A Retransmission–Based Continuous Media Multicast Communication Scheme: The Effect of Buffering Time for Media Synchronization on Application–Level QoS

Toshiro NUNOME and Shuji TASAKA
Graduate School of Engineering,
Nagoya Institute of Technology, Nagoya 466–8555, Japan

ABSTRACT

This paper proposes a retransmission–based error recovery scheme in multicast communications for live audio and video streams transferred over the Internet/intranets. The scheme is referred to as MRVTR (Multicast based on Retransmission with Virtual–Time Rendering). In order to suppress retransmission traffic, MRVTR adopts a method for controlling NACK (Negative Acknowledgment) transmission according to the network load. By simulation, we first demonstrate the effectiveness of the proposed schemes in terms of application–level QoS. We then investigate the relationship between buffering time for media synchronization and application–level QoS of MRVTR.

Keywords: Multicast Communications, Retransmission Control, Live Media, QoS, Media Synchronization

I. INTRODUCTION

Multicasting is an important technique for audio and video streaming, which is one of the most promising applications in the Internet and intranets. Usually, multicasting in the networks employs IP multicast, which provides the best–effort service and no *QoS (Quality–of–Service)* control mechanism.

A variety of studies on retransmission–based reliable multicast protocols for QoS guarantee have been reported [1]. For example, *RMTP (Reliable Multicast Transport Protocol)* [2] was proposed for transfer of discrete media like computer data, while *STORM (STructure–Oriented Resilient Multicast)* [3], *LVMR (Layered Video Multicast with Retransmission)* [4] and *RMTP–II* [5] were proposed for distribution of continuous (or stream) media such as audio and video. RMTP uses positive acknowledgments (ACKs) to recover from packet loss; this approach is often referred to as *sender–initiated* [1], since it is the responsibility of the sender to detect packet loss. On the other hand, in STORM and LVMR, the source of media streams detects packet loss by negative acknowledgments (NACKs) from destinations; this approach is called *receiver–initiated* [1]. RMTP–II combines ACKs and NACKs to achieve reliability in both continuous and discrete media transfer.

These protocols have solved the *feedback–implosion* problem [6]. That is, if every receiver reports about the success or failure of the data transfer, the sender will be overwhelmed with feedback packets (ACKs or NACKs). A typical solution to the problem groups receivers into local regions and generates a single feedback packet per local region [2].

The QoS assessment of the reliable multicast protocols proposed so far has also been made; however, most of them deal with *node–level*, *network–level* and *end–to–end–level QoS* such as transfer efficiency, delay and delay jitter at the transport layer or lower layers of the protocol stack. For the users, on the other hand, the subjective quality (i.e., *user–level QoS*) is the most important QoS; it is closely related to *application–level QoS* [7]. Nevertheless, very few studies on the reliable multicast refer to application–level QoS.

For continuous media, an important feature that distinguishes them from discrete media is the temporal structure of the former. The preservation of the temporal structure is essential to application–level QoS of continuous media; this is referred to as *media synchronization*. Thus, we regard the quality of media synchronization as the major part of the application–level QoS in this paper. It should be noted that retransmission of *media units (MUs)*, each of which is the information unit for media synchronization, aggravates network delay jitter. Therefore, when we use some retransmission–based protocol for error recovery of continuous media, we should give considerable thought to the issue of media synchronization.

In [8], the authors propose a retransmission–based error recovery scheme in cooperation with media synchronization for audio and video streaming. It is referred to as *Retransmission with Virtual–Time Rendering (RVTR)*. However, RVTR is proposed for unicast communications and then is not suited to multicast communications. If we use RVTR with no enhancement for multicast communications, the number of MU retransmission greatly increases owing to many requests from many destinations. Therefore, the output quality of media streams may degrade drastically.

In this paper, we enhance RVTR to cope with multicast communications in middle–scale intranets. We refer to the scheme as *multicast based on RVTR (MRVTR)*. MRVTR employs an receiver–initiated global retransmission scheme in which neither special relay nodes nor routers are needed in the multicast tree. Therefore, it can be applied to the current Internet/intranets easily.

The retransmission–based error recovery scheme should have enough buffering time for media synchronization to absorb the aggravated delay jitter. In live audio and video streaming, on the other hand, *MU delays*, which is defined as the time interval from the moment an MU is generated until the instant the MU is output, should be short for keeping the real–time property of media streams. However, MU delays increase as the buffering time increases. Thus, we are faced with a tradeoff between the improvement of media synchronization quality by retransmission and the preservation of the real–time property of media streams.

In this paper, we demonstrate the effectiveness of the proposed scheme MRVTR for live audio and video streams in terms of application–level QoS by simulation in which a middle–scale intranet is supposed. We then investigate the relationship between the buffering time and the application–level QoS.

The rest of the paper is organized as follows. Section II proposes MRVTR. Section III illustrates a methodology for the assessment of the application–level QoS, including configuration of the simulation and QoS parameters. The simulation results are presented and discussed in Section IV.

