「マルチメディア通信と分散処理ワークショップ」 平成17年11月

Flexible Multimedia Streaming Model in Heterogeneous Networks

Satoshi Itaya†, Naohiro Hayashibara†, Tomoya Enokido††, and Makoto Takizawa†
†Dept. of Computers and Systems Engineering, Tokyo Denki University, Japan
††Faculty of Business Administration, Rissho University, Japan
Email: †{itaya, haya, taki}@takilab.k.dendai.ac.jp, ††eno@ris.ac.jp

Abstract

In a peer-to-peer (P2P) overlay network, a large number and various types of peer processes are cooperating by using multimedia contents like movies. Multimedia streaming is a key technology to realize multimedia applications. Here, multimedia contents are required to be reliable and continuously delivered to processes in a real-time manner. In this paper, we newly discuss a heterogeneous asynchronous multi-source streaming (HAMS) model where multiple contents peers transmit packets of a multimedia content to a requesting leaf peer to increase the throughput, reliability, and scalability in P2P overlay networks.

1. Introduction

Multimedia streaming applications like video on demand [6] are getting more significant in the Internet applications [7]. Here, multimedia contents have to be efficiently and reliably delivered to users from contents providers while real-time constraints are satisfied. In peer-to-peer (P2P) overlay networks [5], a large number of peer processes (*peers*) in various types of computers, mainly personal computers are cooperating by exchanging messages with each other. Here, multimedia contents are in nature distributed in various ways like downloading. Peers supporting multimedia contents are *contents* peers. On the other hand, peers which receive multimedia contents are *leaf* peers. One-to-one/one-to-many types of communication protocols like TCP [4] and RTP [8] are so far developed and widely used for multimedia applications. One-to-one/one-to-many protocols to satisfy Quality of Service (QoS) requirements are also discussed in papers [9].

In this paper, we newly discuss a *heterogeneous asynchronous multi-source streaming* (HAMS) model. Here, each communication channel may support different QoS and each peer may support different transmission rate. Packets of a multimedia content are in parallel transmitted to a leaf peer from multiple contents peers. Every contents peer asynchronously starts transmitting a subsequence of the packets to each leaf peer independently of the others. Each contents peer autonomously selects some packets of the multimedia content by exchanging information on what packets they have sent with others.

In section 2, we present a system model. In section 3, we discuss how to decompose a multimedia content to subsequences of packets. In section 4, we discuss the HAMS model. In section 5, we evaluate the HAMS model in terms of throughput.

2. Multi-source Streaming (MSS) Model

We consider multimedia streaming applications [3, 7]. Applications are realized by cooperation of multiple peers by exchanging multimedia data with other peers. Peers are interconnected in underlying networks. A *packet* is a unit of data transmission in the underlying network. A multimedia content is decomposed into a sequence of packets and packets are transmitted in a network.

First, a *leaf* peer sends a request of a content C to a *contents* peer. On receipt of the request, a contents peer starts transmitting a sequence of packets of the content C to the leaf peer. One contents peer typically supports multiple leaf peers and transmits packets of the multimedia content to each leaf peer asynchronously with the other leaf peers. This model is referred to as *single-source streaming* (SSS) model.

In order to realize the higher scalability, reliability, and throughput, a *multi-source streaming* (MSS) model is discussed [1]. Here, multiple contents peers are used to deliver a multimedia content to each leaf peer. Let $\mathbf{CP}_C$ be a set of contents peers $CP_1, \ldots, CP_n$ ($n \geq 1$) of a content C. Let $\mathbf{LP}_C$ be a set of leaf peers $LP_1, \ldots, LP_m$ ($m \geq 1$) which request a content C. Multiple contents peers $CP_1, \ldots, CP_n$ send packets of the content C to a leaf peer LP_s [Figure 1]. Let CL_{is} shows a logical channel between CP_i and LP_s. A channel CL_{is} is characterized in Quality of Service (QoS), bandwidth bw_{is}, delay time dl_{is}, and packet loss ratio pl_{is}.

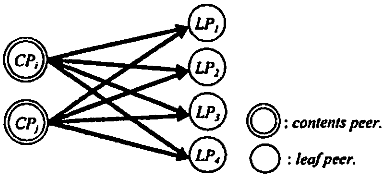


Figure 1. Multi-source streaming model.

