Int. Computer Symposium, Dec. 15-17, 2004, Taipei, Taiwan.

The Consideration of the Sending Host Timing Drift Problem in Streaming Real-time Communication

Ching-Lung Chang and Zhi-Ren Wang
Department of Computer Science and Information Engineering
National Yunlin Univeristy of Science & Technology, Yunlin, Taiwan 640 R.O.C.
Email: {chang, g9113711}@yuntech.edu.tw

***Abstract*— With the successful of the Internet and the use of broadband in homes, the traffics on the Internet are being shifted from data to multimedia communication. For reducing the buffer requirement on the client side and the service request/response time, streaming technology is normally adopted to handle the transmission of multimedia traffic. This work investigates the timing drift problem of the sending host in streaming communication and applies an adaptive rate control mechanism in cooperation with a client feedback packet to prevent stream buffer overflow or underflow. To reduce the network overhead of the feedback mechanism, a sending host self-timing revision scheme is used.**

I. Introduction

Due to the success of the Internet and broadband to the home, it leads the emergence of several new delay-sensitive multimedia services. These applications generally have a large quantity of real-time information to be transmitted. Therefore, the streaming technology is applied to reduce the store-and-forward delay, the information buffer space, and the service request/response time. Generally, these real-time information services depend on a strict quality of service (QoS) to transport packetized streams of data over a network. Indeed, receiver synchronous playout of real-time data streams is required to deal with delay variations due to result from delivery over a packet-based network. This synchronous playout is normally achieved by buffering received digitized delay-sensitive data for sufficient time.

Studies on digital delay-sensitive packet transmission have been undertaken. The issue of controlling playback has been made by [1][2][4]. Based on a calculation of the round-trip-times for TCP retransmission timer [6], Ramjee [1] presented different playout control algorithms for interactive packet-audio applications to maintain the packet loss ratio at almost zero. Tobe et al. [7] presented a framework of QoS management that controls the sending stream according to the conformance of generated traffic measured at the network device driver. Lakshman et al. [8] investigated the integrated management of a composite resources such as the CPU, network-interface, and bus bandwidth, and presented a resource management architecture in which applications and OS cooperate to adapt dynamically to variations in the resource requirements and availability, thereby providing predictable QoS from both endsystem and network resources.

In real-time streaming communication, the sending host generates a digital data packet at a certain rate. The resources of the sending host, including CPU time, memory bandwidth, I/O bus bandwidth, and network interface are shared among the active tasks and operating system. The data-retrieval time varies with the rotation time and the seek time for searching the hard disk data. Furthermore, the reference timer in the sending host may differ from the application required. Accordingly, the sending host cannot easily generate the digitized packet in time as application expected. The difference in time between the actual packet generation time of the sending host and the packet generation time expected by the application program (i.e., theoretical packet generation time) here defines - the sending host timing deviation which doesn't include network interface queuing delay. For example, when considering the constant bit rate applications, the theoretical packet generation time of the nth packet is the previous theoretical packet generation time t plus a constant packetization delay c. If the sending host actually generates the nth packet at time t_n, then the sending host timing deviation is $|t_n - (t + c)|$.

In contrast to previous works, this study considers the timing variation of sending host and focuses on the design of the buffer used in the real-time streaming transmission. The buffering technique is discussed in relation to constant bit rate (CBR) MP3-music playback. However, the presented results can be applied to any similar real-time data. An adaptive rate control mechanism, in cooperation with the feedback packet generated by the application host is presented to prevent the stream-buffer overflow or underflow and thus mitigate the network delay, jitter, and timing drift of the sending host. Under the assumption of jitter-guaranteed networking, the associated key parameters, including initialized buffering delay, stream-buffer space, and high/low threshold for the stream-buffer, are derived. The mechanism is developed and verified by computer simulation.

The rest of this paper is organized as follows. Section 2 addresses some design issues concerning the streaming real-time application to provide continuous stream data playback. Section 3 introduces the operation of the adaptive rate-control mechanism. Section 4 presents simulation results. Finally, concluding remarks are given in Section 5.