II. MRVTR

We suppose that a source multicasts an audio stream and the corresponding video stream to a constant number of destinations. The audio and video are transmitted as two separate transport streams. A video frame is defined as a video MU, and an audio packet consisting of a constant number of audio samples as an audio MU.

Both MRVTR and RVTR employ the *Virtual–Time Rendering (VTR) algorithm* [9], [10] enhanced for managing MU drop and retransmission for media synchronization. When the destination notices any failure in MU transfer, it sends a NACK to the source. Then the retransmission is attempted during a limited time interval specified by the enhanced VTR algorithm.

Enhanced VTR Algorithm

The enhanced VTR algorithm adaptively changes the buffering time according to the amount of delay jitter of MUs received at the destination and MU loss [8]. Initially, the buffering time is set to a rough estimate of the maximum delay jitter, which is denoted by $J_{\max}$ [8]; after the first MU is received, it can be changed by the modification of the *target output time* of each received MU. The target output time is the time when an MU should be output. When the MU arrives at the destination too late after the target output time, the target output time is expanded to absorb the jitter and provide extra time for retransmission; this means increase in the buffering time. In order to preserve the real–time property of live media, we can set the *maximum allowable delay* Δ_{al} [8] so that the modification of the target output time does not make MU delay exceed this limit. Furthermore, the target output time can be contracted when the amount of delay jitter decreases; this means that the buffering time decreases.

In this paper, the audio is selected as the master stream and the video as the slave stream since audio is more sensitive to intra–stream synchronization error than video. Only the master stream can modify the target output time for itself, and accordingly the slave stream modifies it by the same amount at the same time.

NACK Control Scheme

MRVTR adopts a method of unicast NACK transmission on the basis of the results in [11]. Furthermore, the source retransmits requested MUs to all the destinations by multicasting as in [12].

The feedback–implosion is a problem which may occur in multicast communications. Many studies solve the problem by grouping receivers into local regions and generating a single feedback packet per local region; this scheme need a special relay node which gathers feedback packets per local region. In this paper, however, we focus on a global retransmission scheme without any special relay node in order to be compatible with the current Internet/intranets.

We also find other schemes which solve the problem in the literature. For example, each destination sets a random timer before sending a feedback packet [13]. Another example is that each destination generates a single feedback packet which includes feedback information for multiple received or lost packets at the destination [2]. However, these schemes cause notification latency of feedback information. Thereby, they are not appropriate from a real–time property point of view.

In order to suppress the traffic, MRVTR employs a method for controlling NACK transmission according to the network load, which is called the *NACK control scheme*. It is employed by the video stream only. This is because the size of a video MU is usually much larger than that of a voice MU, and then excessively retransmitted video MUs may cause significant degradation of the output quality of the media streams.

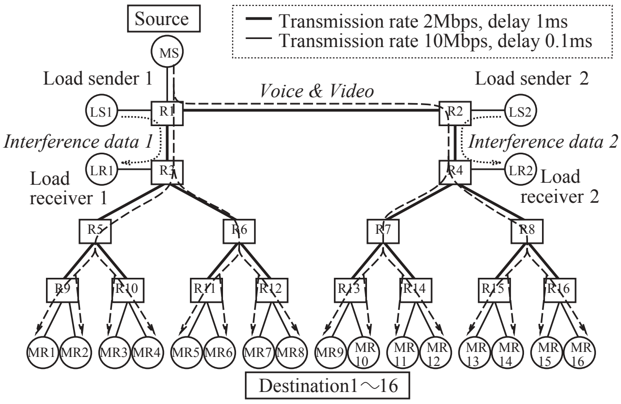

Fig. 1. Network configuration.

In what follows, we describe how to know the network state at each destination and how to control the NACK transmission. In order to know the network state, the destination periodically measures the loss rate of newly transmitted MUs of video. We define it as the ratio of the number of lost MUs newly transmitted to the total number of newly transmitted MUs which should be received during each observation period (say, 5 seconds). The network state is classified into three levels: *CONGESTED*, *LOADED* and *UNLOADED*. When the loss rate is larger than a threshold value L_{high}, the level is set to CONGESTED; that is, network congestion is detected, and then the destination sends no NACK. The loss rates smaller than another threshold L_{low} ($L_{\mathrm{low}} < L_{\mathrm{high}}$) mean the level of UNLOADED, which is considered lightly loaded. At this level, lost MUs are very few, and then the application–level QoS hardly degrades; thus, the destination sends no NACK. The network sate of LOADED means that the network load is moderate; therefore, the destination transmits NACKs.

III. METHOD OF THE SIMULATION

In order to assess application–level QoS of MRVTR, we perform computer simulation with *ns–2 (network simulator version 2)* [14]. In this paper, we suppose sixteen terminals receiving multicast live media; this is an example of a middle–scale intranet.

