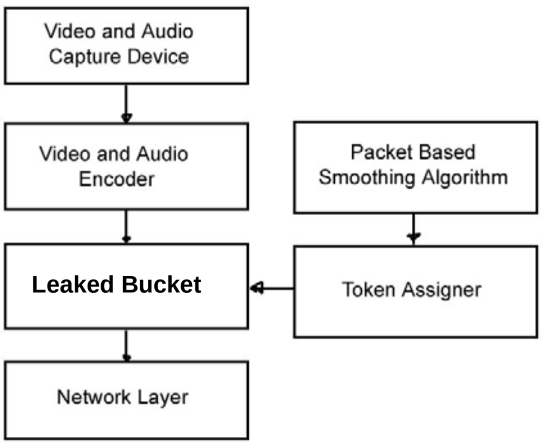



Figure 17: Block diagram of a video transmission server with packet based smoothing algorithm.

# 5. Simulation of Packet Based Smoothing Algorithm

In this chapter, we also use the video trace "Aladdin" to demonstrate examples of Number-of-Packets Based Smoothing Algorithm and packet mapped MVBA. This video trace is in QCIF format, 30 frames per seconds, and compressed with MPEG4 and with I-P-B quantization level 10-14-16. We perform these two algorithms with four different maximum packet sizes, 512 bits, 1024 bits, 1536 bits and 2048 bits and observe the statistical parameters of simulation result to compare with the original MVBA algorithm.

## 5.1 Performance of Number-of-Packets Based Smoothing Algorithm

At the first of the packet based smoothing algorithm, the upper bound has to be modified. The procedure will find the packet number, until which the cumulative data is nearest to the original upper bound but isn't above it. By this way, the upper bound and lower bound in number of packets form are generated. Then the upper bound in packet based smoothing is modified to the cumulative data until the corresponding packet number for each frame. Thus, the modified upper bound and lower bound in amount of data form are found. Fig. 18 and Fig. 19 are the modified upper bound and lower bound of the 100th to 110th frames of video "Aladdin" in packet based smoothing algorithm. Fig. 18 is with 32K bits client buffer and 1024 bits maximum packet size. Fig. 19 is with the same client buffer and 2048 bits maximum packet size. It shows that the larger the maximum packet size is, the more the difference between the original upper bound in MVBA and the modified upper bound in packet based smoothing algorithm.

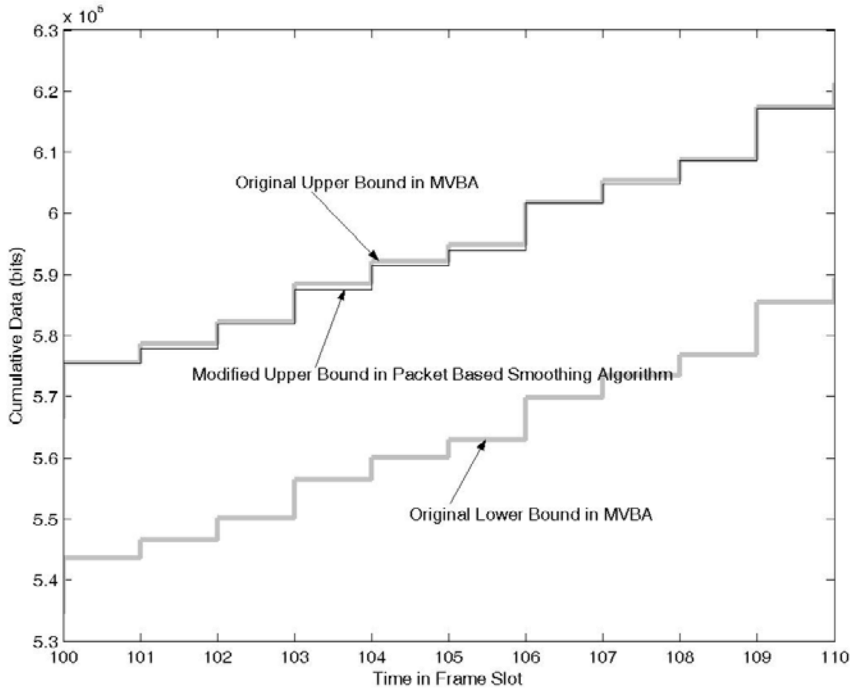



Figure 18: The modified upper bound and lower bound of “Aladdin” with 32K client buffer and 1024 bits maximum packet size.

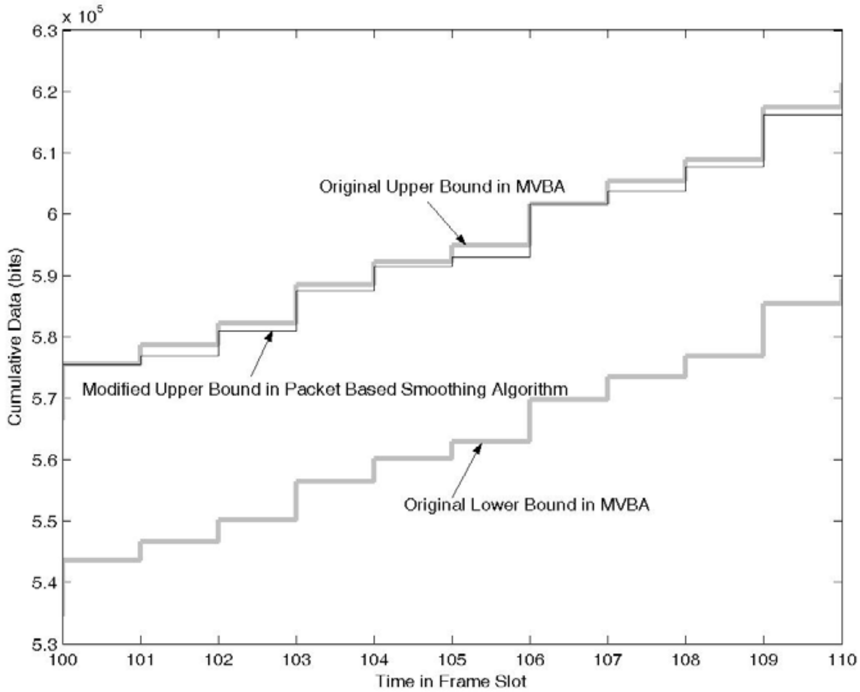



Figure 19: The modified upper bound and lower bound of "Aladdin" with 32K client buffer and 2048 bits maximum packet size.

Then the procedure performs the integer modified MVBA algorithm on the modified upper bound and lower bound in number of packets form to find the transmission schedule of number of packets. Fig. 20 and Fig. 21 are the examples of the transmission schedule of video "Aladdin" with 32K bits client buffer and 1024 bits maximum packet size and 2048 bits maximum packet size respectively. It shows that the number of packets with 1024 bits maximum packet size is about twice as the packet number with 2048 bits maximum packet size at the same frame slot, because the number of packets with 1024 bits maximum packet size is about twice as the number of packets with 2048 bits maximum packet size. The relation between maximum packet size and number of packets will be discussed in section 5.3. Because the extreme difference between the two set of upper bound and lower bound with different maximum packet size, the transmission schedule of number of packets are also quite different.

After the transmission schedule of number of packets is generated, we sum the corresponding number of packets for each frame to calculate the actual transmission rate. Fig. 22 and Fig. 23 are the two results. Compare Fig. 20, Fig. 21 and Fig. 22, Fig. 23, it is apparent that although the server transmits the same number of packets, the actual rate varies because the size of the successive packet may be different.

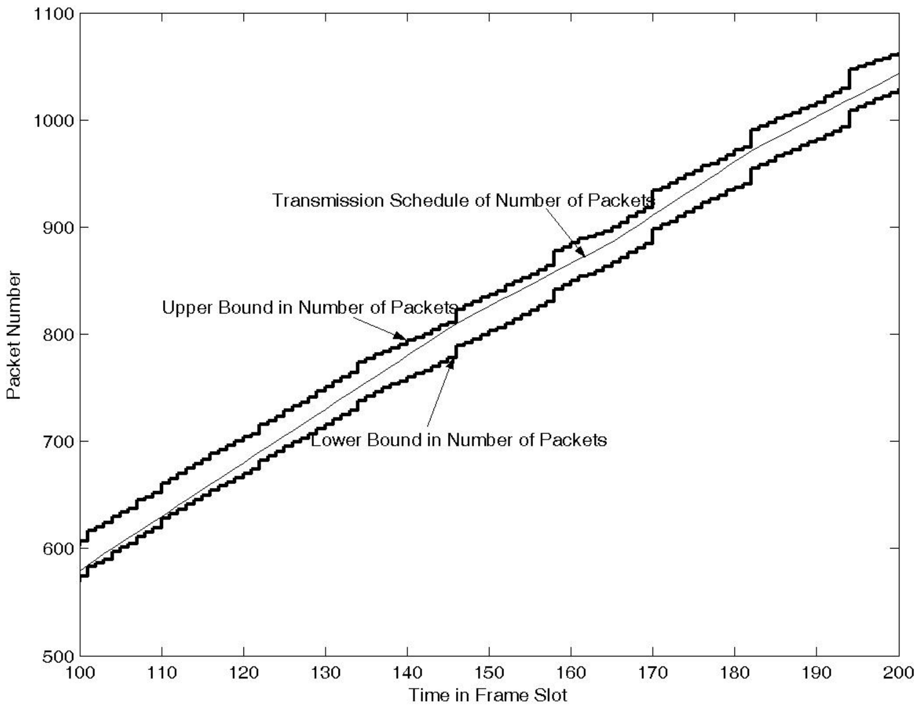



Figure 20: Transmission schedule of number of packets of “Aladdin” with 32K bits client buffer and 1024 bits maximum packet size.

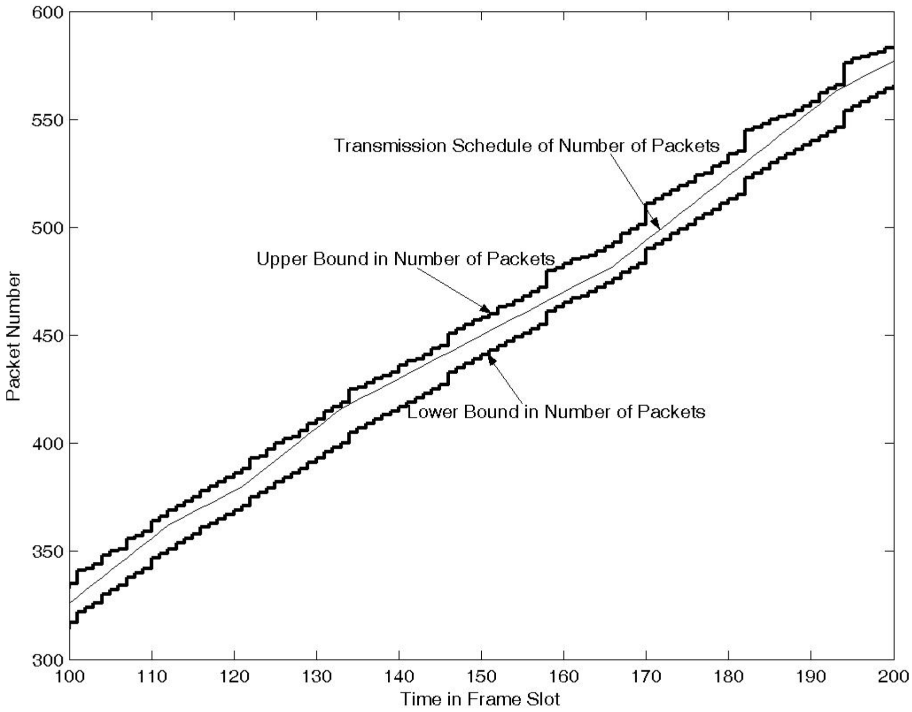



Figure 21: Transmission schedule of number of packets of “Aladdin” with 32K bits client buffer and 2048 bits maximum packet size.

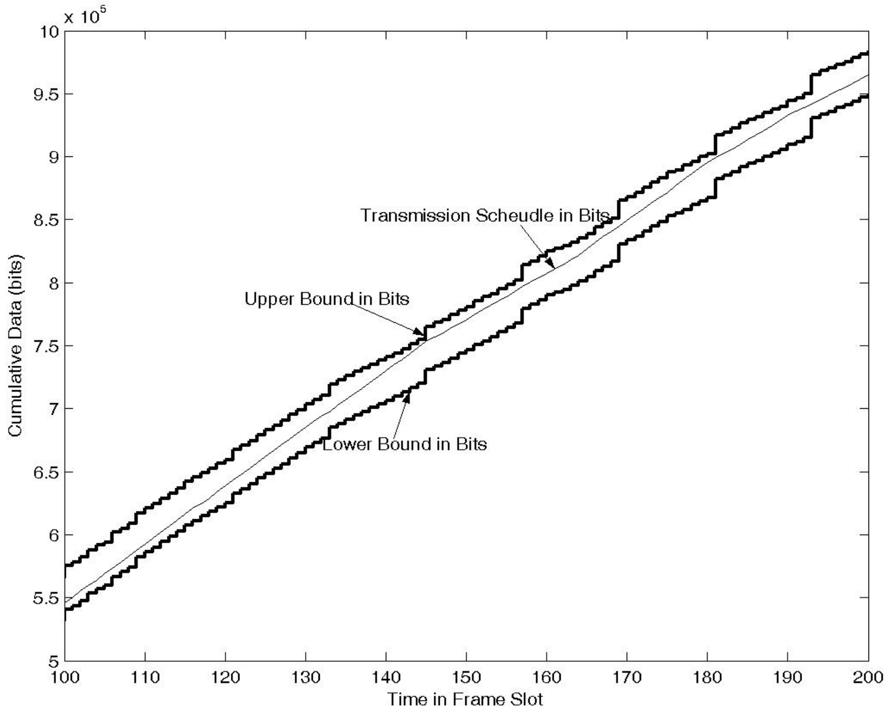



Figure 22: Transmission schedule of actual data rate of “Aladdin” with 32K bits client buffer and 1024 bits maximum packet size.

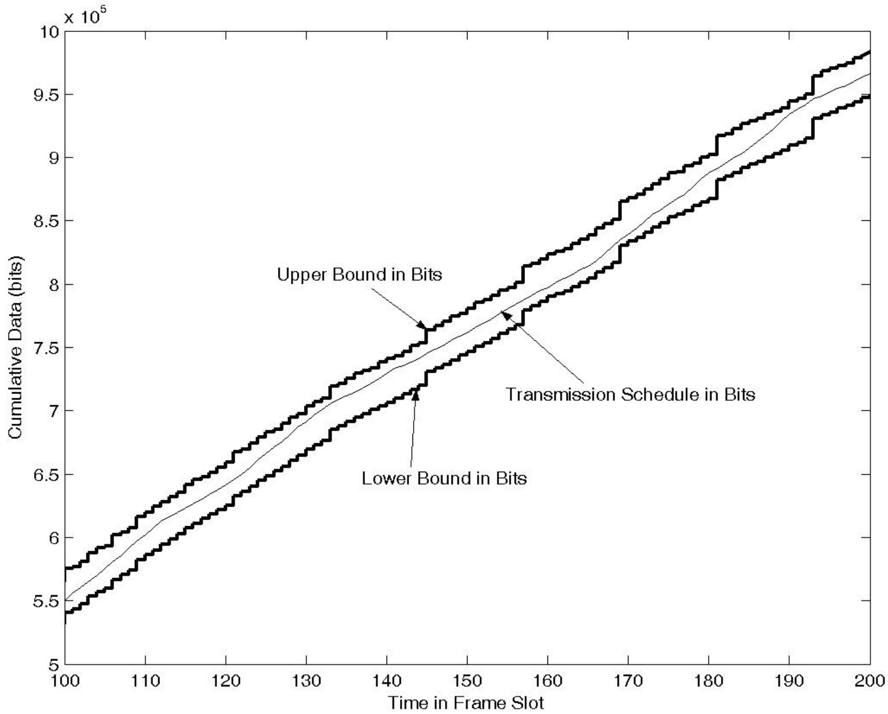



Figure 23: Transmission schedule of actual data rate of "Aladdin" with 32K bits client buffer and 2048 bits maximum packet size.

The following Fig. 24 and Fig. 25 are the transmission rate of the schedule generated by Number-of-Packets Based Smoothing Algorithm with 32K bit client buffer and 1024 bits and 2048 bits maximum packet size respectively. Fig. 24 (a) and Fig. 25 (a) is the number of packets transmitted at each frame slot, and Fig. 24 (b) and Fig. 25 (b) is the actual transmission rate at each frame slot.

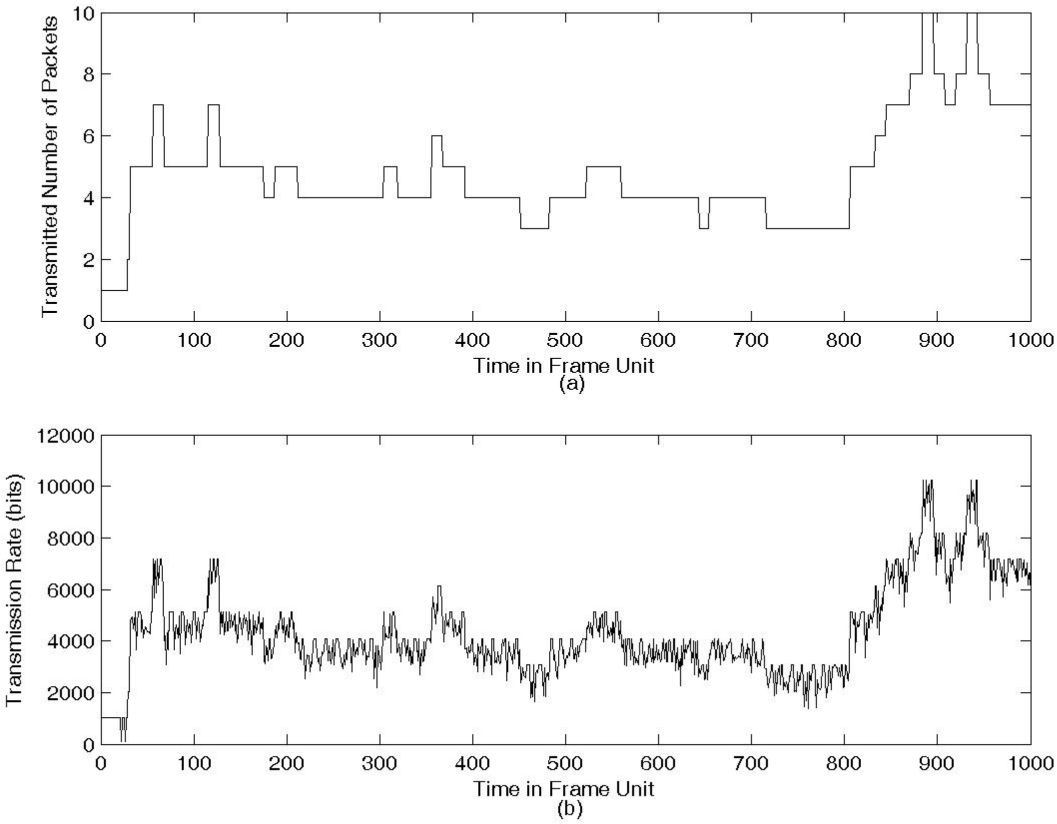



Figure 24: Transmission rate of the first 1000 frames of "Aladdin" generated by Number-of-Packets Based Smoothing Algorithm with 32k bits client buffer and 1024 bits maximum packet size. (a) is the number of packet transmitted at each frame slot. (b) is the actual transmission rate.