II. Design Considerations

A characteristic of the packet-based Internet is that packets may suffer various delays, losses, and disordering. This paper focuses on both delay and delay jitter (the maximum variation in network delay experienced by packets) of an MP3 stream

with streaming technology. This problem can be solved by the receiver using per-stream playout buffers (called a stream-buffer herein), as shown in Fig. 1 [3]. The stream-buffer recovers the original packet spacing, while avoiding overflow or underflow, to guarantee the playback quality as perceived by a human user. Thus, this work considers the following issues.

- Stream-buffer underflow: Figure 2 illustrates packet transmission between an MP3 server and an MP3 client. The MP3 server periodically generates an MP3 packet in every period of I time. The generated packet size is $I \cdot MP3\_music\_encoding\_rate$. The CBR MP3 packets may experience different network delays D. Therefore, a stream-buffer resynchronizes the received packets so that they conform to the original CBR pattern. The period for which the initialized buffering delay (the time between receiving the first MP3 packet and feeding that packet to the MP3 decoder) can reproduce a jitter-free playback is analyzed. Let $D(i)$ represent the end-to-end network delay experienced by the ith MP3 packet and $t(i)$ be the ith packet arrival time. Based on Fig. 2 illustration, to achieve a constant playback rate, the second packet is delayed a variable amount $B(2)$ in stream-buffer to satisfy

$$t(2) + B(2) = t(1) + B(1) + \rho,$$

where the $B(i)$ represents the stream-buffer delay experienced by the ith packet and ρ is the MP3 packet playout time which is equal to I. In general,

$$\begin{aligned} t(i) + B(i) &= t(1) + B(1) + (i-1)\rho, \\ B(i) &= t(1) + B(1) + (i-1)\rho - t(i), \\ B(i) &= B(1) + D(1) - D(i). \end{aligned}$$

If the computed value of $B(i)$ is positive, or

$$B(1) > D(i) - D(1), \forall i, \qquad (1)$$

then that the stream-buffer underflow can be prevented. Since the network delay jitter J is

$$\underset{\forall ij}{Max} |D(i) - D(j)|,$$

if $B(1) > J$, then equation (1) also holds. Thus, if the initialized buffering delay $B(1)$ exceeds the network jitter J, we can prevent the stream-buffer underflow.
- Stream-buffer overflow: Not only stream-buffer underflow, but also stream-buffer overflow must be considered to ensure the quality of the MP3 music playback. Given the network delay jitter J, in the worst case, the MP3 client may receive n back-to-back MP3 packets (the maximum burst size, MBS), as shown in Fig. 3. The value of n is given by,

$$n = \lfloor 1 + \frac{J}{I - \delta} \rfloor,$$

where $\lfloor x \rfloor$ is the maximum integer that is less than or equal to x and δ is the MP3 packet transmission time which is defined as $packet\ size / network\ speed$. Therefore, to prevent stream-buffer overflow, the stream-buffer space must satisfy

$$stream\_buffer\ space > (n+1) \cdot P_{size}, \qquad (2)$$

where the P_{size} is the MP3 packet size.
- Sending host timing drift: If an original stereo (two channels) music source is sampled at a frequency of 44.1 KHz and quantized using 16 bits, then the digitized bit rate of the stereo music is,