Network Configuration

Figure 1 illustrates the network configuration in the simulation. Rk ($k = 1, 2, \cdots, 16$) denotes a router node. MS is a source terminal node, and terminal MRl ($l = 1, 2, \cdots, 16$) is a destination. Furthermore, LS1 and LS2 display load sender terminals, while LR1 and LR2 are the load receiver terminals. Each connection between two router nodes is a 2 Mbps duplex link with a transmission delay of 1 ms; the link is assumed to be a serial line. The link between a router node and a terminal node is an Ethernet; its transmission rate is 10 Mbps, and the transmission delay is 0.1 ms. Each link has a FIFO (First–In First–Out) queue.

In the simulation, we suppose MS to be the voice–video source. MS multicasts the media streams to all the destinations. We employ the *Dense Mode protocol* [14] for multicast routing; the protocol is pre–installed in ns–2 and is similar to PIM–DM (Protocol Independent Multicast – Dense Mode) [15].

We use a voice stream of ITU–T G.711 μ–law and an MPEG1 video stream. Table I shows the specifications of the voice and video. Furthermore, we take media capturing and encoding delay time into consideration in the simulation. The capture duration of a voice MU equals the inter–MU time, which is 50 ms in this paper, and the time needed

TABLE I

SPECIFICATIONS OF THE VOICE AND VIDEO.

item	voice	video
coding scheme	ITU–T G.711 μ–law	MPEG1 GOP I
image size [pixels]	—	176 × 128
original average MU size [bytes]	400	1905
original average MU rate [MU/s]	20.0	
original average inter–MU time [ms]	50.0	
original average bit rate [kbps]	64.0	305.0
measurement time [s]	90.0	

to encode is negligible; therefore, we set the capturing and encoding delay time of each voice MU is 50 ms. On the other hand, the capture duration of a video MU is just a moment. However, it spends much time to encode a video frame. In this paper, we set the capturing and encoding delay time of each video MU to 8 ms, which is the same time as that in the experimental system in [8][1]. Each MU leaves the source the capturing and encoding delay time after its timestamp.

We employ the same values of the thresholds and the parameters for RVTR as those in [8], except for $J_{\max}$ and $\Delta_{\rm al}$. Moreover, as the threshold values for the NACK control scheme, we select $L_{\rm low}$ =5 % and $L_{\rm high}$ =10 %; these values were determined after we had tried several values for each.

LS1, LS2, LR1 and LR2 are used to handle traffic flows of interference. LS1 sends fixed–size IP datagrams of 1500 bytes each to LR1 at exponentially distributed intervals; these data messages are referred to as *interference data 1*. LS2 also transfers data messages to LR2 in the same way as LS1; we refer to them as *interference data 2*. The amount of the interference data traffic is adjusted by changing the average of the interval.

QoS Parameters

In order to assess the application–level QoS of the proposed scheme, we need to examine the media synchronization quality as well as the efficiency of information transfer at the application–level.

For the quality assessment of intra–stream synchronization for audio or video, we first evaluate the *coefficient of variation of output interval*, which represents the smoothness of output of a media stream. In addition, we use the *MU loss rate*, which is the ratio of the number of MUs lost to the total number of MUs generated, to investigate the efficiency of retransmission.

We have also assessed inter–stream synchronization quality in the simulation. As a result, we noticed that all the schemes have high inter–stream synchronization quality. Thus, we do not show the result.

The *average MU delay*, which is the average time of MU delay, is a key measure for live media.

IV. SIMULATION RESULTS

We first examine detrimental effects of the interference data load on the quality when $J_{\max}$ and $\Delta_{\rm al}$ are set to 100 ms and 300 ms, respectively. We then examine the effect of the buffering time for media synchronization on the application–level QoS by changing $J_{\max}$ and $\Delta_{\rm al}$ when the average rate of interference data transmitted by LS1 (i.e., interference data 1) is 1.0 Mbps, and LS2 transmits the interference data (i.e., interference data 2) at 1.7 Mbps.

[1] In [8], JPEG is employed for video codec. On the other hand, this paper handles MPEG video. However, because of the GOP pattern in this paper, we have assumed that the capturing and encoding delay time of each MU is approximately the same as that of JPEG video in [8].

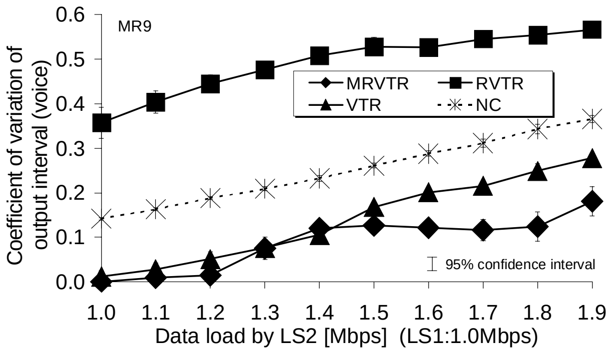

Fig. 2. Coefficient of variation of output interval for voice versus data load at MR9.