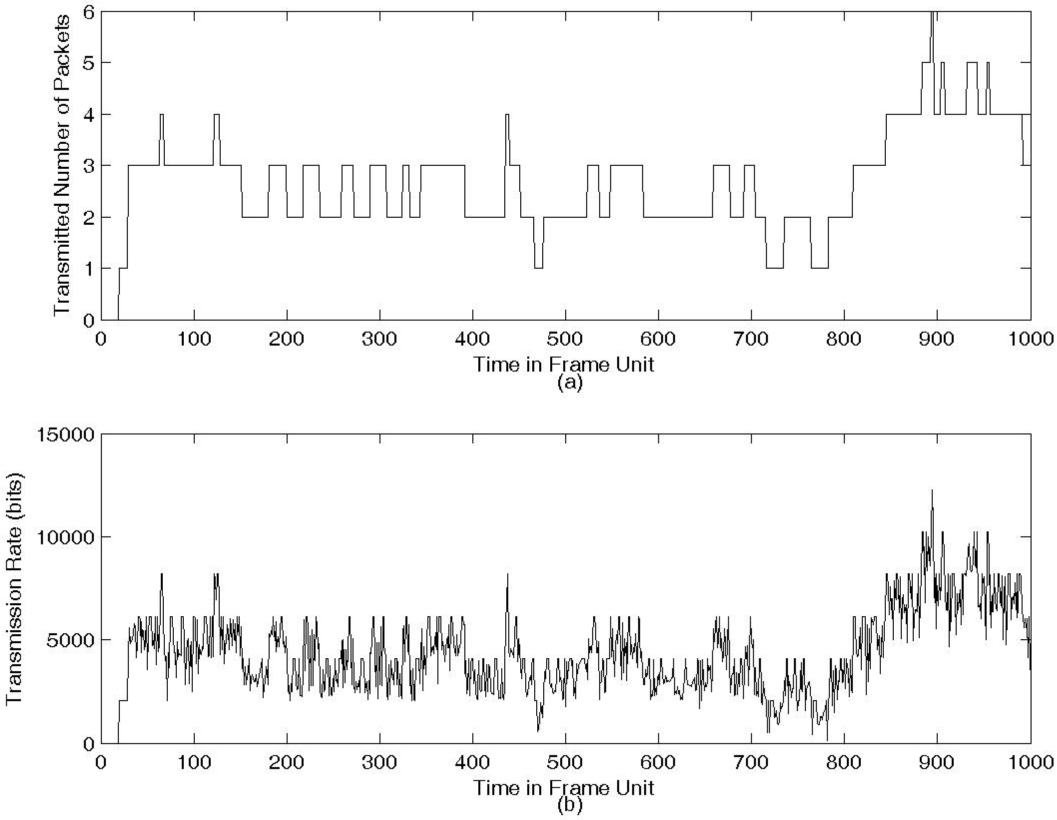



Figure 25: Transmission rate of the first 1000 frames of “Aladdin” generated by Number-of-Packets Based Smoothing Algorithm with 32k bits client buffer and 2048 bits maximum packet size. (a) is the number of packet transmitted at each frame slot. (b) is the actual transmission rate.

The following Fig. 26 and Fig. 27 are the transmission rate of the total video “Aladdin” generated by Number-of-Packets Based Smoothing Algorithm with 100k bits and 1M bits client buffer size respectively and maximum packet size 1024 bits, and Fig. 28 and Fig. 29 are with 100k bits and 1M bits client buffer size respectively and maximum packet size 2048 bits. Where (a) is the number of packets transmitted at each frame slot, and (b) is the actual transmission rate.

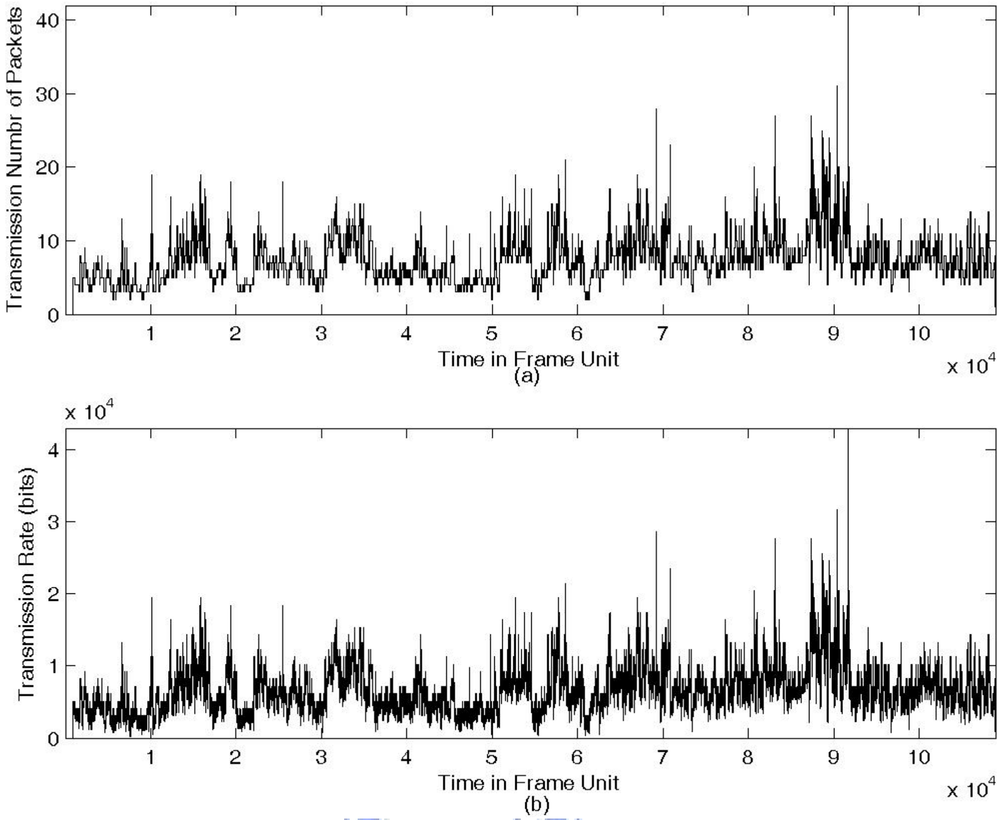



Figure 26: The transmission rate of the total video "Aladdin" generated by Number-of-Packets Based Smoothing Algorithm with 100k bits client buffer and 1024 bits maximum packet size. (a) is the transmission Number of packets at each frame slot. (b) is the actual transmission rate.

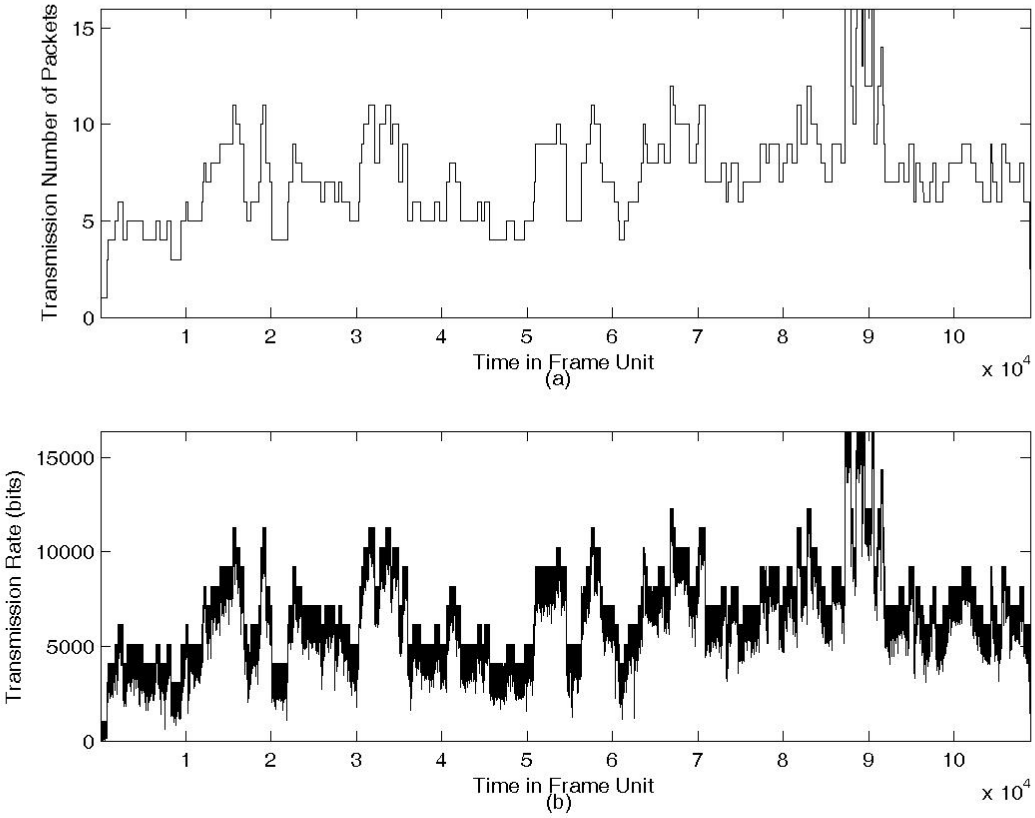



Figure 27: The transmission rate of the total video "Aladdin" generated by Number-of-Packets Based Smoothing Algorithm with 1M bits client buffer and 1024 bits maximum packet size. (a) is the transmission Number of packets at each frame slot. (b) is the actual transmission rate.

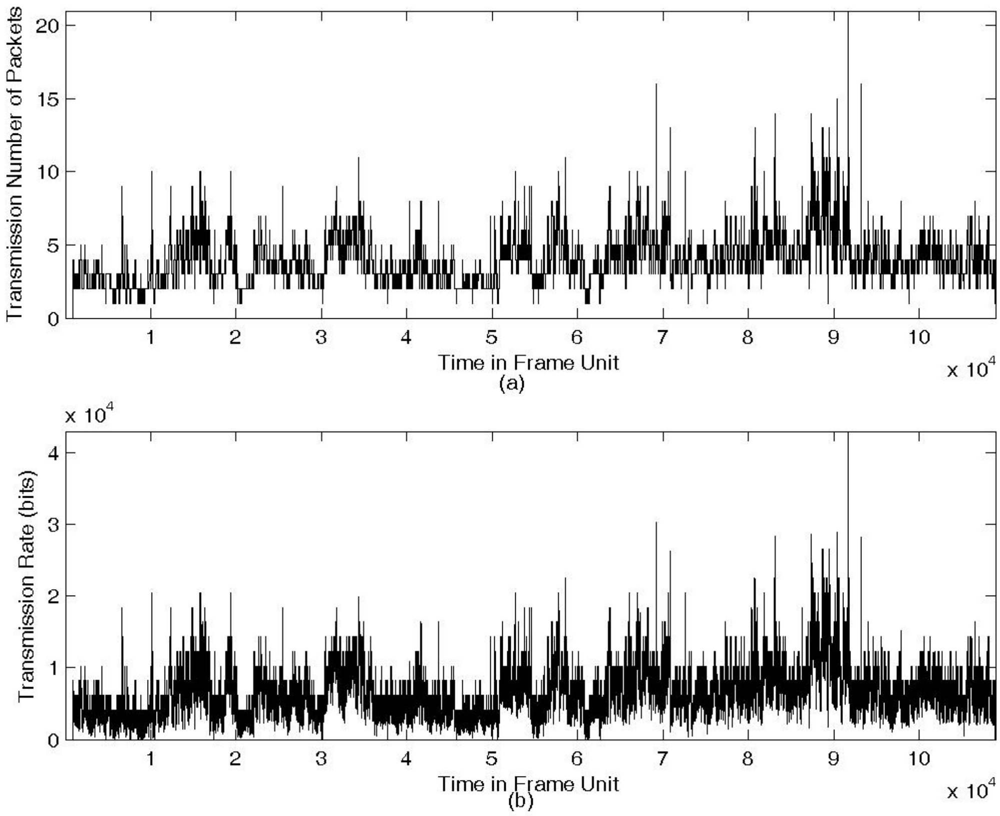



Figure 28: The transmission rate of the total video "Aladdin" generated by Number-of-Packets Based Smoothing Algorithm with 100k bits client buffer and 2048 bits maximum packet size. (a) is the transmission Number of packets at each frame slot. (b) is the actual transmission rate.

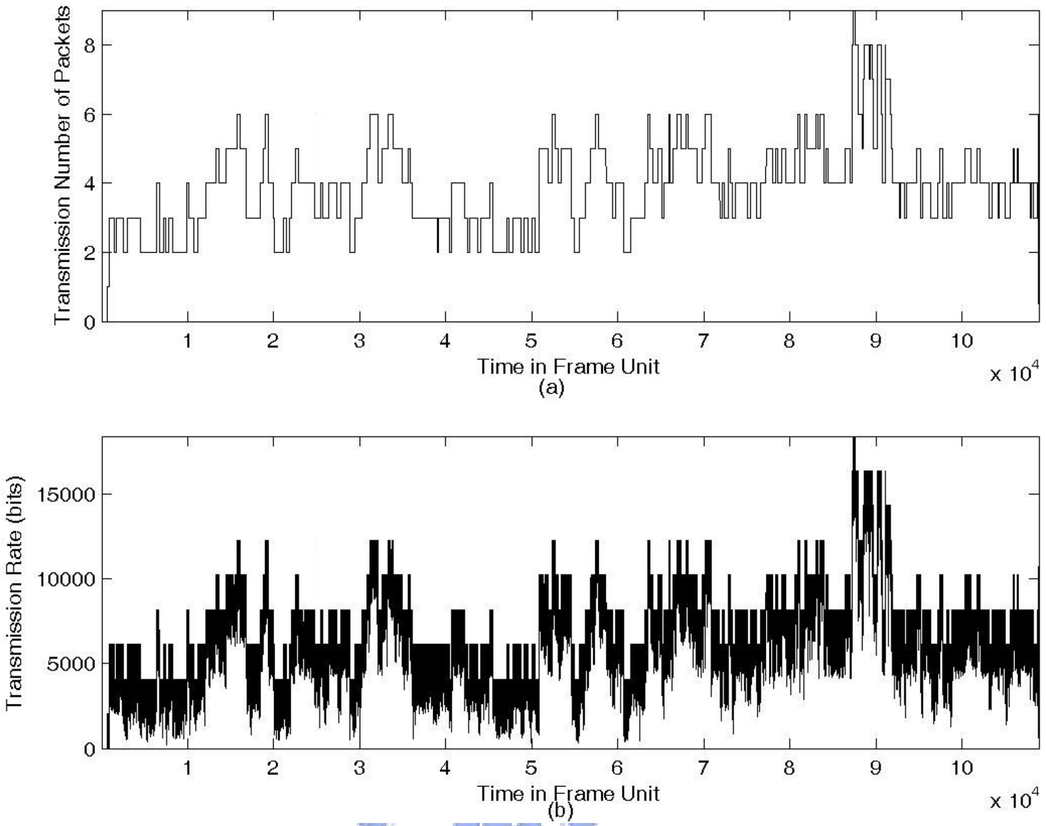



Figure 29: The transmission rate of the total video "Aladdin" generated by Number-of-Packets Based Smoothing Algorithm with 1M bits client buffer and 2048 bits maximum packet size. (a) is the transmission Number of packets at each frame slot. (b) is the actual transmission rate.

According to these examples, it's obviously that even though the number of packets transmitted at each frame slot is smoothed, the actual transmission rate is still with much variation because of the size difference between successive packets. For example, at B frame duration or at the time of static scene of a video, there may be more small packets, and at I frame duration or at the time of scene with high motion components, there will be more full packets. Thus, even the transmitted number of packets is the same, the actual transmission rate, which equals to the sum of the successive packet data, will be different. The variance of transmitted number of packets will tend to decrease as the client buffer increasing, and the actual transmission rate will vary in accordance with the variation of transmitted number of packets and the data variation between the succession packets. Therefore, the transmission rate with larger maximum packet size may tend to be with more variation, because the larger the maximum packet size is, the more the difference between successions packets can be. It's also shows that the transmitted number of packets will be influenced by the integer rate modification. The integer rate modification may induce the transmitted number of packets varying between two adjacent integers.

Because of the two uncertain factors, the transmitted number of packet and the difference size between successive packets, it's hard to predict the performance of Number-of-Packets Based Smoothing Algorithm with different client buffer and different maximum packet size. The larger client buffer may lower down the variation of transmitted number of packets, but the integer rate modification may make the transmitted number of packets varying more times between two adjacent integers with certain maximum packet size. The larger maximum packet size will induce the more variation between the successive packets, but the different maximum packet size will also yield different upper bound of number of packets. The different upper bound and lower bound will generate different transmission schedule. Therefore, we only can confirm that the Number-of-Packets Based Smoothing Algorithm with smaller maximum packet size and large client buffer size may tend to generate a schedule with lower variance in transmission rate in a high probability, but it also may generate a transmission schedule with higher variance in transmission rate in a low probability. This phenomenon is more apparently as the maximum packet size increasing. Fig. 30 illustrates this phenomenon. The curves of triangle, star, square, and cross mark correspond to the variance in transmission schedule generated by Number-of-Packets Based Smoothing Algorithm with maximum packet size 512 bits, 1024 bits, 1536 bits and 2048 bits respectively, as the function of client buffer size. The curve with no mark is the variance in transmission schedule generated by MVBA.