$$44100 \times 16 bits \times 2 = 176 Kbytes/sec.$$

When the digitized stereo music is compressed using the MPEG 1 level 3 technique [5], a 16 Kbytes per second CBR MP3 music can be derived, if the compression rate is around $1 : 11$. If the MP3 server packetizes the MP3 music using a unit of 512 bytes, the MP3 client must receive an MP3 packet every $32\ ms$ $(512\ bytes/16\ Kbytespersecond)$ to provide acceptable QoS to users. Consequently, in addition to the network delay jitter, the time taken to generate a CBR MP3 packet also influences the stream-buffer overflow/underflow. Unfortunately, generating timely CBR MP3 packets in the sending host as required by the application is difficult. In the sending host, it is hard to dedicate resources to a particular task consequently, data retrieval time is unpredictable and the reference timer in the host may deviate. We did an experiment to prove the above statement. As Fig. 4 shown, an periodic packet generation server is emulated to generate an periodic packets (every 32 ms generates 512 bytes) in both MS-DOS and Microsoft Windows-98 environments. The 8254 timer chip [9] is directly controlled to obtain the required precise timing information. An ARM board, which runs on the 50 MHz clock rate, is connected to the server with Ethernet interface. The ARM board measures the time that the generated periodic-packets are captured and forwards the associated timing information of captured packet to the display PC via RS-232 interface. Figure 5 plots the results measured in the Microsoft Windows environment over four minutes (A typical MP3 music playback time is around four minutes). The vertical axis in Fig. 5 represents the accumulated deviation time between the sending host and the NetXRay PC. The time on the X-axis is the emulation time. The accumulated deviation time is defined as,

$$\sum_{i=1}^{n} (t_i - TH_i),$$

where t_i is the ith packet generation time by the sending host and TH_i is the theoretical packet generation time of the ith packet. Therefore the slope of the curve is the drifting rate. Figure 6 is the zoom in of a two-second segment in Fig. 5. Table 1 summarizes this measurement. As shown in Table 1, during the four minutes, the accumulated deviation times are -578 ms and 154 ms in DOS

and Windows environments, respectively. Therefore, the sending host sends 12288 $bytes$ more data than expected in the DOS environment in the four minutes of connection time. In the windows environment, 3072 $bytes$ fewer data than expected by the application host are sent out. Clearly, packet generation in the sending host may be slower or faster than the rate we want. Accordingly, the stream-buffer can either overflow or underflow.

According to the above consideration, not only the network jitter, but also the timing drift of the sending host must be considered to prevent the stream-buffer from overflowing or underflowing. The following section introduces an adaptive rate control mechanism to correct the sending host timing drift.

III. Adaptive Rate Control Mechanism

According to the above analysis, the suitably initialized buffering technique can prevent buffer overflow/underflow only on the the jitter-guaranteed network. Once the sending host timing drift is considered, the buffer space is dependent on the time of the connection.

Based on the MP3-music on demand (MoD) application, this section presents an adaptive rate control mechanism to adjust the sending host timing drift and thus bound the required initialized buffering delay and stream-buffer space. Figure 7 depicts the adaptive rate control mechanism, which is a feedback mechanism.

In the proposed mechanism, a high threshold (HT) and a low threshold (LT) are defined for the stream-buffer occupancy to prevent stream-buffer overflow or underflow. After an MP3 packet arrives and is stored into the stream-buffer, if the stream-buffer is below the low threshold or over the high threshold, the MP3 client sends a warning packet, which contains the stream-buffer occupancy, to the MP3 server. The MP3 server adjusts the MP3 packet generation rate and does the self-timing revision scheme to adjust the timing drift according to the information of the stream-buffer occupancy.

A. Thresholds

When considering the server timing drift, the MP3 packet generation rate (R_i) is,

$$R_i = \frac{P_{size}}{I + \alpha},$$

where P_{size} is the MP3 packet size; I is the packetization delay expected by the application host, and α is the sending timing deviation in each I, which may be positive or negative. The MP3 decoder rate R_o is,

$$R_o = \frac{P_{size}}{I}.$$

If α is negative, then the sending timing drift may induce stream-buffer overflow. In contrast, the stream-buffer may underflow if α is positive. Thus the pertinent question concerns how to define the threshold values to prevent stream-buffer overflow or underflow in the feedback mechanism.