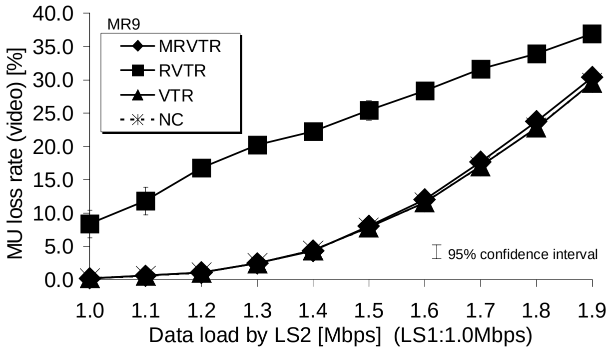

Fig. 3. MU loss rate of video versus data load at MR9.

We compare the application–level QoS of four schemes: MRVTR, RVTR, VTR and *NC (No Control)*. VTR exerts the enhanced VTR algorithm in [8] with no retransmission. NC means that neither retransmission control nor media synchronization control is carried out.

In the comparison, we focus on the application–level QoS at MR1 and MR9. This is because the application–level QoS at MR1 is the same as that at MR2 through MR8, while the application–level QoS at MR9 is the same as that at MR10 through MR16.

In this paper, each symbol in the figures to be shown represents the average of 10 measured values which were obtained by changing the random seed for generating the interference traffic. We also show 95 % confidence intervals of the QoS parameters in the figures. However, when the interval is smaller than the size of the corresponding symbol representing the simulation result, we do not show it in the figures.

Effect of Data Load

In this section, we keep the amount of interference data 1 at 1.0 Mbps and change that of interference data 2 from 1.0 Mbps to 1.9 Mbps.

We first show the application–level QoS at MR9, which is inferior to that at MR1 because of the difference in the interference traffic. We then present the application–level QoS at MR1.

Results of Terminal MR9: We present the simulation results at MR9 in Figs. 2 and 3. Figure 2 shows the coefficient of variation of output interval for voice as a function of data load (interference data 2). Figure 3 displays the MU loss rate of video versus data load (interference data 2).

We can confirm in Fig. 2 that for all the data loads here, RVTR has the largest coefficient of variation of output interval for voice among all the schemes. The reason is as follows. In RVTR, whenever each destination detects

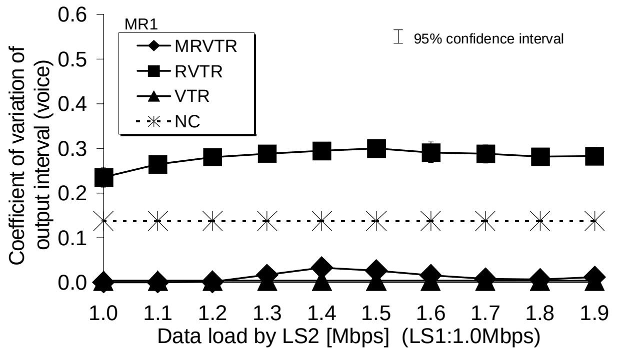

Fig. 4. Coefficient of variation of output interval for voice versus data load at MR1.

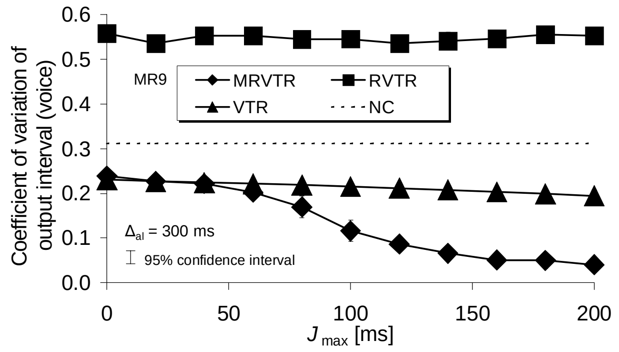

Fig. 5. Coefficient of variation of output interval for voice versus $J_{\max}$ at MR9.

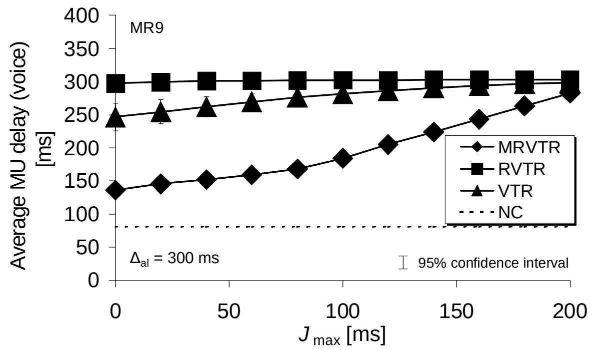

Fig. 6. Average MU delay of voice versus $J_{\max}$ at MR9.

some lost MU of which the *retransmission deadline* [8] is not reached, it sends a NACK to the source. The retransmission deadline is the target output time of a MU which will be output to the next of the missing MU. In the network configuration, eight duplicate NACKs may arrive at the source successively, because eight destinations may detect MU loss at the same time. The source retransmits a requested MU immediately upon getting a NACK, if it judges that the MU can arrive at the destination by its retransmission deadline. Thereby, the source may successively retransmit eight duplicate MUs, which become interference traffic for newly transmitted MUs. Therefore, the coefficient of variation with RVTR is large.