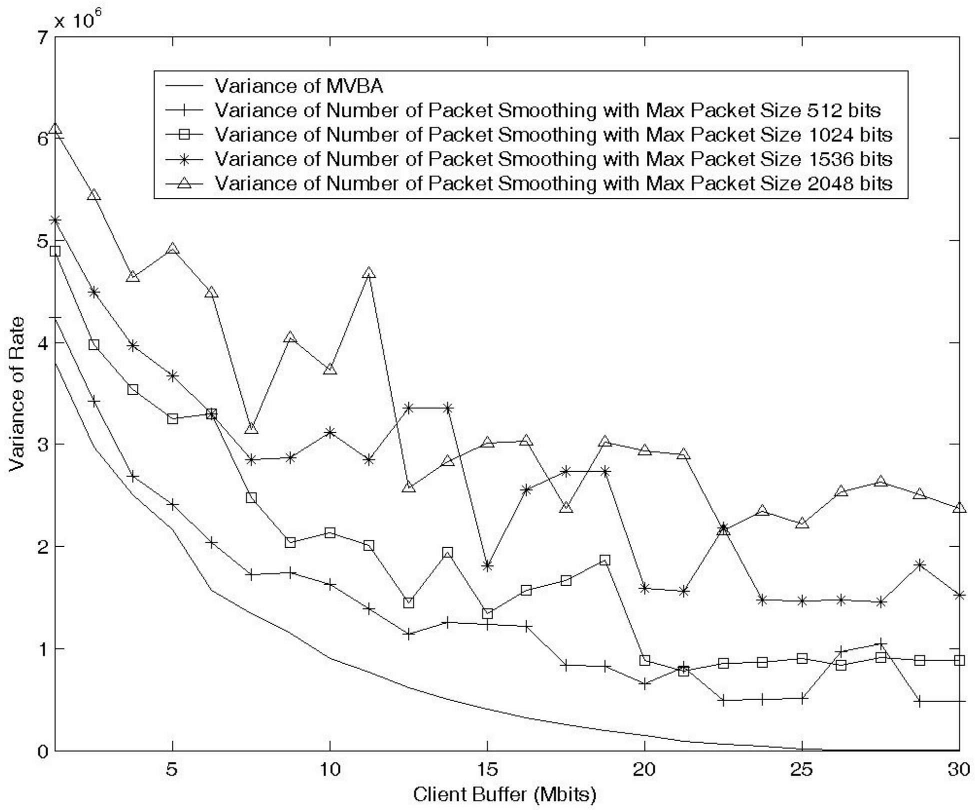



Figure 30: The variance in transmission rate as a function of client buffer with different maximum packet size.

Fig. 31 demonstrates the peak rate as a function of client buffer with different maximum packet sizes. It shows that peak rate is also influenced a lot by integer modification mechanism. The peak rate with smaller maximum packet size will goes down without ripple almost as the client buffer increases. When the maximum packet size grows up, the curve of peak rate as the function of client buffer will ripple violently. According to Fig.30, the peak rate is always a multiple of maximum packet sizes because the integer transmission rate modification mechanism. The peak rate of schedule generated by this algorithm equals to maximum transmitted number of packets multiply by maximum packet size.

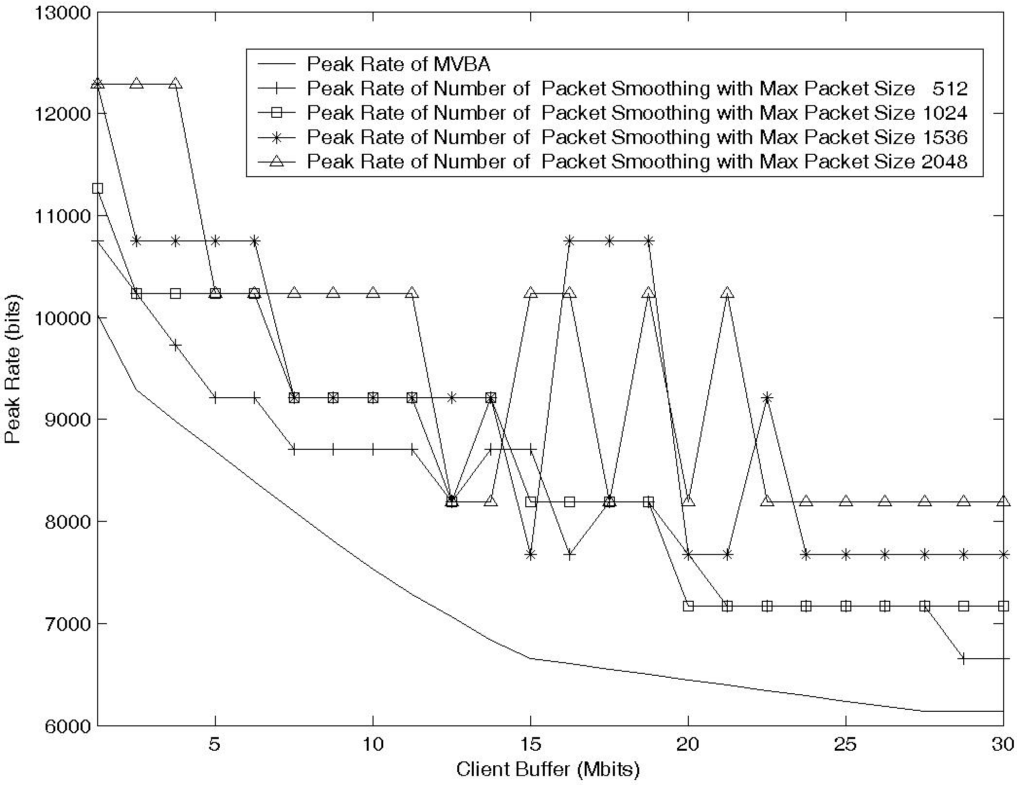



Figure 31: The graph illustrates the peak rate as the function of client buffer.

The curve of BFF, which is the inverse ratio of peak rate, will also ripple as the peak rate. The larger the maximum packet size is, the more violent the ripple is. See the following Fig. 32 as an example.

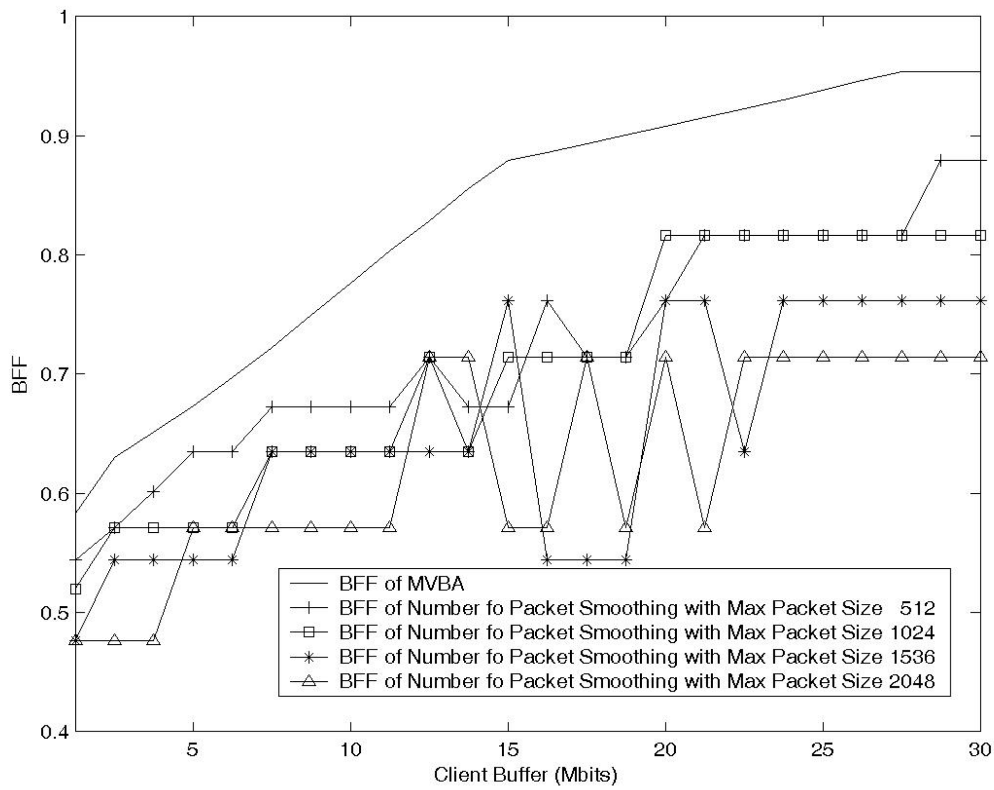



Figure 32: The graph illustrates the BFF as the function of client buffer.

## 5.2 Performance of Packet Mapped Smoothing Algorithm

In this section, we see the examples of Packet Mapped smoothing algorithm and compare its performance.

First, this algorithm finds the modified upper bound and lower bound in amount of data form by the same way described in previous section 5.1. Then it performs MVBA with the modified upper bound and lower bound to find the optimum transmission schedule of MVBA. After the optimum transmission schedule is calculated, the procedure will find the packet number at each frame slot, until which the cumulative sum of the packet data is nearest to the optimum transmission schedule generated by

MVBA, obeying to the principles described in section 4.2.1. Fig. 33 and Fig. 34 are the examples of the transmission schedule of the 150th to 180th frame of “Aladdin” under condition of $s_{\min_{a,b}}$ generated by MVBA and Packet Mapped Smoothing algorithm with 32K bits client buffer and 1024 bits and 2048 bits maximum packet size respectively. Similarly, Fig. 35 and Fig. 36 are the examples of the transmission schedule of the 690th to 720th frame of “Aladdin” under condition of $s_{\max_{a,b}}$ generated by MVBA and Packet Mapped Smoothing algorithm with 32K bits client buffer and 1024 bits and 2048 bits maximum packet size respectively.

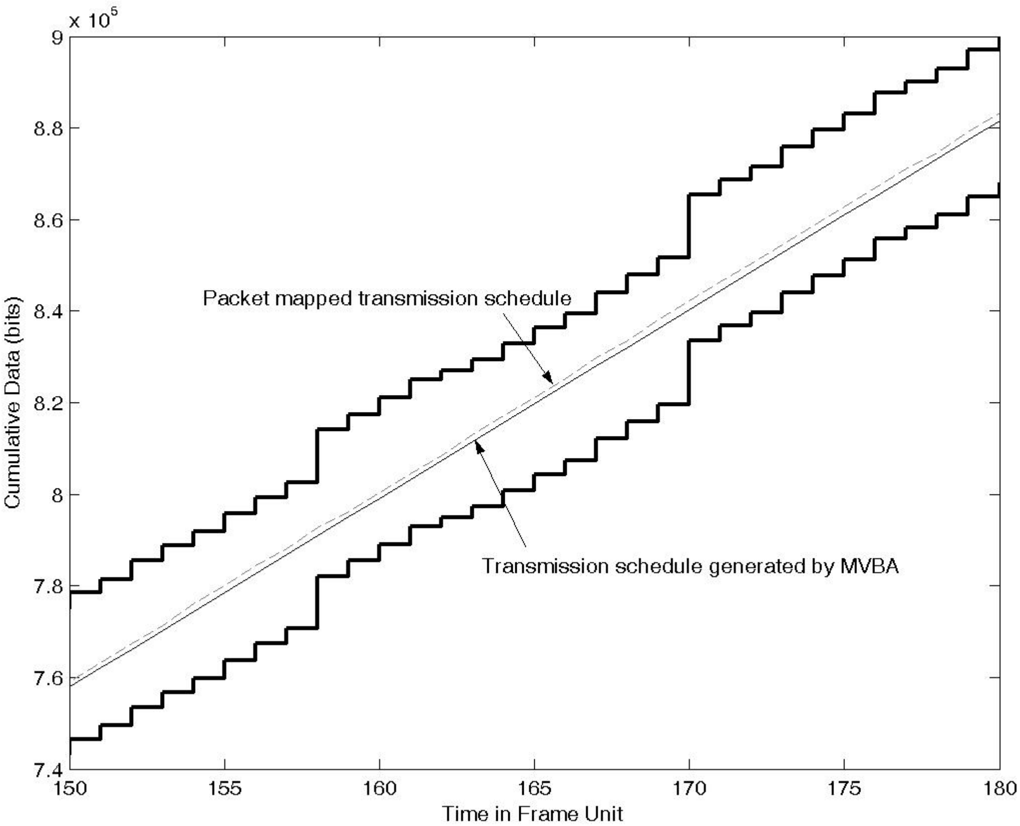



Figure 33: Transmission schedule generated by MVBA and by Packet Mapped Smoothing algorithm with 32K bits client buffer and 1024 bits maximum packet size under $s_{\min_{a,b}}$ condition. Where the solid line represents the transmission schedule of MVBA, and the dash line represents the packet mapped transmission schedule.

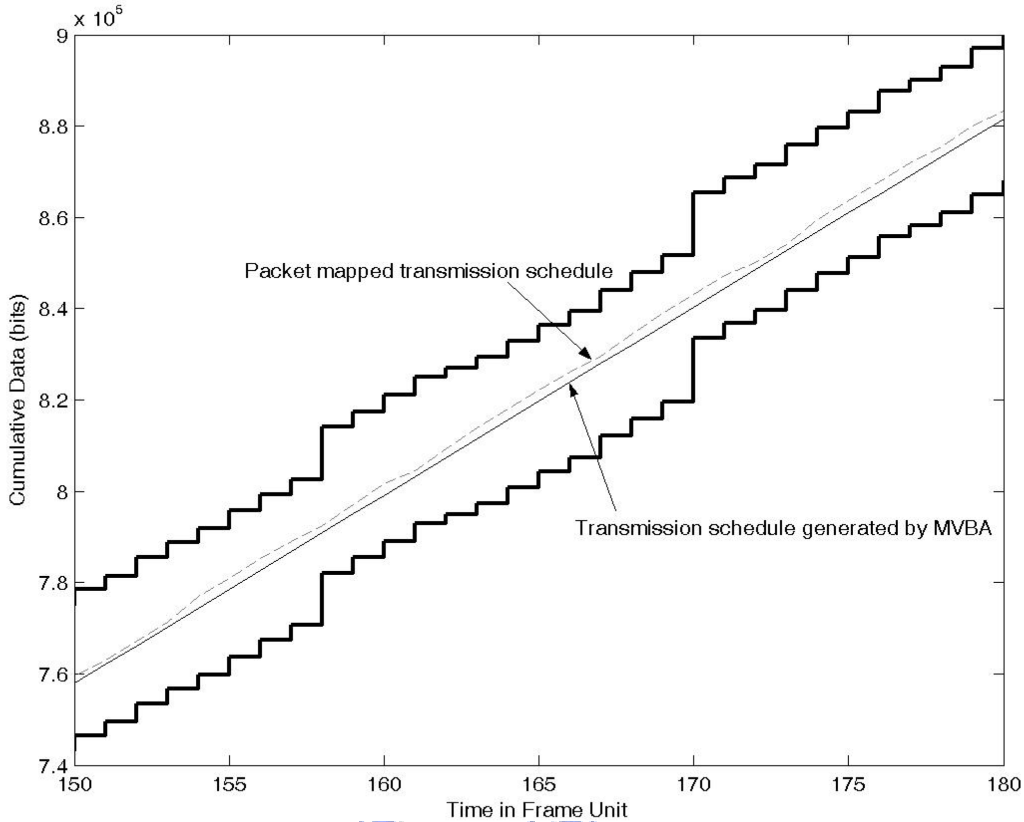



Figure 34: Transmission schedule generated by MVBA and by Packet Mapped Smoothing algorithm with 32K bits client buffer and 2048 bits maximum packet size under $s_{\min_{a,b}}$ condition. Where the solid line represents the transmission schedule of MVBA, and the dash line represents the packet mapped transmission schedule.

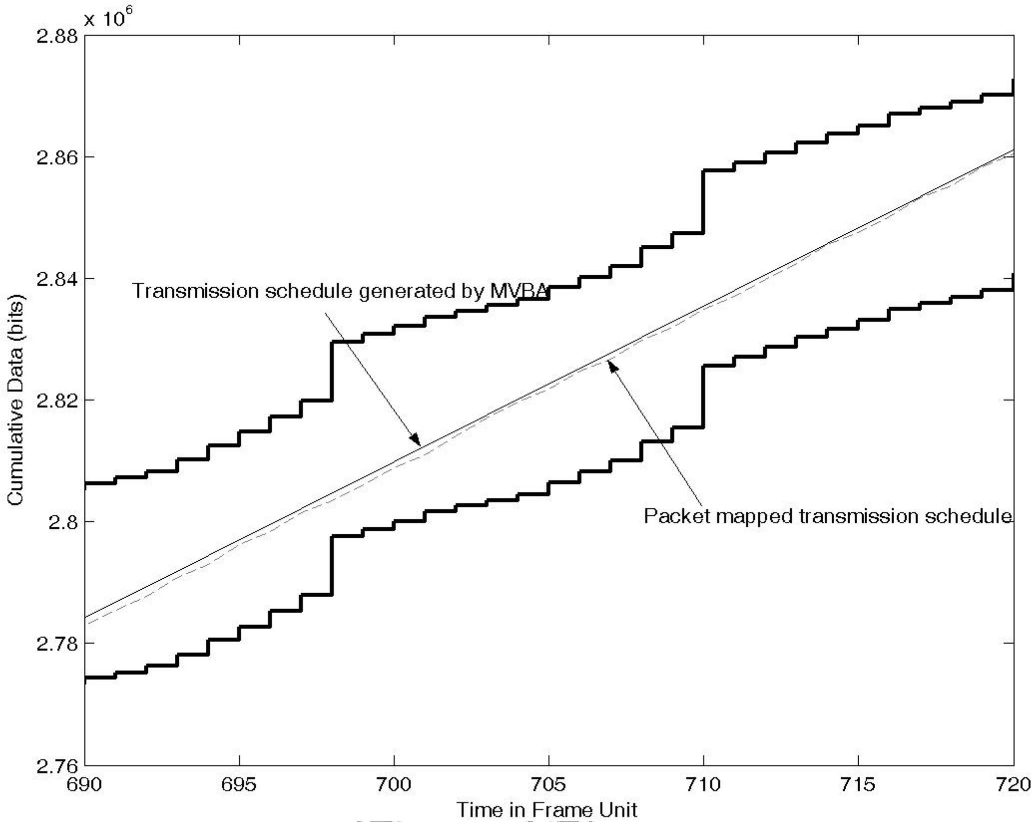



Figure 35: Transmission schedule generated by MVBA and by Packet Mapped Smoothing algorithm with 32K bits client buffer and 1024 bits maximum packet size under $s_{\max_{a,b}}$ condition. Where the solid line represents the transmission schedule of MVBA, and the dash line represents the packet mapped transmission schedule.