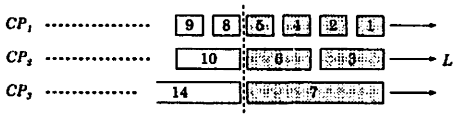
3. Packet Distribution to Multiple Channels

Suppose contents peers $CP_1, \ldots, CP_n$ ($n \geq 1$) send packets of a content C to a leaf peer LP_s. A packet is a unit of data transmission in an underlying network. In CP_i, a content C is decomposed into a sequence $pkt =$

$\langle t_1, \ldots, t_l \rangle$ of packets. Then, CP_i transmits the packets in the network. Suppose a sequence $pkt = \langle t_1, \ldots, t_8 \rangle$ of packets is obtained from a content C. Multiple contents peers $CP_1, \ldots, CP_n$ transmit packets in pkt to LP_s. Each CP_i transmits a subsequence pkt_{is} of pkt to LP_s. A *union* $pkt_1 \cup pkt_2$ is a packet sequence including every packet in packet sequences pkt_1 and pkt_2 where the packets are totally ordered in the sequence number and no redundant packets are included. Let $pkt\langle t_i]$ and $pkt[t_i\rangle$ show a prefix $\langle t_1, \ldots, t_i \rangle$ and postfix $\langle t_i, t_{i+1}, \ldots, t_l \rangle$ of a packet sequence pkt, respectively.

The larger bandwidth bw_{is} a channel CL_{is} implies, the more number of packets are transmitted through the channel CL_{is}. $|pkt_{is}| \geq |pkt_{js}|$ if the bandwidth bw_{is} from CP_i to LP_s is larger than the bandwidth bw_{is} of another CP_j. Next, we discuss which packets each contents peer CP_i transmits to LP_s. In the single-source streaming (SSS) model, one contents peer sends a sequence $t_1, t_2, \ldots$ of the packets to the leaf peer as shown in Figure 2a. In our multi-source streaming (MSS) model, each of CP_1, CP_2, and CP_3 transmits different packets of the content C from others as shown in Figure 2b. Each CP_i transmits packets at rate proportional to the bandwidth bw_{is}. The fastest contents peer CP_1 transmits four packets t_1, t_2, t_4, and t_5, the second fastest contents peer CP_2 transmits t_3 and t_6, and the slowest contents peer CP_3 transmits t_7 to LP_s, i.e. $pkt_{1s} = \langle t_1, t_2, t_4, t_5, \ldots \rangle$, $pkt_{2s} = \langle t_3, t_6, \ldots \rangle$, and $pkt_{3s} = \langle t_7, \ldots \rangle$. Here, $|pkt_{1s}| : |pkt_{2s}| : |pkt_{3s}| = 4 : 2 : 1$. First, LP_s receives the top packet t_1 from CP_1. Here, LP_s delivers t_1. Then, LP_s receives a pair of packets t_2 and t_3 from CP_1 and CP_2, respectively, at the same time. LP_s delivers t_2 and t_3. Then, LP_s receives t_4 from CP_1. LP_s delivers t_4 without waiting for other packets since every packet preceding t_4 has been delivered. On receipt of t_7 from the slowest contents peer CP_3, LP_s delivers t_5, t_6, and t_7. Here, a subsequence $\langle t_1, \ldots, t_7 \rangle$ of packets is referred to as *segment*. The next segment is $\langle t_8, \ldots, t_{14} \rangle$. Since packets are in parallel transmitted by CP_1, CP_2, and CP_3, the transmission time can be reduced.

a. SSS model.



b. MSS model.

Figure 2. Transmission of packets.