On the other hand, in Fig. 2, we see that the coefficient of variation of output interval for voice with MRVTR is the smallest among all the schemes for the data loads lighter than about 1.2 Mbps or heavier than around 1.5 Mbps. This is because MRVTR can effectively reduce the number of retransmitted video MUs by the NACK control scheme. Hence, MRVTR can use more bandwidth for retransmission of voice MUs than RVTR. Thus, MRVTR can recover lost voice MUs effectively.

In addition, we find in Fig. 2 that VTR gives smaller coefficients of variation for voice than MRVTR for the data loads of around 1.4Mbps. This is because MRVTR judges that the network is LOADED under this condition, and then the source retransmits video MUs. Thus, the retransmission control may cause the degradation of the application–level QoS. However, the difference between the two schemes is negligible.

We find in Fig. 3 that in the whole range of the data load considered here, RVTR has the highest MU loss rate of video among all the schemes. Figure 3 also reveals that for the data loads heavier than around 1.7 Mbps, the MU loss rate of video with MRVTR is a little larger than the loss rates with VTR and NC. This is because the retransmitted voice MUs affect the video MU transfer in MRVTR.

Results of Terminal MR1: Figure 4 presents the coefficient of variation of output interval for voice at MR1 as a function of data load (interference data 2).

We notice in Fig. 4 that in the whole range of the data load considered here, RVTR has the largest coefficient of variation of output interval for voice among all the schemes. This is due to interference of retransmitted video MUs which are requested by MR9 through MR16. On the other hand, MRVTR has much smaller coefficients of variation for voice than RVTR, since MRVTR can reduce the number of retransmitted video MUs requested by MR9 through MR16 effectively.

Furthermore, in Fig. 4, the coefficient of variation for voice with VTR takes an approximately constant value of 0. This is because VTR has no retransmission mechanism, and then there is no interference of retransmitted video MUs. Therefore, newly transmitted voice MUs hardly drop at MR1.

We have also confirmed the feasibility of MRVTR by a simple experiment, in which two destinations receive multicast live media.

Effect of $J_{\max}$ and $\Delta_{\rm al}$

In this section, we examine the effect of the buffering time for media synchronization on the application–level QoS by changing $J_{\max}$ and $\Delta_{\rm al}$. First, we vary $J_{\max}$ from 0 ms to 200 ms with $\Delta_{\rm al}$ fixed at 300 ms. In addition, we set $J_{\max}$=100 ms and change $\Delta_{\rm al}$ from 150 ms to 400 ms; ITU–T Recommendation G.114 regards delays of this range as acceptable provided that Administrations are aware of the transmission time impact on the transmission quality of user applications.

Results of Terminal MR9: We present the simulation results at MR9 in Figs. 5 through 8. Figure 5 shows the coefficient of variation of output interval for voice as a function of $J_{\max}$ in the case of $\Delta_{\rm al}$=300 ms. Figure 6 displays the average MU delay of voice in the same way as that of Fig. 5. Furthermore, Fig. 7 plots the coefficient of variation of output interval for voice versus $\Delta_{\rm al}$ when $J_{\max}$ is set to 100 ms, and Fig. 8 presents the average MU delay of voice likewise.

In Fig. 5, we notice that for all values of $J_{\max}$ here, the coefficient of variation of output interval for voice with RVTR remains almost constant and the largest among all the schemes. In addition, we can confirm in Fig. 6 that the average MU delay of voice with RVTR takes an approximately constant value of 300 ms, which is equal to $\Delta_{\rm al}$. The reason is as follows. In RVTR, voice MU loss occurs frequently because of many retransmitted video MUs; accordingly, the buffering time increases largely from the initial buffering time, which increases as $J_{\max}$ increases, to absorb delay jitter and provide extra time for retransmission. However, owing to the constraint of $\Delta_{\rm al}$, the buffering time cannot exceed the limit and therefore saturates.

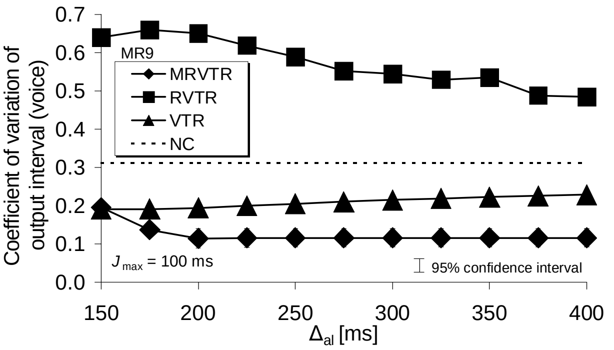

Fig. 7. Coefficient of variation of output interval for voice versus $\Delta_{\rm al}$ at MR9.