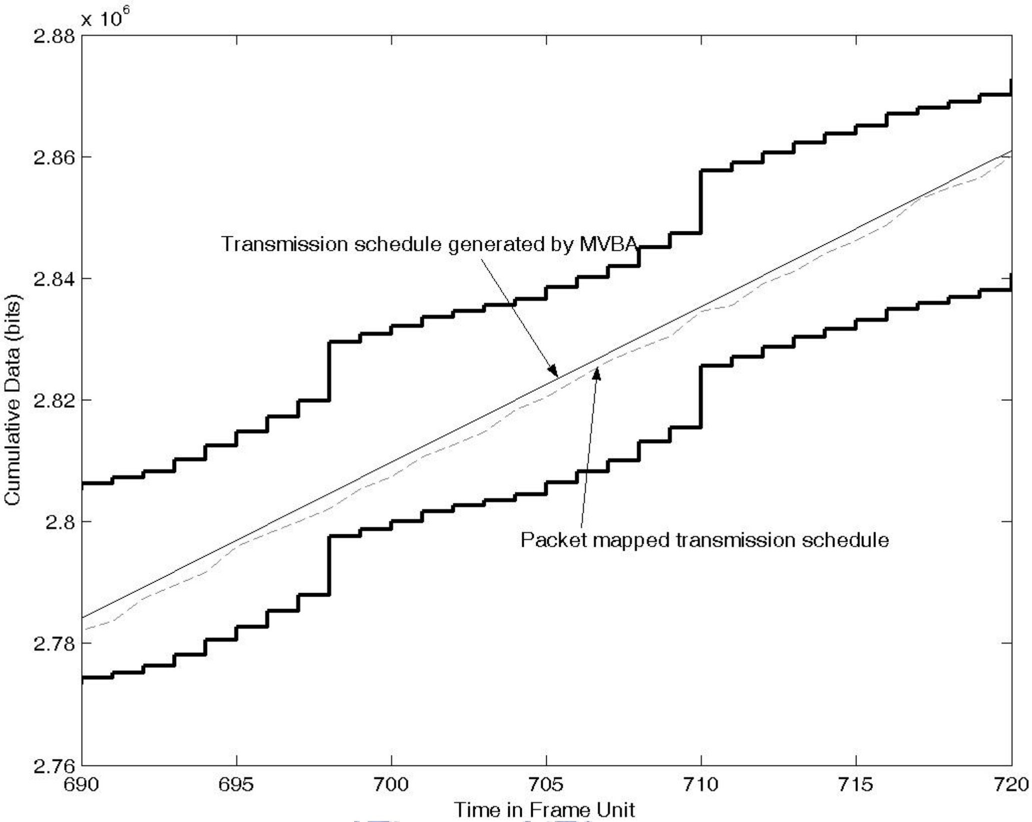



Figure 36: Transmission schedule generated by MVBA and by Packet Mapped Smoothing algorithm with 32K bits client buffer and 2048 bits maximum packet size under $s_{\max_{a,b}}$ condition. Where the solid line represents the transmission schedule of MVBA, and the dash line represents the packet mapped transmission schedule.

Then, the amount of cumulative data until the corresponding packet number at each frame slot is the actual transmission schedule of Packet Mapped Smoothing Algorithm. According to the above four figures, from Fig. 33 to Fig. 36, it's apparent that the large maximum packet size will induce more error between transmission schedules of MVBA and of Packet Mapped smoothing algorithm.

The following Fig. 37 and Fig. 38 are the transmission rate of MVBA and of

Packet Mapped smoothing algorithm of the first 1000 frame of “Aladdin” with 32K bits client buffer and 1024 bits and 2048 bits maximum packet size respectively. Similarly, Fig. 39 and Fig. 40 are the transmission rate of MVBA and of packet smoothing algorithm of total video “Aladdin” with 100K bits client buffer and 1024 bits and 2048 bits maximum packet size respectively. Fig. 41 and Fig. 42 are the transmission rate of MVBA and of packet smoothing algorithm of total video “Aladdin” with 1M bits client buffer and 1024 bits and 2048 bits maximum packet size respectively. It’s obvious that the transmission rate of Packet Mapped smoothing algorithm vibrates along the transmission rate of MVBA, and the amplitude of the vibration is about the maximum packet size. Because Packet Mapped smoothing algorithm approximates the transmission schedule to MVBA optimum schedule as near as possible, the maximum difference between real transmission rate and rate of MVBA optimum schedule can only be less than one maximum packet size. Therefore, the larger the maximum packet size is, the larger the variation in rate can be.

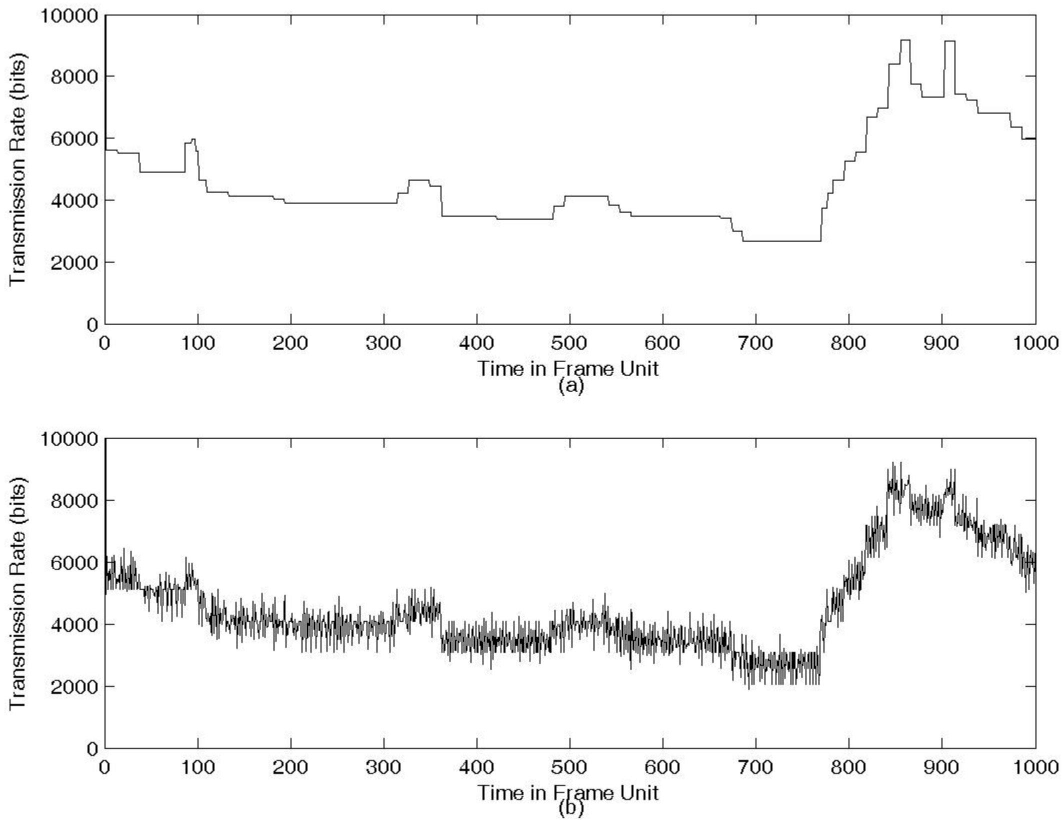



Figure 37: The transmission rate of first 1000 frames of “Aladdin” generated by Packet Mapped smoothing algorithm with 32K bits client buffer and 1024 bits maximum packet size. (a) is the transmission rate of MVBA. (b) is the transmission rate of Packet Mapped smoothing algorithm.

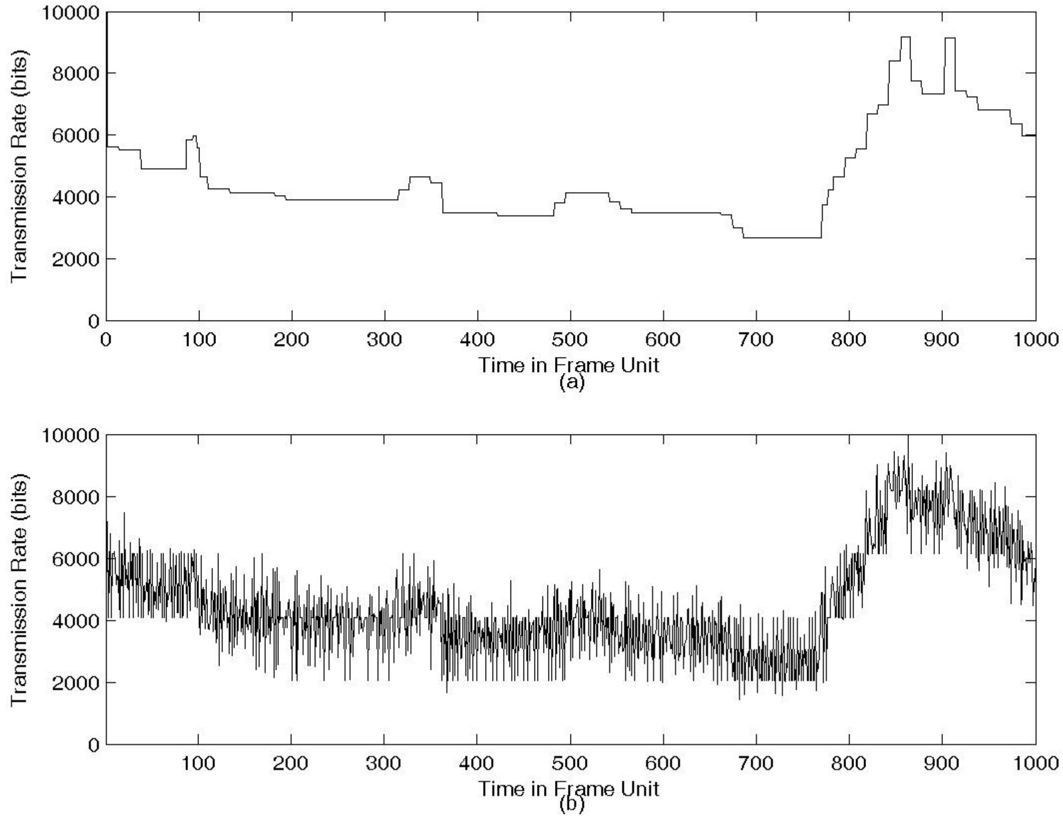



Figure 38: The transmission rate of first 1000 frames of “Aladdin” generated by Packet Mapped smoothing algorithm with 32K bits client buffer and 2048 bits maximum packet size. (a) is the transmission rate of MVBA. (b) is the transmission rate of Packet Mapped smoothing algorithm.

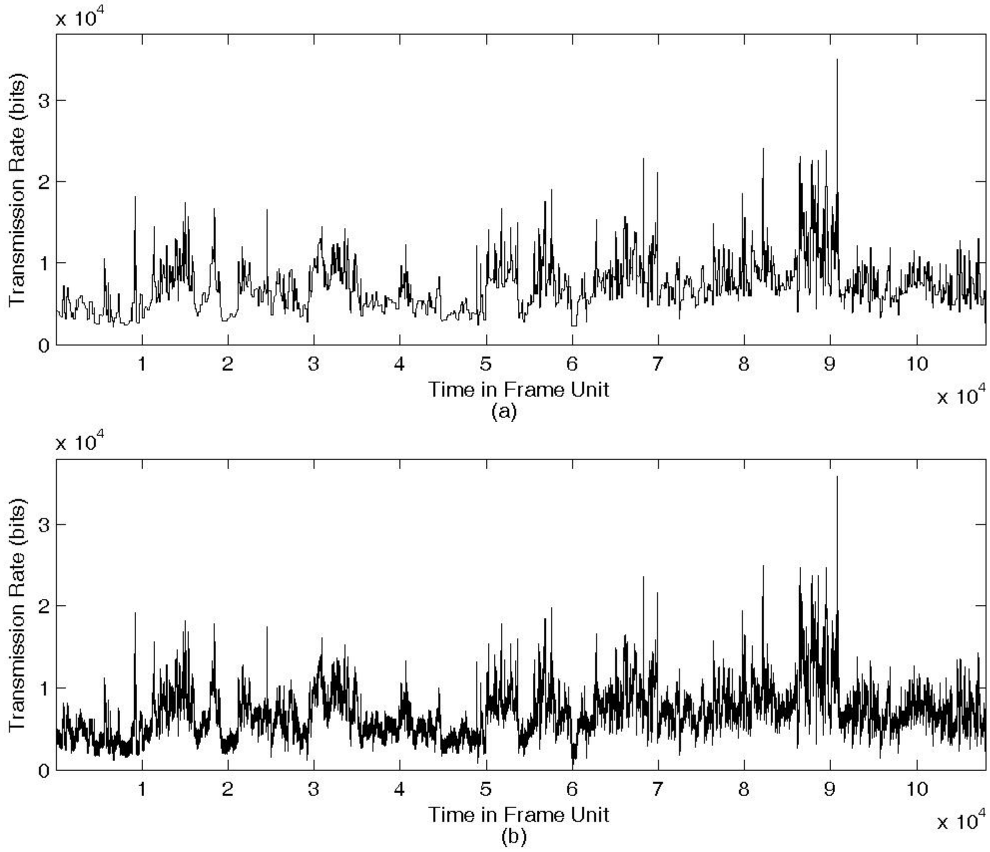



Figure 39: The transmission rate of total video of "Aladdin" generated by Packet Mapped smoothing algorithm with 100K bits client buffer and 1024 bits maximum packet size. (a) is the transmission rate of MVBA. (b) is the transmission rate of Packet Mapped smoothing algorithm.

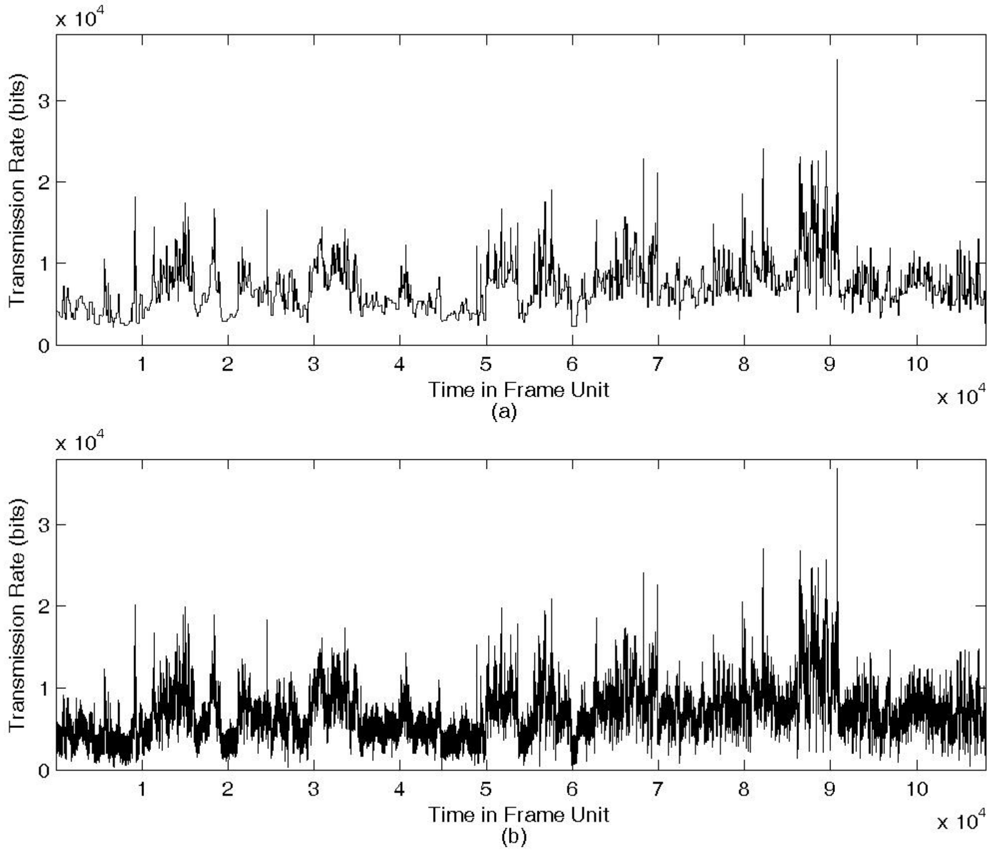



Figure 40: The transmission rate of total video of “Aladdin” generated by Packet Mapped smoothing algorithm with 100K bits client buffer and 2048 bits maximum packet size. (a) is the transmission rate of MVBA. (b) is the transmission rate of Packet Mapped smoothing algorithm.

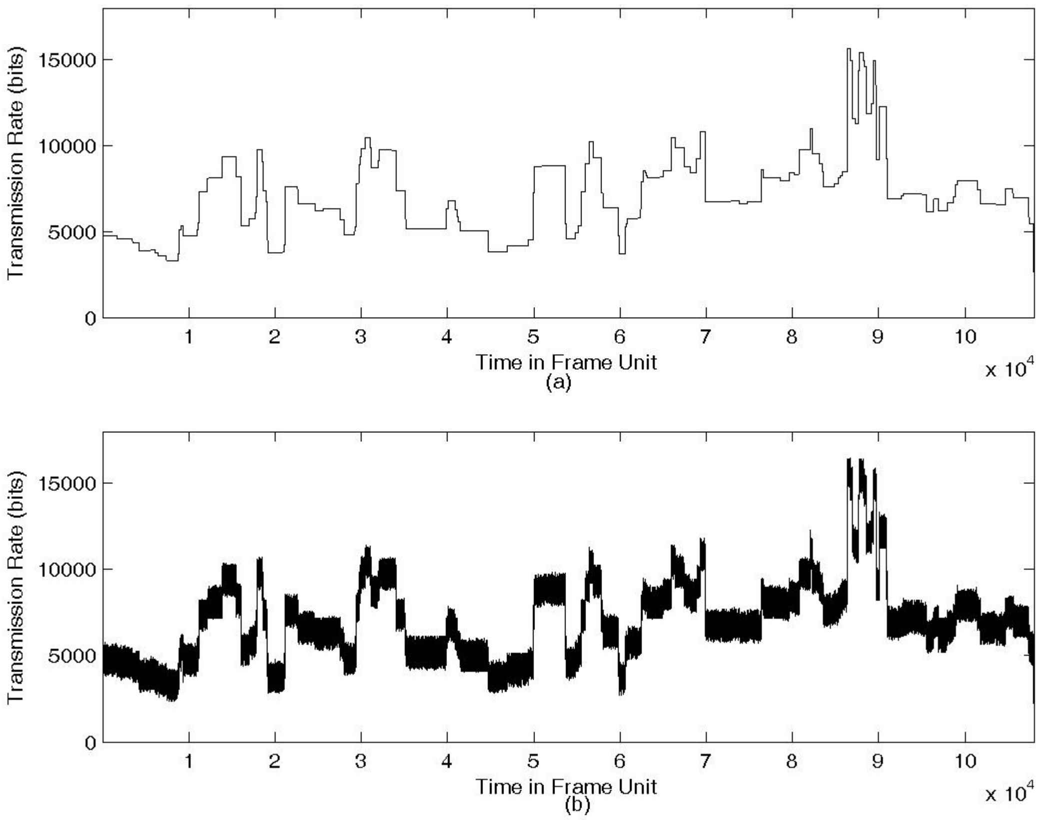



Figure 41: The transmission rate of the total video “Aladdin” generated by Packet Mapped smoothing algorithm with 1M bits client buffer and 1024 bits maximum packet size. (a) is the transmission rate of MVBA. (b) is the transmission rate of Packet Mapped smoothing algorithm.