- High threshold value: The stream-buffer will undergo overflow at time $T_{overflow}$, if it does not have a mechanism to compensate for the negative α. The $T_{overflow}$ is derived as,

$$T_{overflow} = \frac{Buf_{max} - B(1) \cdot R_i}{R_i - R_o}, \quad (3)$$

where Buf_{max} is the stream-buffer space. Thus, before the stream-buffer overflows, the sending host should receive a warning packet and adjust the associated stream data transmission rate to prevent it. First, a network round-trip time (RTT) that is less than the packetization delay I is considered. Consequently, the worst case network behavior that induces the stream-buffer overflow is that the $(i-1)$th MP3 packet arrives at time t with maximum network delay and the ith MP3 packet arrives with minimum network delay. In this worst case, the arrival of nth packet will not cause buffer overflow if

$$Buf(t^+) - R_o(I + \alpha - J) + P_{size} \leq Buf_{max},$$
$$\Rightarrow Buf(t^+) \leq Buf_{max} - R_o(J - \alpha),$$

where $Buf(t^+)$ is the stream-buffer occupancy when the $(i-1)$th MP3 packet arrives at time t and is stored into the stream-buffer and herein the P_{size} is the packet size of the ith packet. Hence, when the value of $Buf(t^+)$ exceeds $Buf_{max} - R_o(J - \alpha)$, then the MP3 client generates a warning packet to the MP3 server to adjust the size of the ith MP3 packet thereby avoiding the stream-buffer overflow. Consequently, the high threshold should be set to,

$$High\ threshold = Buf_{max} - R_o(J - \alpha). \quad (4)$$

- Low threshold value: When the sending timing deviation α is positive, the stream-buffer will underflow at $T_{underflow}$. The value of $T_{underflow}$ can be represented as,

$$T_{underflow} = \frac{B(1) \cdot R_i}{R_o - R_i}. \quad (5)$$

If the RTT is also assumed to be less than the packetization delay I, the low-threshold value can only consider the network jitter only. The worst case network behavior that induces the stream-buffer underflow is that the $(i-1)$th MP3 packet arrives at time t with minimum network delay and the ith MP3 packet arrives with maximum network delay. To prevent buffer underflow at the time that the nth packet arrives, the following inequality needs to hold:

$$Buf(t^-) + P_{size} - R_o(I + \alpha + J) > 0,$$
$$\Rightarrow Buf(t^-) > R_o(J + \alpha),$$

where the $Buf(t^-)$ represents the stream-buffer occupancy before storing the $(i-1)$th packet and the P_{size} is the size of the $(i-1)$th packet. Hence, when the $(i-1)$th packet is received, if the value of $Buf(t^-)$ is less than $R_o(J+\alpha)$, then the client generates and sends a warning

packet to the server to adjust the size of the ith MP3 packet. Consequently, the low threshold value should be set to,

$$Low\ threshold = R_o(J + \alpha). \quad (6)$$

In relation to the condition of RTT greater than the packetization delay I, the interval between the time that the warning packet is issued to the time that a revised-length packet arrives at the client is RTT. Therefore, there are $n = \lfloor \frac{RTT}{I+\alpha} \rfloor$ normal-size packets arrival during the interval. These n packets cause additional $(n \cdot \alpha \cdot R_o)$ bytes difference in buffer occupancy. Therefore, equation (4) and (6) are revised to,

$$High\ Threshold = Buf_{max} - R_o[J - (n+1)\alpha],\ (\alpha \leq 0) \quad (7)$$

$$Low\ Threshold = R_o[J + (n+1)\alpha],\ (\alpha \geq 0). \quad (8)$$

B. Self Timing Adjustment

This subsection introduces a method of timing drift adjustment in the sending host to reduce the amount of the warning packets. Thus, the network overhead of the adaptive rate control mechanism can be reduced.

The concept behind the timing drift adjustment is that the sending host computes the period between the consecutive warning packets arrival. The duration is represented as T_{fb} herein. The amount of accumulated deviation data DEV, in the T_{fb} interval, can be derived. Thus, the sending time deviation of the sending host within a packet generation interval I is estimated as

$$\alpha = \frac{DEV/coding\ rate}{T_{fb}/I} \quad (9)$$

However, the amount of DEV may deviate because of the network delay. In the worst case, the deviation of DEV is $J \cdot R_o$. Thus, α may not given an accurate estimate of the sending time deviation. However, if the packetization delay I is adjusted by α, then the timing drift between the sending host and the application client can still be reduced to some extent. Therefore, the number of the warning packets, and hence the network overhead of the feedback rate control mechanism, can be reduced.