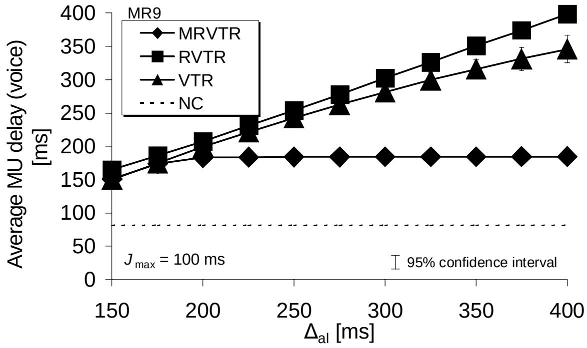

Fig. 8. Average MU delay of voice versus $\Delta_{\rm al}$ at MR9.

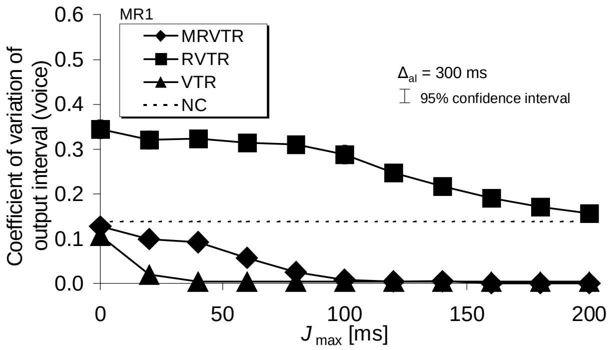

Fig. 9. Coefficient of variation of output interval for voice versus $J_{\max}$ at MR1.

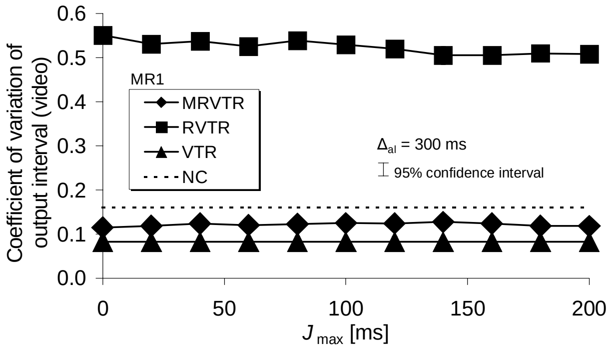

Fig. 10. Coefficient of variation of output interval for video versus $J_{\max}$ at MR1.

Thus, the coefficient of variation for voice with RVTR is large, and the average MU delay of voice remains constant at about $\Delta_{\rm al}$.

We also find in Fig. 5 that the coefficient of variation of output interval for voice with MRVTR decreases as $J_{\max}$ increases. Even in congested situations, MRVTR can recover dropped voice MUs with high probability since MRVTR does not retransmit video MUs; accordingly, the buffering time does not increase so largely from the initial buffering time. That is, $J_{\max}$ dominates the buffering time in MRVTR. Therefore, MRVTR can absorb delay jitter effectively as $J_{\max}$ increases. In fact, in Fig. 6, the average MU delay of voice with MRVTR becomes large as $J_{\max}$ increases.

In Fig. 7, we find that the coefficient of variation of output interval for voice with RVTR decreases as $\Delta_{\rm al}$ increases. This is because the buffering time increases as $\Delta_{\rm al}$ increases in RVTR. However, even if we set $\Delta_{\rm al}$ to 400 ms, RVTR has the largest coefficient of variation among all the schemes.

Figure 7 also reveals that when $\Delta_{\rm al}$ is larger than 200 ms, the coefficient of variation for voice with MRVTR keeps its value at approximately 0.1. Furthermore, Fig. 8 shows that the average MU delay of voice with MRVTR remains almost constant for $\Delta_{\rm al}$ larger than 200 ms. Thus, in MRVTR, $\Delta_{\rm al}$ does not affect the buffering time so much.

We have also investigated the coefficient of variation of output interval for video at MR9. However, we saw that neither $J_{\max}$ nor $\Delta_{\rm al}$ affects the coefficient of variation for video so much in all the schemes.

Results of Terminal MR1: We present the simulation results at MR1 in Figs. 9 through 13. Figures 9 and 10 show the coefficient of variation of output interval for voice and that for video, respectively, as a function of $J_{\max}$ in the case of $\Delta_{\rm al}$=300 ms. Figure 11 displays the average MU delay of voice in the same way as that of Fig. 9. Furthermore, Fig. 12 plots the MU loss rate of video versus $\Delta_{\rm al}$ when $J_{\max}$ is set to 100 ms, and Fig. 13 presents the average MU delay of voice likewise. We do not show the average MU delay of video here, since it is approximately the same as that of voice in all the schemes except for NC because of high quality of inter–stream synchronization.