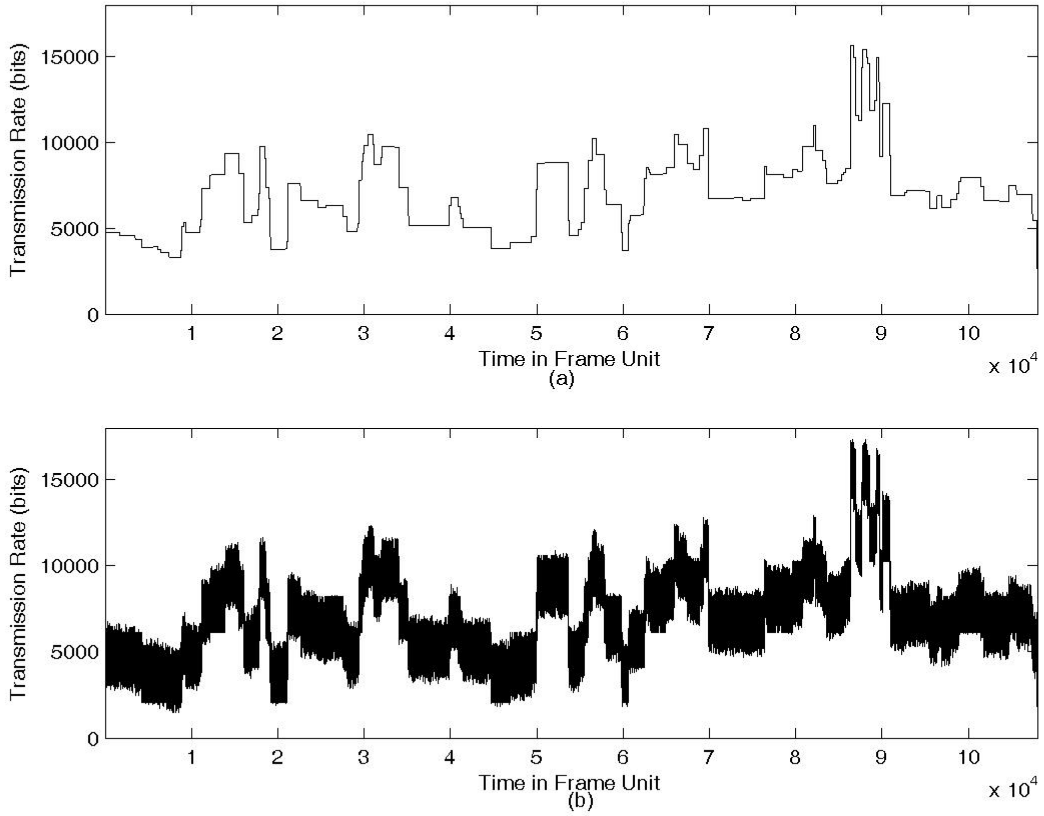



Figure 42: The transmission rate of first 1000 frames of “Aladdin” generated by Packet Mapped smoothing algorithm with 1M bits client buffer and 2048 bits maximum packet size. (a) is the transmission rate of MVBA. (b) is the transmission rate of Packet Mapped smoothing algorithm.

According to the above examples, from Fig. 37 to Fig. 42, it shows that the variance in transmission rate is influenced by two factors, the variation in rate of the transmission schedule generated by MVBA and the variation in the succession packet sizes. It’s unlike the Number-of-Packet Based Smoothing algorithm, the client buffer only influences the variation in transmission rate of MVBA, and the maximum packet size only influences the variation in succession packet sizes. Thus, the larger client buffer results in lower variance in transmission rate, and the smaller maximum packet size induces the lower variation in succession packet sizes. Consequently, the performance of transmission schedule generated by Packet Mapped Smoothing algorithm with larger client buffer and smaller maximum packet size is better.

Fig. 43 represents the variance in transmission rate generated by packet mapped smoothing algorithm as a function of client buffer size with different maximum packet sizes. According to Fig. 43, when the maximum packet size is smaller, the variance in transmission will go down as the client buffer size increases. But it is also influenced by the variation in successive packets. The curve with cross mark represents the variance in transmission rate with maximum packet size 512 bits. This curve quite closes to the curve of MVBA which is the curve without mark. As the maximum packet size increases, the ripple of the curve will intensify, and the variance itself will increase because the larger maximum packet size induces more variation in successive packets. When the maximum packet size is 1024 bits and 1536 bits, the difference of variance between packet mapped smoothing algorithm and MVBA is increasing, and the ripple of the curve becomes intensive when the maximum packet size is 2048 bits. Because when the maximum packet size is too large, the variance caused by the variation of successions packet size may greater than the reduction of variance resulted by the client buffer increasing.

As we mentioned before, the transmission schedule generated by packet mapped smoothing algorithm will vibrate along the MVBA optimum transmission schedule, and the amplitude of vibration equals to maximum packet size. Thus, the curve of peak rate of packet mapped smoothing algorithm is vertically shifted version of curve of MVBA at a distance of maximum packet size. Fig. 44 illustrates this result.

Similarly, the BFF, which is an inverse ration of the peak rate will increase steadily according to the decreasing of peak rate as the client buffer size increases. It is shown in Fig. 45.

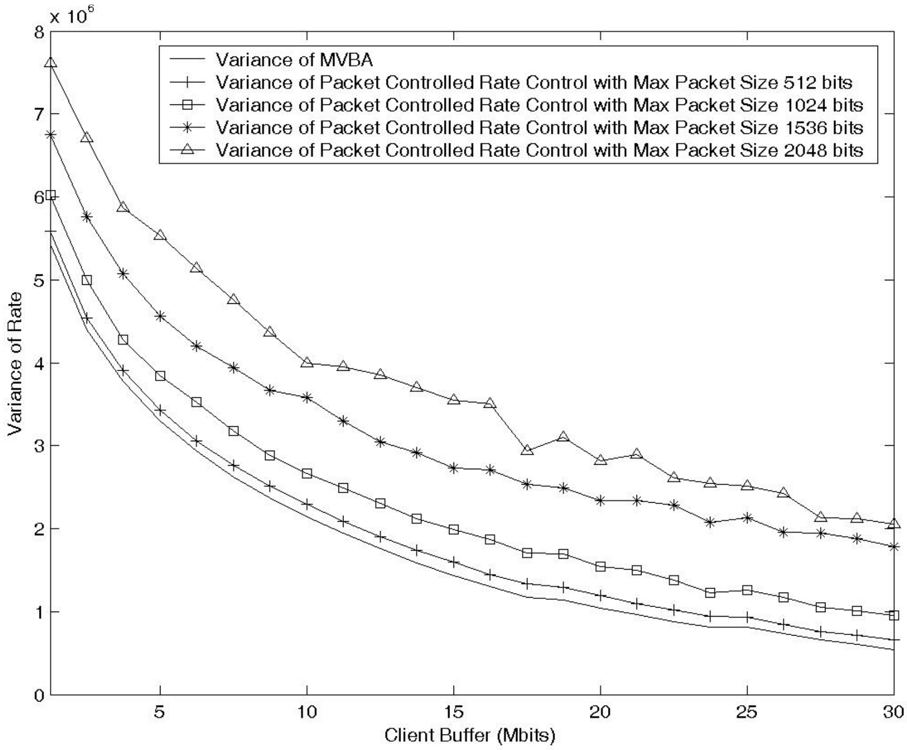



Figure 43: This graph illustrates the variance in “Aladdin” transmission rate as a function of client buffer size with different maximum packet sizes.

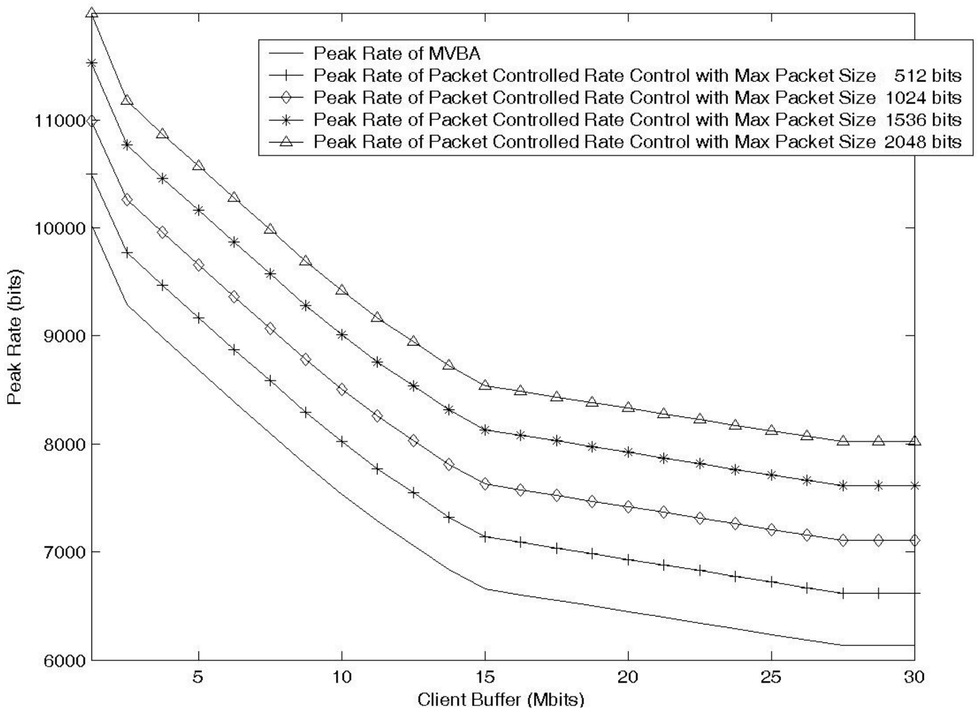



Figure 44: This graph illustrates the peak rate of “Aladdin” transmission schedule as a function of client buffer size with different maximum packet sizes.

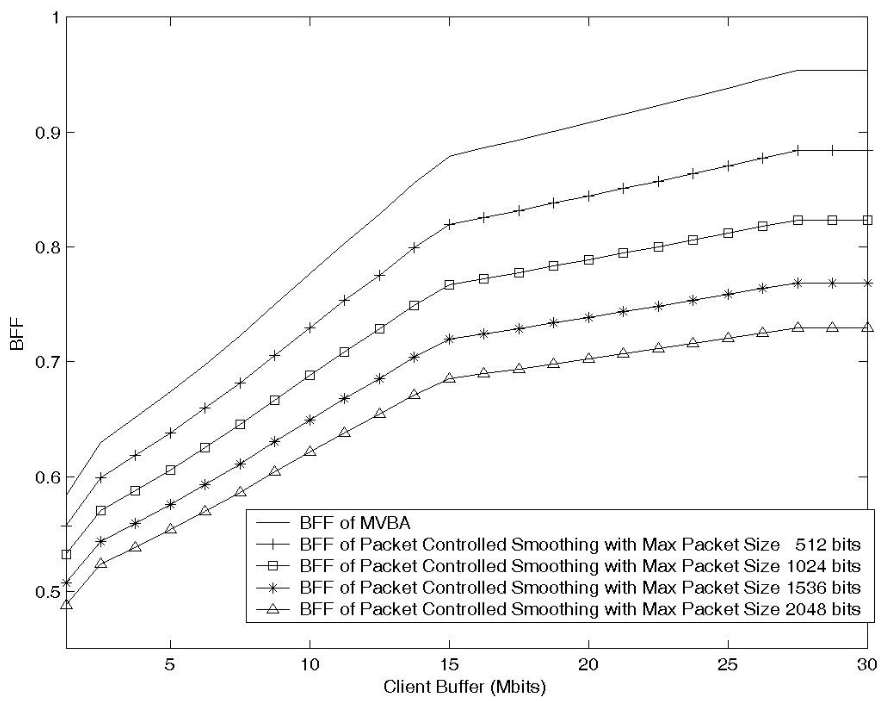



Figure 45: This graph illustrates BFF of “Aladdin” as the function of client buffer size with different maximum packet sizes.

## 5.3 Analysis of Simulation Result

Number-of-Packet Based Smoothing Algorithm makes the number of packet transmitted at each frame slot steadier. Therefore, when server tries to transmit some data on to the network, the media access contention is steadier, and is easier to management. But the transmission rate will vary as the variation between succession packet sizes. For example, the schedule of a certain time interval of 2 frame slots is that 3 packets each frame slot, and the packet sizes are 1024, 1024,1024,128,128 and 128 respectively. Thus, the transmission rate at this two frame slot are 3072 and 284 respectively. It's obviously that although the transmitted numbers of packets are the same, the rate may be very different. The mathematical representation of transmission rate is as following:

$$Rate_i = \sum_{j \in i} Packet\_Size_j \tag{5-1}$$

where $Rate_i$ is the transmission rate of $i_{th}$ frame slot, and j is the index of packet number. Then $Rate_i$ equals to the summation of all packet sizes that are transmitted at $i_{th}$ frame slot. In our video transmission system, each packet sizes can be consider as a random variable, distributed in the interval from 0 to maximum packet size. Thus, the transmission rate of Number-of-Packet Based Smoothing Algorithm is considered as the summation of several random variables. Therefore, the rate variation of each CBR segments can be considered as the variation of the summation of random variables, and the more the packet rate of this CBR segment is, the greater the variation is.

The mapped error in Packet Mapped Smoothing Algorithm, which is defined as the difference between the actual transmission rate and the rate of MVBA, can be considered as a random variable. This random variable is distributed in the interval from 0 to maximum packet size. Therefore, the rate can be represented as following:

$$Rate_i = Rate\_of\_MVBA_i + Error_i \tag{5-2}$$

The rate variation of each CBR segments is only resulted by the mapped error.

Therefore, in each CBR segments, the rate variation of Packet mapped Smoothing Algorithm is much smaller than the rate variation of Number-of-Packet Based Smoothing Algorithm.

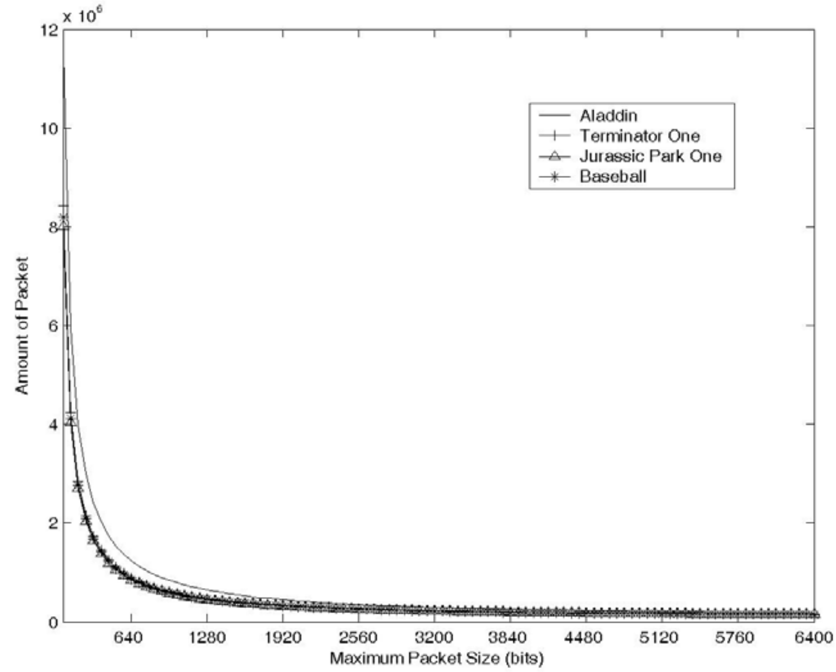
## 5.4 Influence of Maximum Packet Sizes

According to the simulation result shown previously, the smaller maximum packet size induces better performance. But smaller maximum packet size also results in more packets. A large amount of packets will increase the collision probability resulted by network media contention. Now we show the relation between the number of packets and maximum packet size.

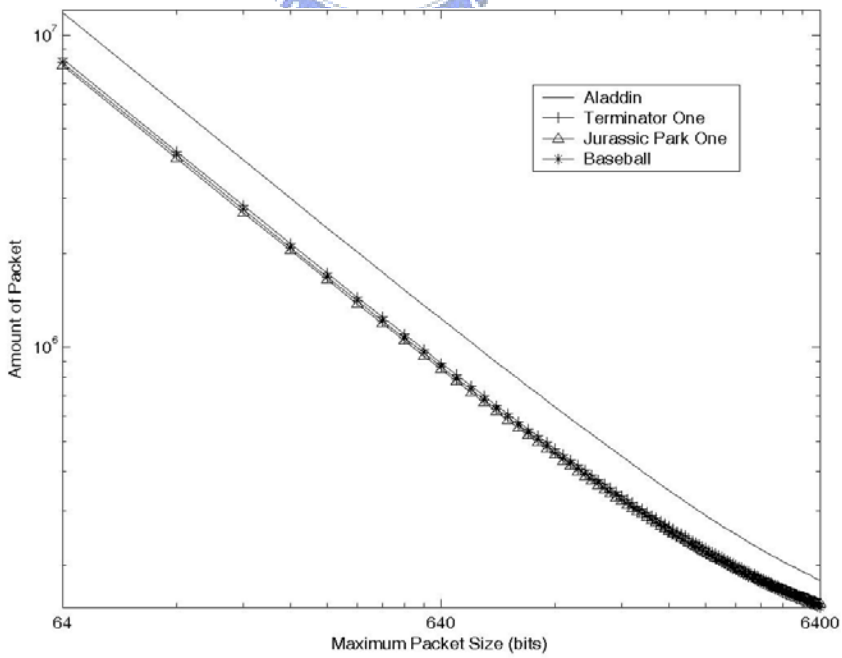
Fig. 46 is the total number of packets as the function of maximum packet size. (a) is shown in linear format. When maximum packet size is greater than about 2000 bits, the number of packets is almost constant. (b) is shown in logarithmic format. It shows that total number of packets has the exponential relation with maximum packet size.