IV. Simulation

Figure 8 presents the system simulation model. This simulation focuses on the consideration of both network delay and server time drift for the MoD system. When an MP3 server decides to generate a MP3 packet, the packet generation time incurs a timing deviation. Then the generated packet experiences a network interface queuing delay in the MP3 server and a transportation delay in the network before the packet reaches the stream-buffer of the MP3 client. The network interface of the MP3 server is assumed to be the 100 Mbps Ethernet. The buffering delay of the arrived MP3 packet depends on the occupancy of the stream-buffer.

This simulation involves the streaming of MP3 music at 128 Kbps for 4 minutes' connection time. The MP3 server generates a 512-byte MP3 packet every 32 ms ($512\ bytes/128\ Kbps$) in an MP3 connection. The figures show simulation results that follow two lines, which represent the occupancy of the stream-buffer when an MP3 packet is received (lower line) and the occupancy of the stream-buffer when the received MP3 packet is stored in the stream-buffer (upper line), respectively. The times of the stream-buffer overflow or underflow indicate the QoS of the realtime streaming communication.

Without using any control scheme except for the buffering technology, the effects of the sending time drift on the occupancy of the stream-buffer are considered. Figure 9 presents the simulation results. For negative α, the first stream-buffer will overflow during the 8th second, as shown in Fig. 9 (a). In this situation, the stream-buffer overflows 666 times during the four minutes of connection. Figure 9 (b) presents the results for positive α. As Fig. 9 (b) shows, the stream-buffer underflows 746 times in four minutes of playing MP3 music.

Clearly, buffering alone does not suffice to mitigate the sending time drift that may cause poor QoS. Thus, an adaptive rate control mechanism is presented to compensate for the effects of the sending time drift. For absorbing the sending host timing drift, an extra elastic buffer is provided. Therefore, B(1) and the stream-buffer size are modified to $2J$ and $R_o(4J + I) \cdot R_o$, respectively. Accordingly, Fig. 10 shows the simulation results of the associated rate control mechanism. The stream-buffer will not experience overflow and underflow, as expected. Under these conditions, warning packets are fed back to the server 61 times to compensate for the sending time drift during four minutes of connection time for streaming the MP3.

A sending host self timing compensation method to adjust the sending host timing to the required timing of application to reduce the overhead of the feedback mechanism. The sending host records the time between the interval of the consecutive warning packets and coordinates of the feedback warning packet to derive the timing deviation value α, according to Eq. (9). Figure 11 presents the simulation results. In this mechanism, the number of warning packets is reduced to 18 and 4 under the faster and slower timing deviation conditions, respectively.

V. Conclusions

The two major QoS factors, essential for end-to-end communication in distributed client-server real-time and multimedia applications, are the delay time associated with server packet generation and that of network packet transportation. The sending host timing bias causes the accumulated timing deviation between the sending host and the application host to depend on the connection time. This paper considers MP3 digital music on demand systems, although the results are applicable to all delay-sensitive data. An adaptive rate control mechanism is proposed. It is a feedback mechanism to compensate for the sending host timing bias and prevent stream-buffer overflow or underflow under limited stream-buffer conditions. For a jitter-guaranteed network, the key parameters of initialized build-out delay, low threshold, high

threshold, and the space of stream-buffer have been analyzed. In cooperation with the feedback mechanism, a sending host self timing revision scheme is used to reduce the network overhead of the feedback mechanism. Simulation results show that the proposed mechanism that includes revised sending host self timing provides good QoS at the cost of a little network overhead.

ACKNOWLEDGMENT

The authors would like to express their appreciation to the National Science Council of Taiwan R.O.C. for supporting under grant NSC 93-2213-E-224-026.

REFERENCES

[1] Ramachandran Ramjee, Jim Kurose, Don Towsley, and Henning Schulzrinne, "Adaptive Playout Mechanisms for Packetized Audio Applications in wide-Area Networks," *in Proc. IEEE Inforcom'94*, pp. 680-688, April, 1994.