In Fig. 9, we observe that the coefficients of variation of output interval for voice with MRVTR, RVTR and VTR decrease as $J_{\max}$ increases. We also see in this figure that when we set $J_{\max}$ to a larger value than 40 ms in VTR and 100 ms in MRVTR, the coefficient of variation becomes almost 0. The reason is as follows. The data load at MR1 is not so heavy as that at MR9; accordingly, even in RVTR, the buffering time does not increase largely from the initial buffering time. Hence, the buffering time in MRVTR, RVTR and VTR increases as $J_{\max}$ increases at MR1. Therefore, the coefficients of variation with these schemes at MR1 decrease as $J_{\max}$ increases.

In Fig. 10, we notice that the coefficient of variation of output interval for video with MRVTR and that with VTR keep small and approximately constant values. This is due to no retransmission of video MUs in these schemes. In MRVTR, under this condition, MR1 through MR8 judge that the network is UNLOADED, while it is judged CONGESTED by MR9 through MR16; accordingly, all the destinations do not send NACKs for lost video MUs. On the other hand, we can confirm in this figure that the coefficient of variation with RVTR slightly decreases as $J_{\max}$ increases.

We find in Fig. 11 that the average MU delay of voice with RVTR does not saturate owing to the constraint of $\Delta_{\rm al}$=300 ms. This is because the buffering time of RVTR at MR1 does not become so large.

In Fig. 12, we can confirm that the MU loss rate of video with RVTR increases as $\Delta_{\rm al}$ increases. In addition, we find in Fig. 13 that the average MU delay of voice with RVTR increases as $\Delta_{\rm al}$ increases. This is because the buffering time with RVTR at MR9 through MR16 increases as $\Delta_{\rm al}$ increases. When the buffering time at MR9 through MR16

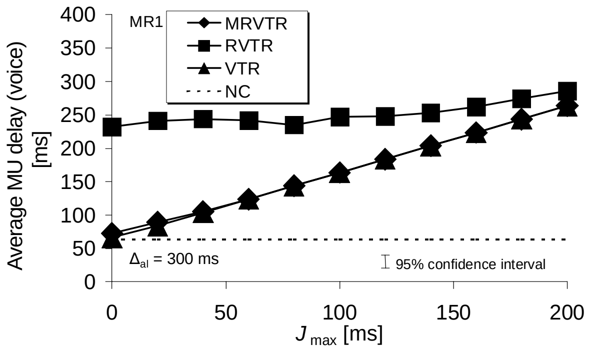

Fig. 11. Average MU delay of voice versus $J_{\max}$ at MR1.

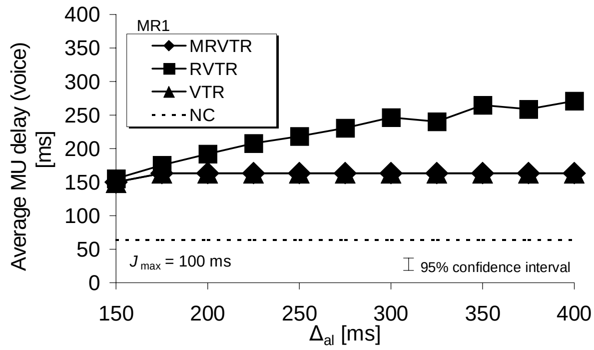

Fig. 13. Average MU delay of voice versus $\Delta_{\rm al}$ at MR1.

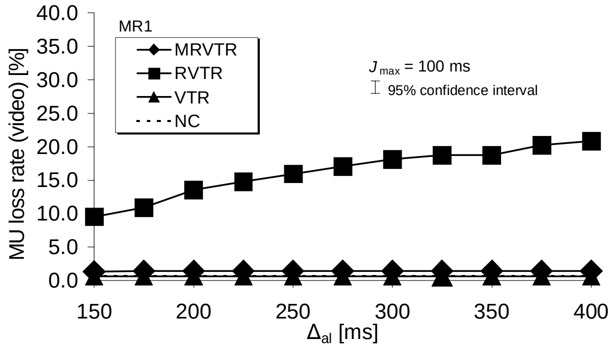

Fig. 12. MU loss rate of video versus $\Delta_{\rm al}$ at MR1.

becomes large, the source judges that MUs requested by MR9 through MR16 can arrive at the destination by the retransmission deadlines and then retransmits them. This causes many retransmitted MUs which become interference traffic of newly transmitted MUs; thus, many MUs drop even at MR1. Therefore, the MU loss rate of video and the average MU delay of voice with RVTR increase as $\Delta_{\rm al}$ increases.