Fig. 47 is the total number of packets as the function of maximum packet size to mean rate ratio. (a) is shown in linear format. (b) is shown in logarithmic format.

According to the two figures, they show that the four curves are almost the same. Thus, we predict that the number of packets is inverse ration to the exponent of maximum packet size for all video.

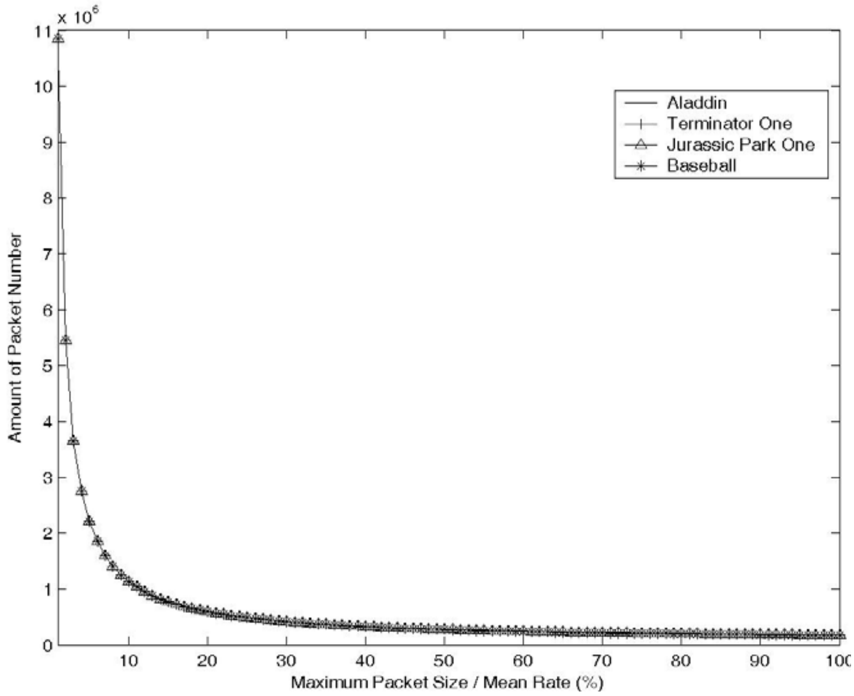



(a)

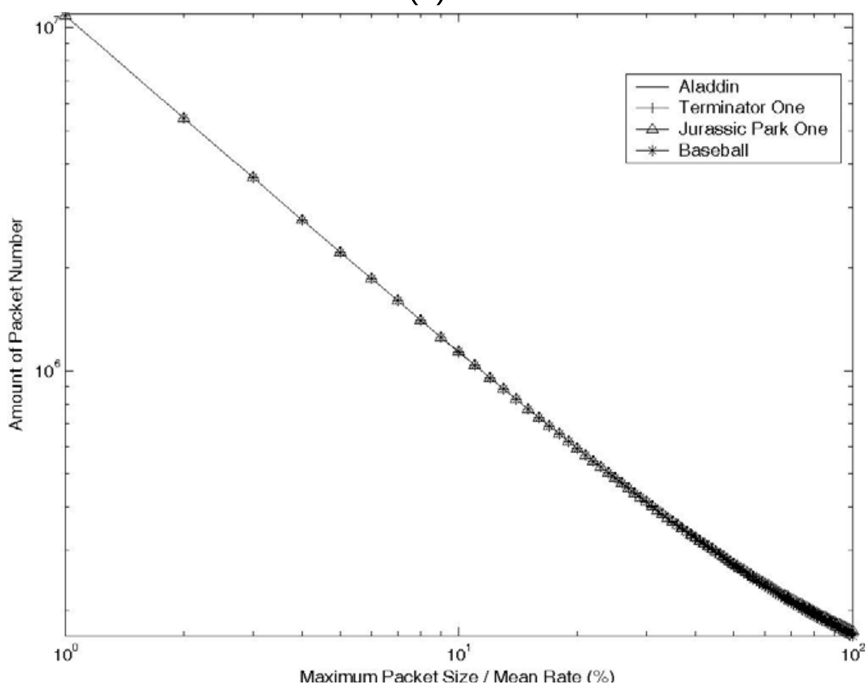



(b)

Figure 46: These figures illustrate the total number of packets as a function of maximum packet size. (a) is shown in linear forma. (b) is shown in logarithmic format.




(a)




(b)

Figure 47: These figures illustrate the total number of packets as the function of maximum packet size to mean rate ratio. (a) is shown in linear format. (b) is shown in logarithmic format.

The following Fig. 48 and Fig. 49 demonstrate the variance of packet size with different maximum packet sizes. Fig. 48 is the variance of all packets, whether it is a full packet or not. Fig. 49 is the variance of the small packets which are not full. These two figures show that when the maximum packet size increase, the variance of packet size also increase.

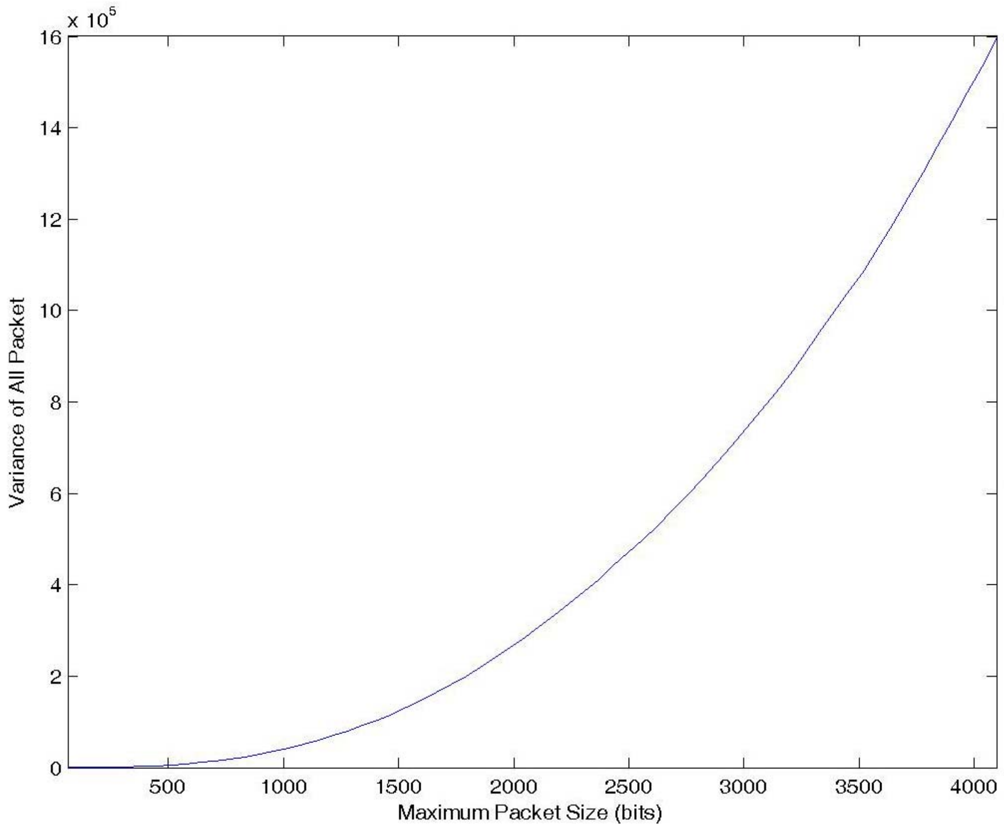



Figure 48: The variance of all packet size.

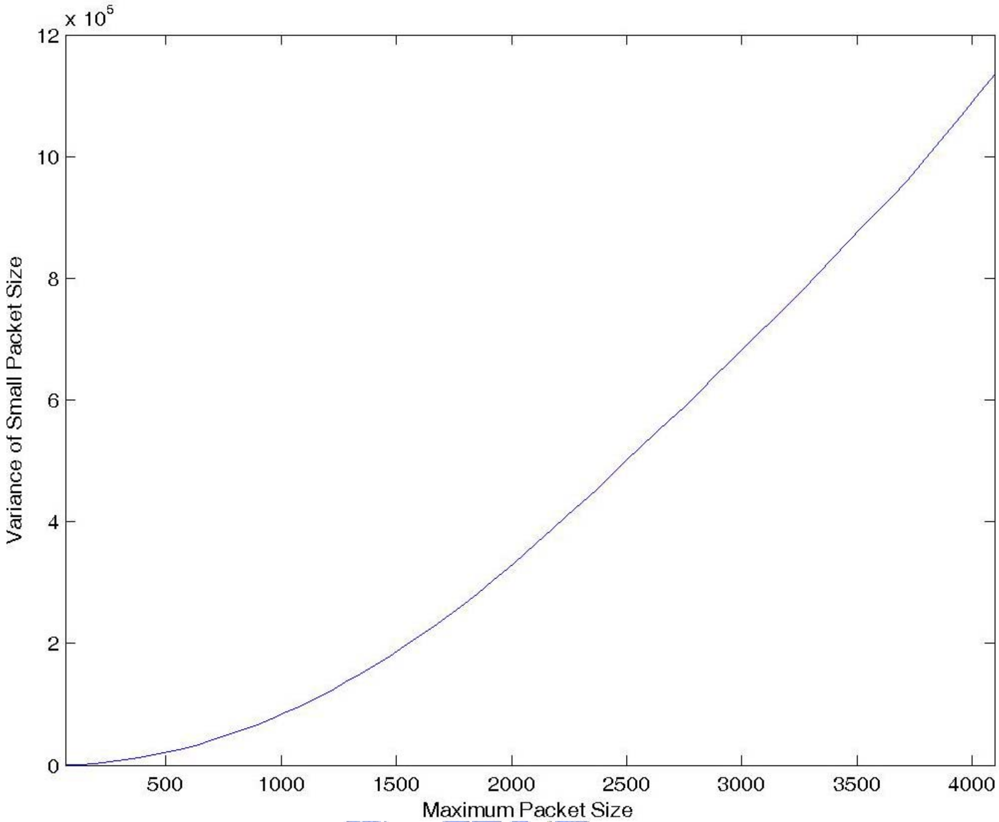



Figure 49: The variance of small packet size, which is not full.



Next, we show the variance of rate difference between the MVBA schedule and Packet Mapped smoothing algorithm. The transmission rate of Packet Mapped smoothing algorithm is like the rate of MVBA schedule added a random noise. Fig. 50 shows the variance of this "noise" with different maximum packet sizes.

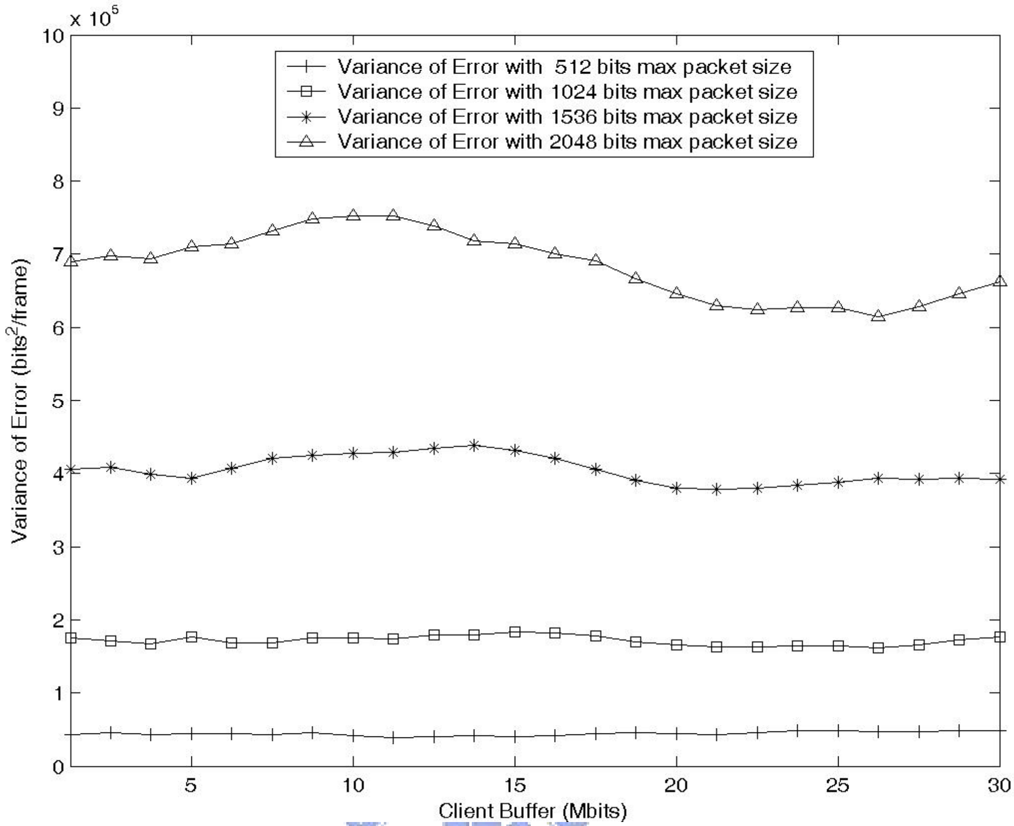



Figure 50: The variance of rate difference between MVBA schedule and Packet Mapped Smoothing Algorithm.

This figure shows that when the maximum packet size increase, the intensity of the “noise” also increase. The client buffer almost doesn’t influence this noise when the maximum packet size is small. As the maximum packet size increase, the variance is different as the client buffer increasing. It’s hard to find the relationship between maximum packet size and the variation of the variance of “noise”. Compare Fig. 48 and Fig. 49, we can’t find any relation between the two parameters.

## 5.4 Implementation Result

In this section, we demonstrate the network utilization phenomenon of the video streaming system with and without smoothing algorithm. We insert the Packet Mapped smoothing algorithm into our video streaming algorithm with the method described in Section 4.3, and use the external program, winpcap, to record each packet size and its corresponding transmitted(received) time in server(client). The record method is as following. In the server terminal, when the first packet is send to the network, winpcap will record its size and mark the time as 0. Then when the following packets are transmitted, winpcap will record their size and the time that how long this packet is transmitted later than the departure of the first packet. In the client terminal, the method is similar, when the first packet arrival, winpcap records its size and the arrival time is 0. Then the arrival time of the following packets are the time difference between each packets and the first packet.

We use the video file, cindy.mp4, which is compressed with MPEG4, and the frame rate is 25 frames per second, for experiment. Fig. 51 is the example of the transmission record. The server transmits data according to the schedule generated by Packet Mapped smoothing algorithm with 1024 byte maximum packet size and 128K byte client buffer. Fig. 51(a) is the record transmitted packet in server from 9.9 sec to 10.1 sec. Fig. 51(b) is the record of arrival packet in client. Each stems represent a packet, the position of horizontal axis represents the recorded time, and the length of each stems is the packet sizes. This figure shows that the packet will transmitted group by group with an interval of about 40 ms, and the packet groups will arrival at almost the same time interval.

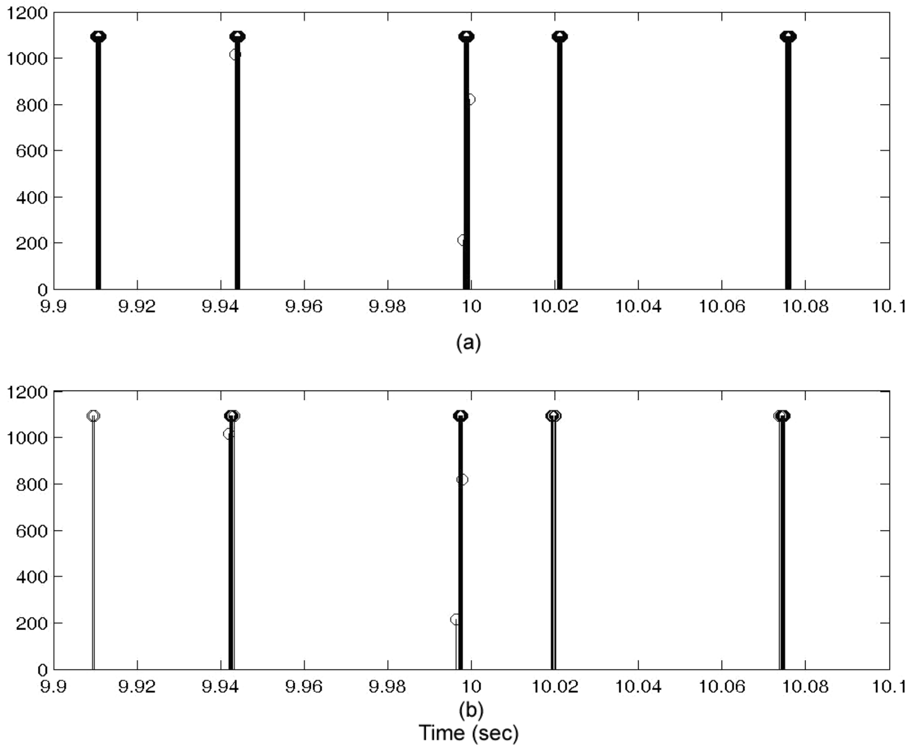



Figure 51: The record of the packet size and time. (a) is the transmitted packets in server. (b) is the received packets in client.

In Fig. 51, we only can see that each packet groups is transmitted and received in a very short period of about 1 ms. Fig. 52 is the record from 9.941 sec to 9.945 sec. There is only one packet group in this time interval. The server transmits 8 packets in about 1 ms, the packet interval is about 150 micro sec. The client receives these packets also in about 1 ms, but the packet interval is different to the interval in server. The transmission network jitter results in this variation. The group position is not the same, because of the error in our experiment. The time recorded in server will be larger than the time recorded in client. For every 100 sec, the error is about 10 ms.