[2] Cormac J. Sreenan, Jyh-Cheng Chen, and B. Narendran, "Delay Reduction Techniques for Playout Buffering," *IEEE Trans. on Multimedia*, Vol. 2, No. 2, June 2000.

[3] William Stallings, "High-speed Networks TCP/IP and ATM Design Principles," *Prentice Hall*, 1998.

[4] Kouhei Fujimoto, Shingo Ata, and Masayuki Murata, "Statistical Analysis of Packet Delays in the Internet and Its Application to Playout Control for Streaming Applications", *IEICE Trans. Commun.*, Vol. E84-B, No. 6, June 2001.

[5] ISO/IE 11732-3, "Information Technology-Coding of Moving Pictures and Associated audio for Digital Storage Media up to 1.5 Mbit/s", *Part3-Audio*.

[6] V. Jacobson, "Congestion avoidance and Control", *in Proc. ACM SIGCOMM*, pp. 314-329, Aug. 1988.

[7] Yoshito Tobe, Yosuke Tamura, and Hideyuki Toluda, "Rate-based QoS Control of Multiple Flows over a Real-Time OS", *in Proc. IEEE Fifth Int. Conf. on Real-time Computing Systems and Applications*, pp. 83-90, 1998.

[8] K. Lakshman, Raj Yavalkar, and Raphael Finkel, "Integrated CPU and Network-I/O QoS Management in an Endsystem", *in Proc. Fifth Int. Workshop on Quality of Service*, pp. 167-178, 1997.

[9] M. A. Mazidi and J. G. Mazidi, "The 80x86 IBM PC and Compatible Computers (Volume II)- Design and Interfacing of the IBM PC, PS, and Compatibles", Third Edition, Prentice Hall, 1998.

Table 1 Summary of the timing deviation in sending host

OS	Total Time(ms)	Dev. Time(ms)	Dev. packet	Dev. Data(byte)
DOS	239421.5	-578.5	24	12288
Windows	240154.5	154.5	-6	-3072