Figure 12 also reveals that for all values of $\Delta_{\rm al}$ here, the MU loss rate of video with MRVTR is much smaller than that with RVTR and keeps an approximately constant value. Furthermore, Fig. 13 shows that the average MU delay of voice with MRVTR remains almost constant for $\Delta_{\rm al}$ larger than 175 ms. This is because $\Delta_{\rm al}$ does not affect the buffering time so much in MRVTR at all the destinations.

V. CONCLUSIONS

In this paper, we enhanced RVTR to cope with multicast communications; it is called MRVTR. Then, we examined the effectiveness of MRVTR in terms of application–level QoS by simulation. As a result, we saw that MRVTR is superior to RVTR and the scheme with no control (NC) in terms of the intra–stream synchronization quality of voice. In addition, we noticed that MRVTR not only reduces the interference of retransmitted traffic at the destinations which are lightly loaded but also improves the application–level QoS at the destinations with heavy data loads.

Furthermore, we investigated the relationship between the buffering time for media synchronization and application–level QoS of MRVTR by changing the parameters of the enhanced VTR algorithm, which is employed by MRVTR. As a result, we saw that the increase in $J_{\max}$ improves the media synchronization quality of MRVTR, though it increases MU delays. We also noticed that although MRVTR incurs increment of the average MU delay, its upper bound is controllable by setting $\Delta_{\rm al}$ to a desirable value; this is also the case with RVTR and VTR.

As the next step of our research, we plan to assess the user–level QoS of MRVTR. We also need to investigate the relationship between the user–level QoS and the application–level QoS.

ACKNOWLEDGMENT

This work was supported by the Grant–In–Aid for Scientific Research of Japan Society for the Promotion of Science under Grant 14350200 and the Nitto Foundation.

REFERENCES

[1] K. Obraczka, “Multicast transport protocols: A survey and taxonomy,” *IEEE Commun. Mag.*, vol.36, no.1, pp.94–102, Jan. 1998.
[2] S. Paul, K. K. Sabnani and J. C. Lin, “Reliable multicast transport protocol (RMTP),” *IEEE J. Sel. Areas in Commun.*, vol.15, no.3, pp.407–421, Apr. 1997.
[3] X. R. Xu, A. C. Myers, H. Zhang and R. Yavatkar, “Resilient multicast support for continuous–media application,” in *Proc. NOSSDAV’97*, May 1997.
[4] X. Li, S. Paul, P. Pancha and M. Ammar, “Layered video multicast with retransmission (LVMR): Evaluation of error recovery schemes,” in *Proc. NOSSDAV’97*, May 1997.
[5] B. Whetten and G. Taskale, “An overview of reliable multicast transport protocol II,” *IEEE Network*, vol.14, no.1, pp.37–47, Jan./Feb. 2000.
[6] G. Carle and E. W. Biersack, “Survey of error recovery techniques for IP–based audio–visual multicast applications,” *IEEE Network*, vol.11, no.6, pp.24–36, Nov./Dec. 1997.
[7] S. Tasaka and Y. Ishibashi, “Mutually compensatory property of multimedia QoS,” in *Conf. Rec. IEEE ICC 2002*, pp.1105–1111, Apr. 2002.
[8] S. Tasaka, T. Nunome and Y. Ishibashi, “Live media synchronization quality of a retransmission–based error recovery scheme,” in *Conf. Rec. IEEE ICC 2000*, pp.1535–1541, June 2000.
[9] Y. Ishibashi and S. Tasaka, “A synchronization mechanism for continuous media in multimedia communications,” in *Proc. IEEE INFOCOM’95*, pp.1010–1019, Apr. 1995.
[10] S. Tasaka, H. Nakanishi and Y. Ishibashi, “Dynamic resolution control and media synchronization of MPEG in wireless LANs,” in *Conf. Rec. IEEE GLOBECOM’97*, pp.138–144, Nov. 1997.
[11] D. Towsley, J. Kurose and S. Pingali, “A comparison of sender–initiated and receiver–initiated reliable multicast protocols,” *IEEE J. Sel. Areas Commun.*, vol.15, no.3, pp.398–406, Apr. 1997.
[12] S. Pejhan, M. Schwartz and D. Anastassiou, “Error control using retransmission schemes in multicast transport protocols for real–time media,” *IEEE/ACM Trans. Networking*, vol.4, no.3, pp.413–427, June 1996.
[13] S. Floyd, V. Jacobson, C. Liu, S. McCanne and L. Zhang, “A reliable multicast framework for light–weight sessions and application level framing,” *IEEE/ACM Trans. Networking*, vol.5, no.6, pp.784–803, Dec. 1997.
[14] The network simulator – ns–2 –, (http://www.isi.edu/nsnam/ns/).
[15] A. Adams, J. Nicholas and W. Siadak, “Protocol independent multicast – dense mode (PIM–DM): protocol specification,” Internet Draft, draft-ietf-pim-dm-new-v2-01, Feb.2002.