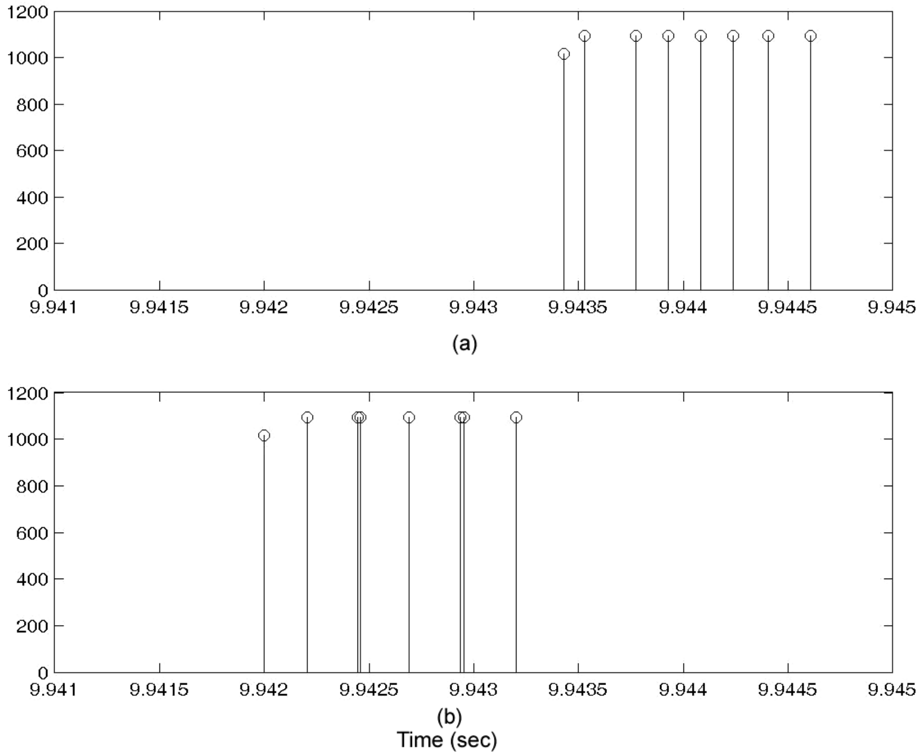



Figure 52: The record of the packet size and time observed in detail. (a) is the transmitted packets in server. (b) is the received packets in client.

We experiment on different client buffer, different maximum packet size and different transmission distance, and then gather the statistics of the packet interval and group interval to observe what smoothing algorithm can do in video streaming system.

Fig. 53 is the record from the server in NCTU to the client in CCU. The transmission schedule is generated by Packet Mapped smoothing algorithm with 1024 bytes maximum packet size and 128K bytes client buffer. (a) is the record in server, and (b) is the record in client. This record shows that the packet group will arrive in a longer period, the time interval of each packet in a group is with more variation, and some of the time interval is elongated because the network jitter results in the different transmission delay. We show this phenomenon in detail in Fig. 54, which is the same record in detail.

Fig. 55 and Fig. 56 is the transmission record of an unsmoothed schedule. The server transmits all of the frame data at each corresponding frame slot. We can see that when the server transmits with an unsmoothed schedule, the number of packets in each group is with more variation. The server accesses the network for a longer period, and the network jitter is more critical.

Fig. 57 and Fig. 58 is the transmission record of an unsmoothed schedule and from NCTU to CCU. As we mention above, the server may transmits too many packet in a frame slot and the time interval of each packet may be elongated for long distance transmission. These may result in some packet arrive the client too late to be played on time.

According to the record of time and packet size, we can see that the server with smoothing algorithm can transmit data in a more stable state; the number of packet in each group and the network access time is almost constant. Therefore, even for long distance transmission the network jitter won't result in critical effect.

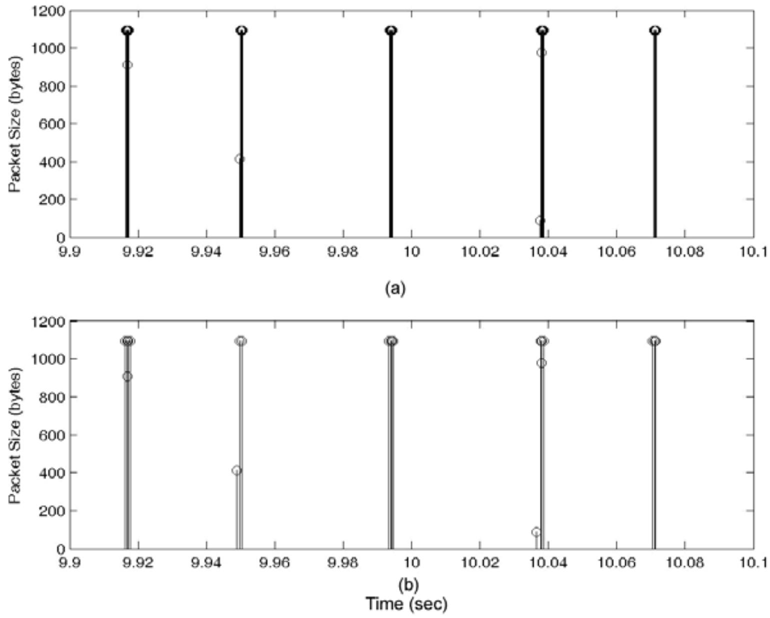



Figure 53: The record of the packet size and time form NCTU to CCU. (a) is the transmitted packets in server. (b) is the received packets in client.

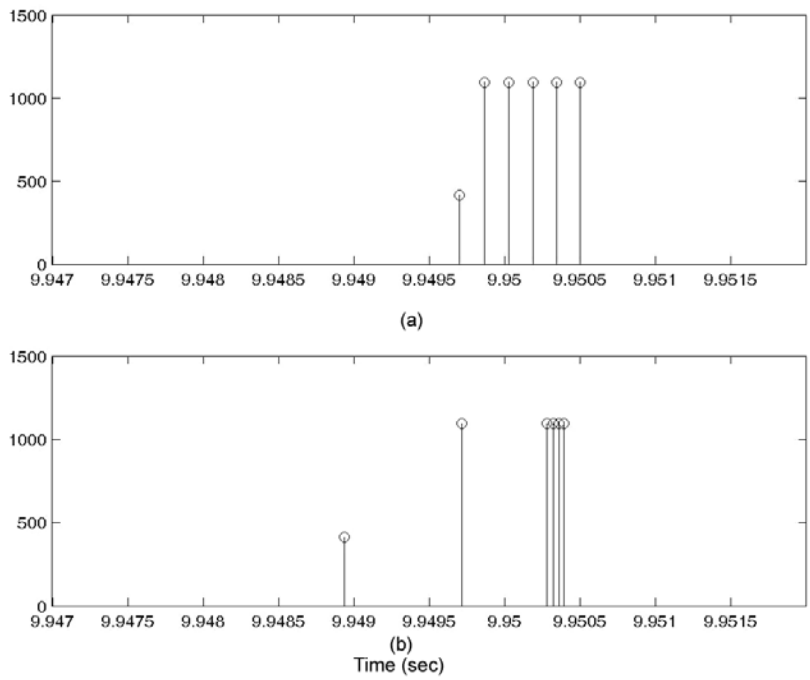



Figure 54: The record of the packet size and time form NCTU to CCU in detail. (a) is the transmitted packets in server. (b) is the received packets in client.

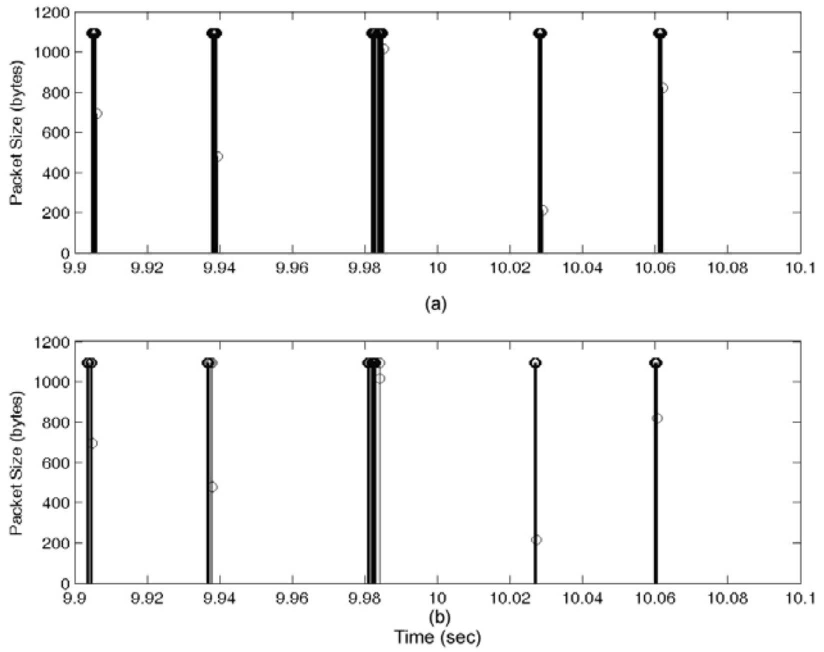



Figure 55: The record of the packet size and time with unsmoothed schedule. (a) is the transmitted packets in server. (b) is the received packets in client.

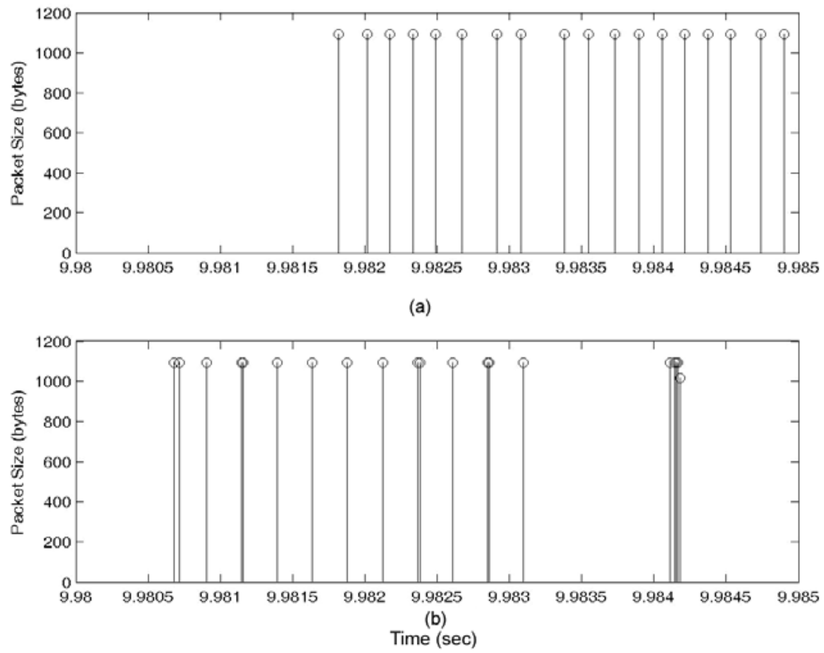



Figure 56: The record of the packet size and time with unsmoothed schedule in detail. (a) is the transmitted packets in server. (b) is the received packets in client.

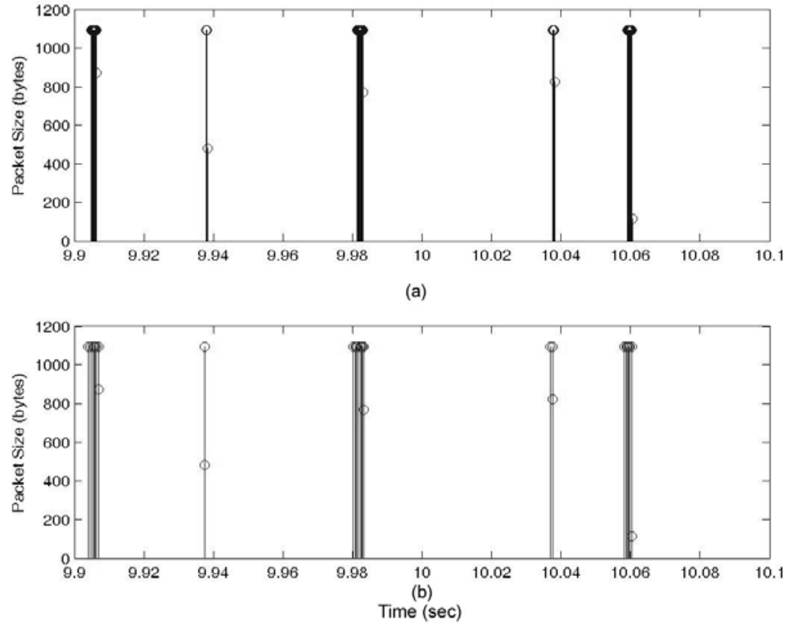



Figure 57: The record of the packet size and time with unsmoothed schedule from NCTU to CCU. (a) is the transmitted packets in server. (b) is the received packets in client.

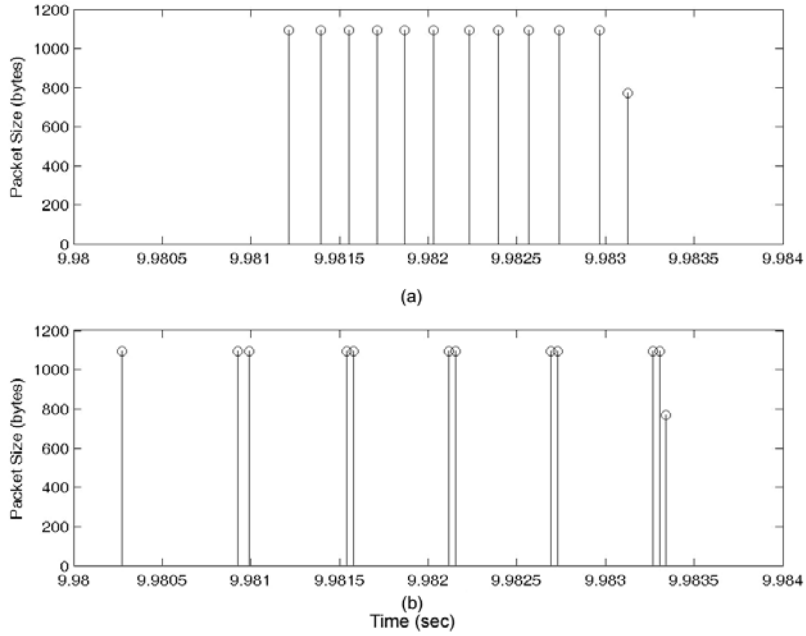



Figure 58: The record of the packet size and time with unsmoothed schedule from NCTU to CCU in detail. (a) is the transmitted packets in server. (b) is the received packets in client.

Next, we show the histogram of the group interval and packet interval to observe the transmission jitter and network jitter.

Fig. 59 to Fig. 62 are the histograms of group interval. Fig. 59 is with smoothed schedule from NCTU to NCTU, Fig. 60 is with smoothed schedule from NCTU to CCU, Fig. 61 is with unsmoothed schedule form NCTU to NCTU and Fig. 62 is with unsmoothed schedule from NCTU to CCU. In our algorithm, we expect that the server transmits data once every 40 ms. However, these histogram in shows that there are errors of about 5 ms to 8 ms in transmission time. Thus a large part of group interval is about 32 ms and 45 ms. Then, we see the histogram in client will spread at the around the peak in server. For long distance transmission, the spread range is larger. The schedule with or without smoothing won't influence the transmission jitter for one client.

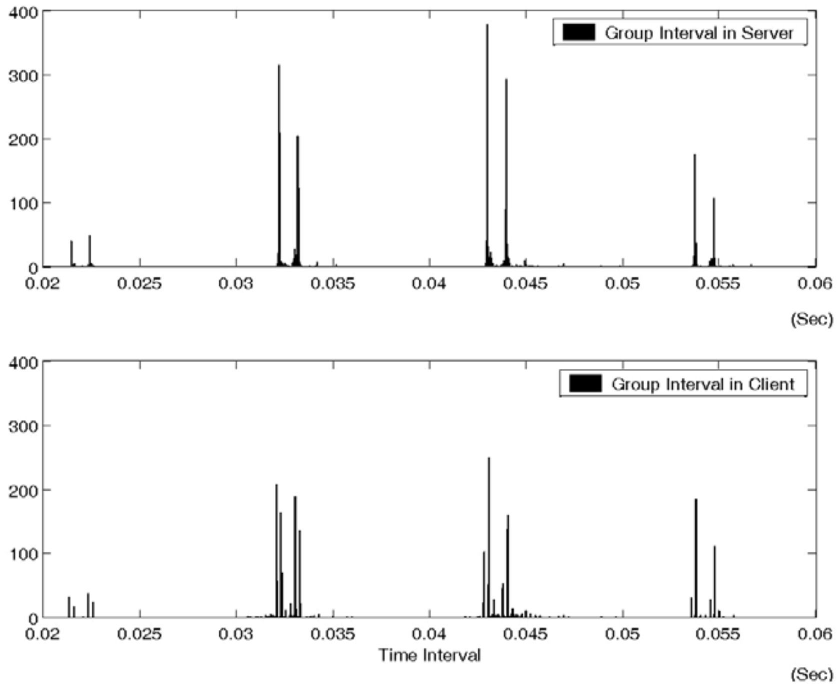



Figure 59: Histogram of group interval with smoothed schedule form NCTU to NCTU.

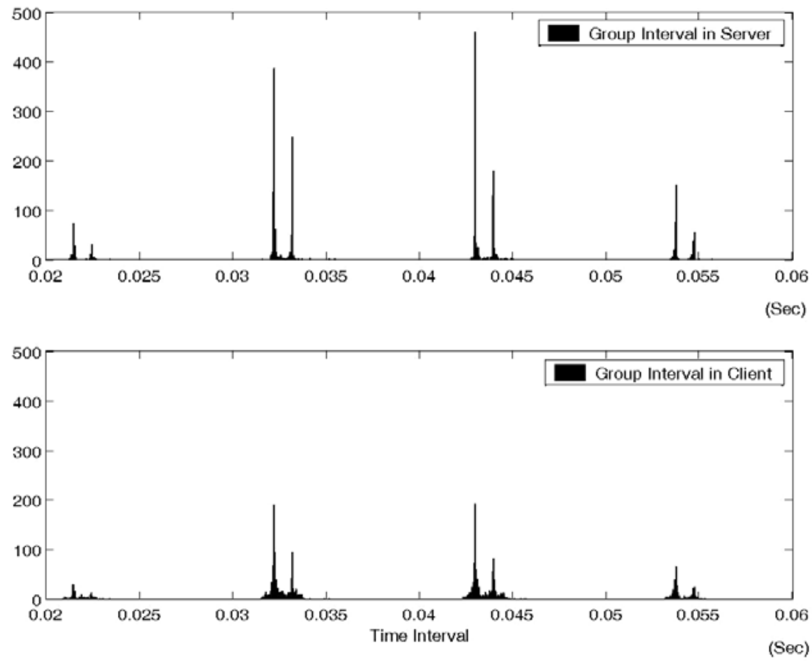



Figure 60: Histogram of group interval with smoothed schedule form NCTU to CCU.