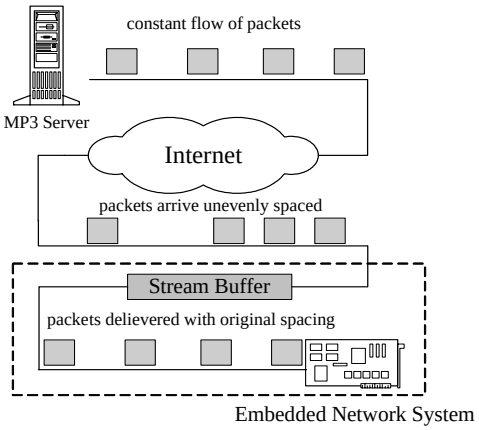


Fig. 1. Communication between MP3 server and MP3 client

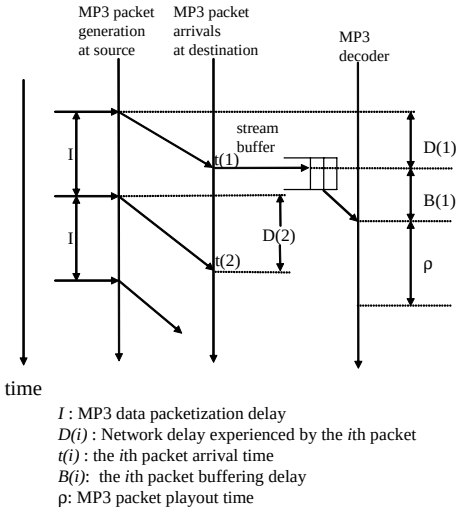


Fig. 2. Stream buffer underflow

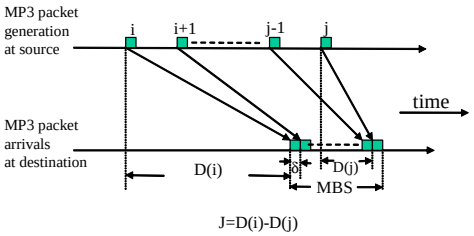


Fig. 3. Stream buffer overflow

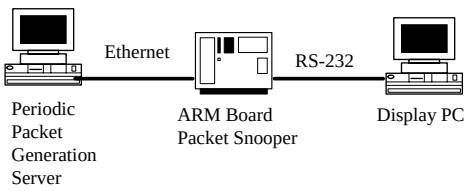


Fig. 4. Snooped Environment for Sending Host Timing Drift Problem

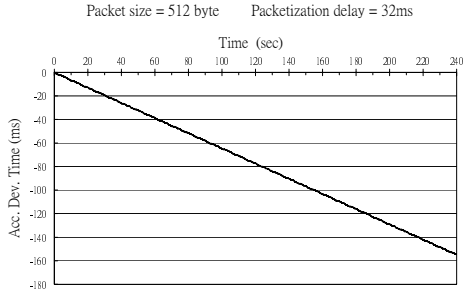


Fig. 5. Measured deviation in the timing of the sending host

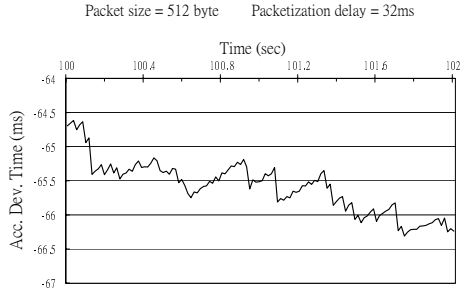
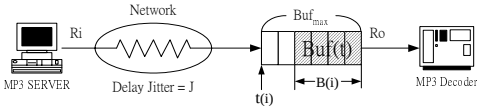


Fig. 6. Zooming in on Fig. 5



t(i) : *i*th packet arrival time
B(i) : *i*th packet buffering delay (B(1) = J)
R_i : Sending host transmission rate
R_o : MP3 data decoding rate
Buf(t) : Buffer occupancy at time t
Buf_{max} : Stream-buffer space

Fig. 7. Adaptive rate control mechanism

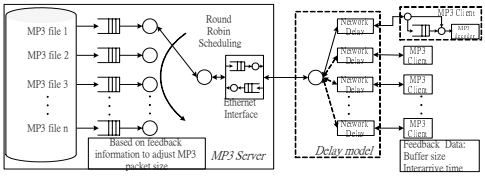
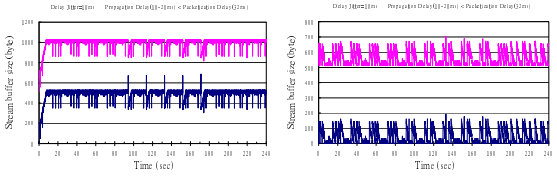
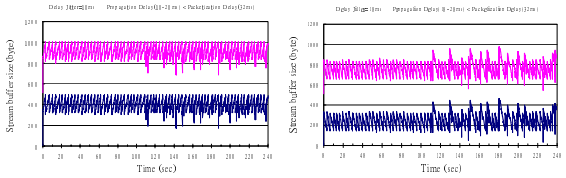


Fig. 8. Simulation model of the MoD system



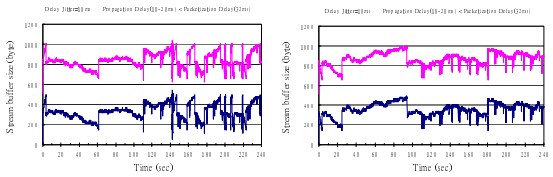
(a) α < 0 (b) α > 0

Fig. 9. Occupancy of the stream-buffer considering the sending host timing bias



(a) α < 0 and B(1) = 2J (b) α > 0 and B(1) = 2J

Fig. 10. Occupancy of stream-buffer with adaptive rate control mechanism



(a) α < 0 and B(1) = 2J (b) α > 0 and B(1) = 2J

Fig. 11. Occupancy of stream-buffer with adaptive rate control mechanism and server self timing compensation scheme for B(1)=2J