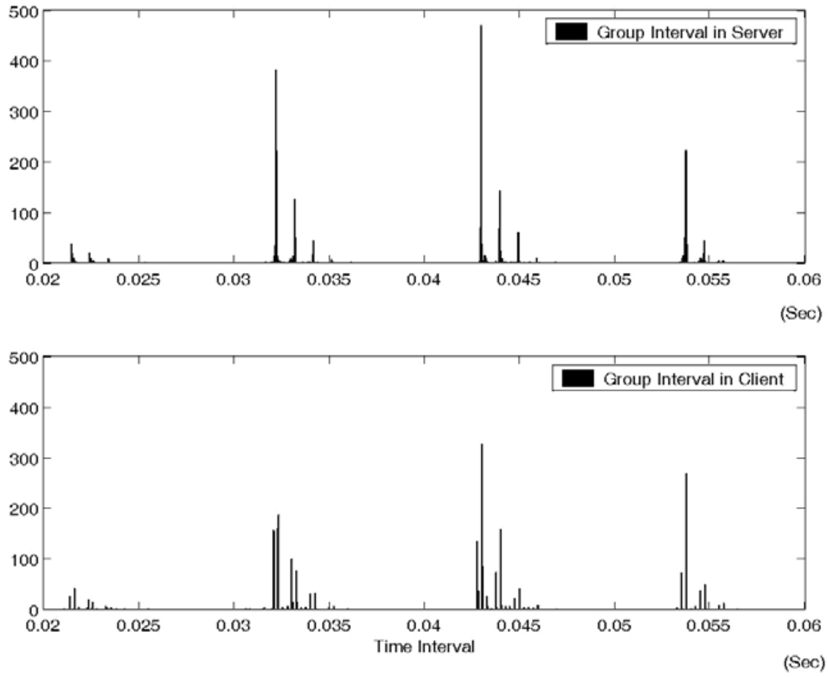



Figure 61: Histogram of group interval with unsmoothed schedule form NCTU to NCTU

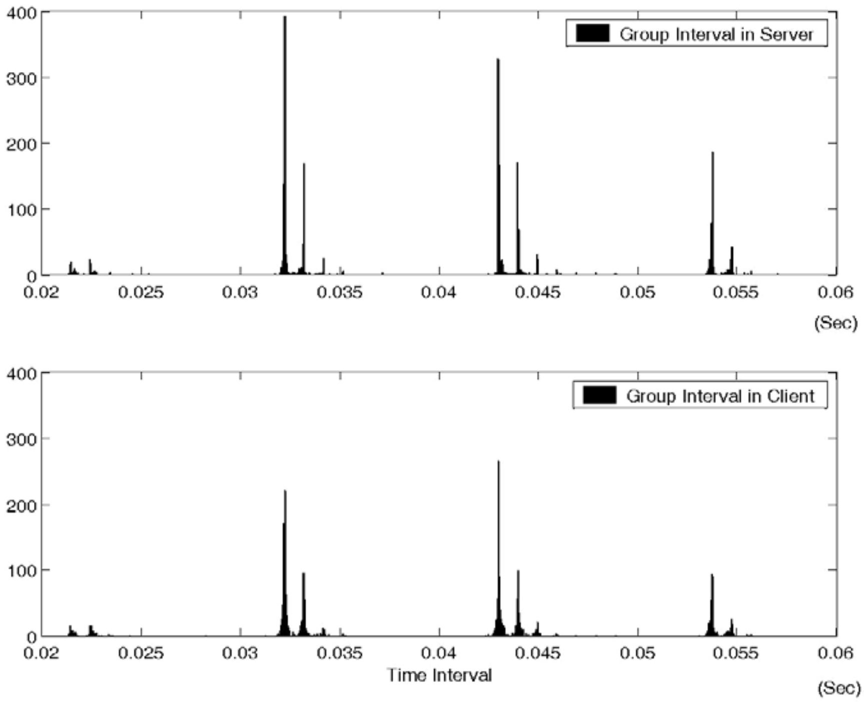



Figure 62: Histogram of group interval with unsmoothed schedule form NCTU to CCU

Fig. 63 to Fig. 66 are the histograms of packet interval. Fig. 63 is with smoothed schedule from NCTU to NCTU, Fig. 64 is with smoothed schedule from NCTU to CCU, Fig. 65 is with unsmoothed schedule form NCTU to NCTU and Fig. 66 is with unsmoothed schedule from NCTU to CCU. According to these histograms, we can see that when the maximum packet size is 1024 bytes, the time interval of each packet in server is about 150 micro sec. Through the network, the time interval in client will be divided into two parts, one part is greater than 150 micro sec, the other is smaller than 150 micro sec, and greater than 0, because when a packet is a little bit late to arrive the client, the time interval between this packet and prior packet elongate, and the time interval between this packet and the next packet shorten. The longer the transmission distance is, the larger the distance between these tow part is, and the smoothing algorithm won't affect this phenomenon.

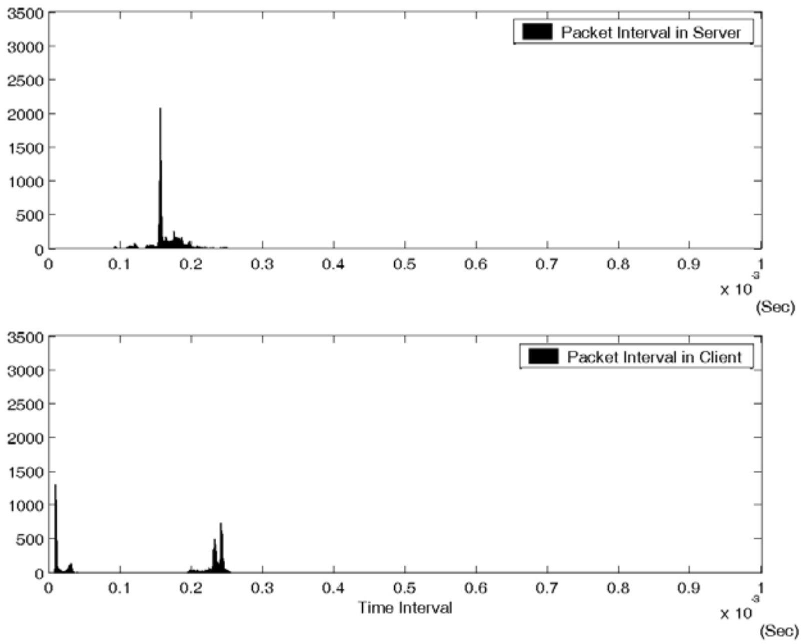



Figure 63: Histogram of packet interval with smoothed schedule form NCTU to NCTU.

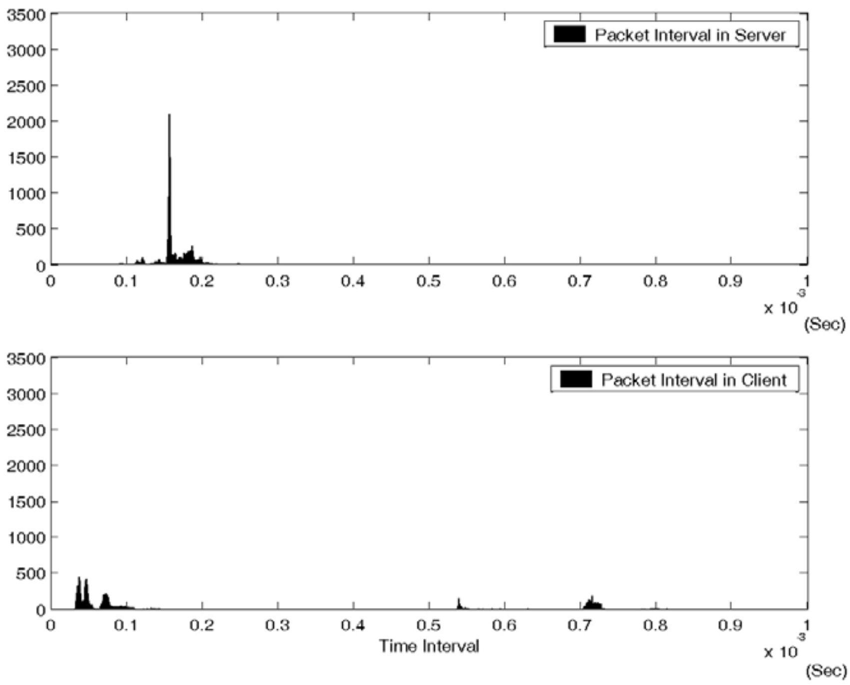



Figure 64: Histogram of packet interval with smoothed schedule form NCTU to CCU.

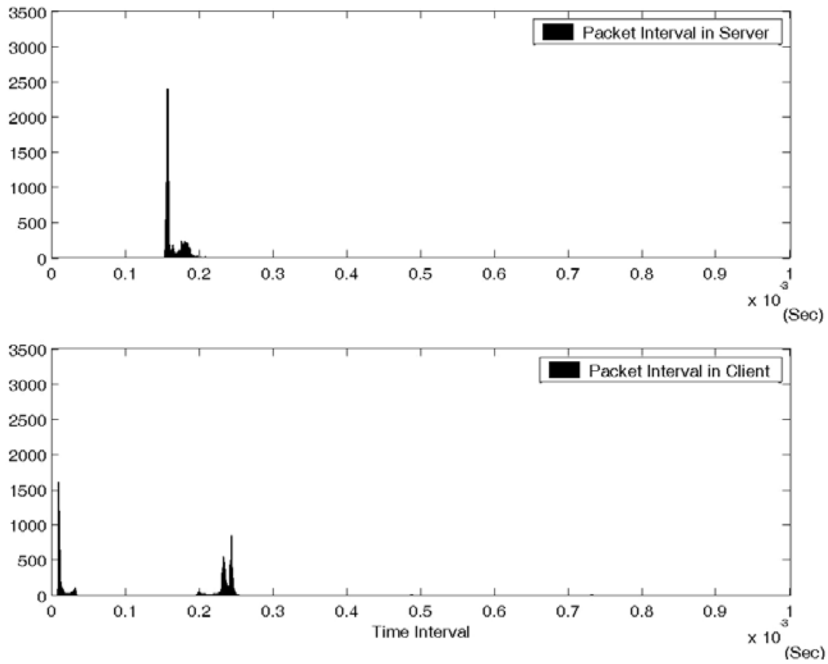



Figure 65: Histogram of packet interval with unsmoothed schedule form NCTU to NCTU.

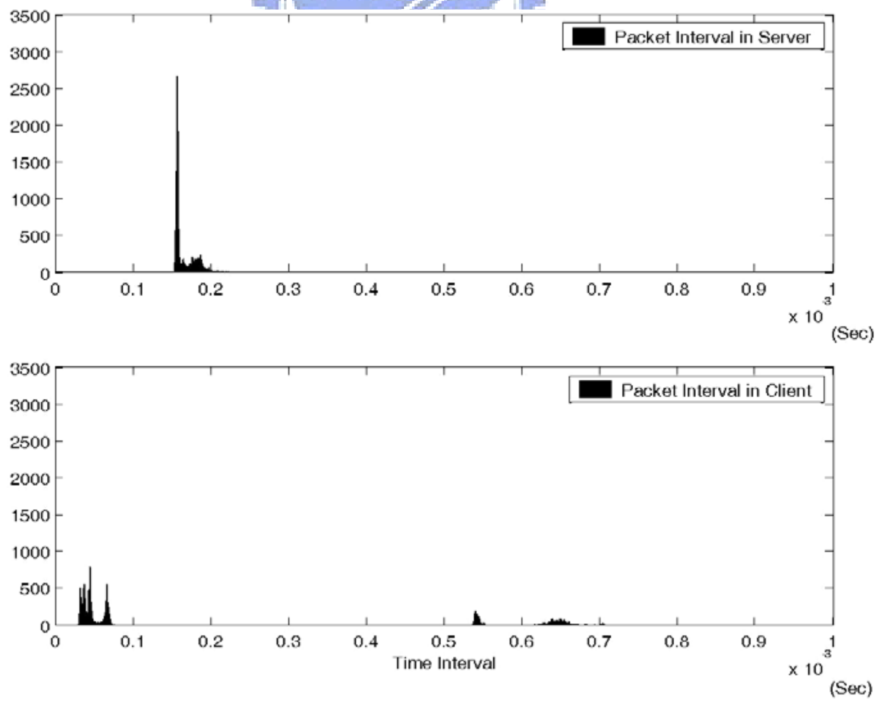



Figure 66: Histogram of packet interval with unsmoothed schedule form NCTU to CCU.

According to these experiments, it shows that the smoothing algorithm won't improve the network jitter, but a server with smoothing algorithm can transmit data at a stable rate and average the transmitted number of packets at each frame slots. Therefore, the transmission period of each packet group shortens, and the effects of network jitter will be reduced.

# 6. Conclusion

In this thesis, we proposed two smoothing algorithms which make the transmission rate smoothing and reduce the variance in transmission rate without modifying the video compression standard. Our algorithms build on MVBA algorithm and incorporate the approach that video streaming is transmitted on network in packet format. For stored video, we control the number of transmitted packet to achieve smoothing transmission schedule under client buffer constraint. One of these two algorithms is the Number-of-Packets Based Smoothing Algorithm. This algorithm aims to reduce the variance in transmitted number of packets. Another one is packet mapped smoothing algorithm that schedules the transmitted number of packets to approximate the schedule generated by MVBA.

According to our simulation result, the first algorithm reduces the variance in transmitted number of packets. However, because of the variant successive packet, the variance in transmission rate doesn't lower down, and to increase the client buffer may not result in the decreasing of variance in transmission rate. This algorithm make the transmitted number of packets stable, thus when there are several video streaming which have to be transmitted on network simultaneously, the collision resulted by transmission media contention will reduce. The transmission schedule generated by the second algorithm is the approximation of MVBA. The transmission rate is more stable and the variance in transmission rate is lower, and the more the client buffer is, the better the performance is. The stable transmission rate makes the utilization of network efficient and the management simple.

We also showed the influence of maximum packet size on transmission schedule.

The smaller the maximum packet size is, the better the performance is. When the maximum packet size is smaller, the peak rate and variance in rate is lower and the bandwidth utilization is more efficient. As the maximum packet size increases, the variance in transmission rate will increase because of the greater variance in successive packets and the peak rate also increases. Thus, the bandwidth utilization is poor efficiency. However, the little maximum packet size results in more packets. The large amount of packet header wastes the bandwidth. Moreover, the increasing of number of packets had to be transmitted will raise the collision probability resulted by transmission media contention. Therefore, maximum packet size is also an important coefficient in our algorithms to result in better performance.

# Reference


[1] P. Seeling, M. Reisslein and B Kulapala, "Network Performance Evaluation Using Frame Size and Quality Traces of Single-Layer and Two-layer Video: A Tutorial," IEEE Communications Surveys and Tutorials Vol. 6, No. 2, 3rd Quarter 2004, pp. 58-78.

[2] J. D. Salehi, Z.-L. Zhang, J. F. Kurose, and D. Towsley, "Supporting Stored Video: Reducing Rate Variability and End-to-End Resource Requirements through Optimal Smoothing" IEEE/ACM Transaction on Networking, 6:397-410, Dec 1998.

[3] S. Sen, D. Towsley, Z.-L. Zhang, and J. Dey, "Optimal Multicast Smoothing of Streaming Video Over the Internet," IEEE Journal on Selected Areas in Communications, Vol. 20, No. 7, September 2002, pp. 1345-1359.

[4] Z.-L. Zhang, J. F. Kurose, J. D. Salehi, and D. Towsley, "Smoothing, Statistical Multiplexing, and Call Admission Control for Stored Video," IEEE Journal on Selected Areas in Communications, Vol. 15, No. 6, August 1997, pp. 1148-1166.

[5] S. Sen, J. L. Rexford, J. K. Dey, J. F. Kurose and F. Towsely "Online Smoothing of Variable-Bit-Rate Streaming Video," IEEE Transactions on Multimedia, Vol. 2, No. 1, March 2000, pp. 37-48.

[6] O. Hadar, S. Greenberg, "Statistical Multiplexing and Admission Control Policy for Smoothing Video Streams Using e-PCRTT Algorithm," The

International Conference on Information Technology: Coding and Computing 2000, pp 272-277.

[7] W.-C. feng and J. L. Rexford, "Performance Evaluation of Smoothing Algorithms for Transmitting Prerecorded Variable-Bit-Rate Video," IEEE Transactions on Multimedia, Vol. 1, No. 3, September 1999, pp 302-313.

[8] J. Zhang and J. Hui, "Applying Traffic Smoothing Techniques for Quality of Service Control in VBR Video Transmissions," Computer Communications, April 1998, pp. 375-389.

[9] S. Sahu, S.-L. Zhang, J, Kurose and D. Towsely, "On the Efficient Retrieval of VBR Video in a Multimedia Server," International Conference on Multimedia Computing and System 1997, pp. 46-53.

[10] J. M. Mcmanus and K. W. Ross, "Video-on-Demand Over ATM: Constant-Rate Transmission and Transport," IEEE Journal on Selected Areas in Communications, Vol. 14, No. 6, August 1996, pp. 1087-1098.

[11] M. Grossglaurser, S. Keshav, and D. N. C. Tse, "RCBR: A Simple and Efficient Service for Multiple Time-Scale Traffic," IEEE/ACM Transactions on Networking, Vol. 3, No. 6, December 1997, pp 741-755.

[12] W.-C. Feng, F. Jahanian and S. Sechrest, "An Optimal Bandwidth Allocation Strategy for the Delivery of Compressed Prerecorded Video," Multimedia Systems, Vol. 5, No. 5, 1997, pp. 297-309.

[13] C. Bewick, R. Pereira and M. Merabti, "Network Constrained Smoothing:

Enhanced Multiplexing of MPEG-4 Video," Seventh International Symposium on Computers and Communications 07 2002, pp. 01-07.

[14] W.-C. Feng, "Rate-Constrained Bandwidth Smoothing for the Delivery of Stored Video," SPIE Multimedia Computing and Networking Conference Feb 1997.

[15] Z. Jiang and L. Kleinrock, "A General Optimal Video Smoothing Algorithm," Proc. IEEE INFOCOM, March 1998.