Data transmission in each channel CL_{is} from a contents peer CP_i to a leaf peer LP_s is modeled to be a sequence of time slots $CL_{is}^1, CL_{is}^2, \ldots, CL_{is}^{c_i}$ where the kth packet t_{is}^k in a subsequence $pkt_{is} = \langle t_{is}^1, t_{is}^2, \ldots, t_{is}^{c_i} \rangle$ can be transmitted in the kth time slot CL_{is}^k $(k = 1, \ldots, c_i)$ where c_i is the number of packets in pkt_{is}. Figure 3 shows time slots of the channels CL_{1s}, CP_{2s}, and CL_{3s}, where $4\tau_{1s} = 2\tau_{2s} = \tau_{3s}$ since $bw_{1s} : bw_{2s} : bw_{3s} = 4 : 2 : 1$. The larger the bandwidth bw_{is} of CL_{is} is, the shorter each CL_{is}^k is. The size τ_{is} [msec] shows the transmission time of a packet in CL_{is} with inter-packet gap. Let $st(CL_{is}^k)$ and $et(CL_{is}^k)$ show when CP_i starts and finishes transmitting the kth packet in pkt_{is}, respectively. First, $st(CL_{is}^0)$ is defined to be 0 for every CL_{is}. Then, $et(CL_{is}^{k+1}) = st(CL_{is}^k) + \tau_{is}$. $st(CL_{is}^{k+1}) = et(CL_{is}^k)$. Here, a time slot CL_{is}^k *precedes* another CL_{js}^h ($CL_{is}^k \rightarrow CL_{js}^h$) if $et(CL_{is}^k) < et(CL_{js}^h)$. Let $\mathbf{CL}$ be a set of all the time slots in $CL_{is}, \ldots, CL_{ns}$. A time slot CL in $\mathbf{CL}$ is *initial* iff there is no time slot CL' such that CL' precedes CL ($CL' \rightarrow CL$) in $\mathbf{CL}$.

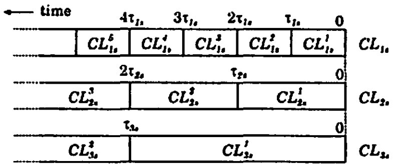


Figure 3. Time slots.

Packets in a packet sequence pkt are allocated to time slots of the set $\mathbf{CL} = \{CL_{1s}, \ldots, CL_{ns}\}$:

[Allocation of packets] For each packet t_k in a packet sequence pkt $(k = 1, \ldots, l)$,

1. Find an initial time slot CL such that $st(CL) \geq st(CL')$ for every initial time slot CL' in $\mathbf{CL}$.
2. Allocate the packet t_k with the time slot CL.
3. Remove the time slot CL from $\mathbf{CL}$.

Let us consider the channels CL_{1s}, CL_{2s}, and CL_{3s} shown in Figure 3. Each channel CL_{is} is modeled to be a sequence of time slots, $CL_{is}^1, CL_{is}^2, \ldots, CL_{is}^{c_i}$ for $i = 1, 2, 3$, i.e. $\mathbf{CL} = \{CL_{is}^1, CL_{is}^2, \ldots, CL_{is}^{c_i} \mid i = 1, 2, 3\}$. The time slots in $\mathbf{CL}$ are partially ordered in the precedent relation $\rightarrow$. According to the packet allocation algorithm, the initial time slot CL_{1s}^1 is first selected and the top packet t_1 in the sequence pkt is assigned with CL_{1s}^1. CL_{1s}^1 is removed from $\mathbf{CL}$. Next, there are a pair of initial time slots CL_{1s}^2 and CL_{2s}^1. Here, $st(CL_{1s}^2) > st(CL_{2s}^1)$ since the channel CL_{1s} is faster than CL_{2s} ($bw_{1s} > bw_{2s}$). CL_{1s}^2 is taken for the second packet t_2. CL_{1s}^2 is removed from $\mathbf{CL}$. Then, the initial time slot CL_{2s}^3 is taken for t_3. Thus, packets are assigned with time slots as shown in Figure 2b.

4. HAMS Model

4.1. Asynchronous coordination

Itaya *et al.* [1] discussed the asynchronous approach to synchronize transmission of packets from multiple contents peers. In the asynchronous coordination, each CP_i independently starts transmitting packets of the content C on receipt of a content request from a leaf peer LP_s. While transmitting packets to LP_s, each contents peer exchanges the *control* packets on which *content* packets have been sent and information on the bandwidth of a channel between the contents peer and the leaf peer.

4.2. Data structure

Each content packet t is identified by a unique sequence number $t.SQ$ in a packet sequence *pkt*. It is noted that each contents peer sends content packets to a leaf peer but the sequence numbers of the content packets may be gapped because each contents peer does not send every packet. Each contents peer CP_i perceives CP_j to be *active* if CP_i receives a control packet from CP_j. Otherwise, CP_i perceives CP_j to be *dormant*. Here, VW_i shows a *view* of CP_i, i.e. a subset of contents peers which CP_i perceives to be active. VW_i is realized in a bitmap $\langle V_1, \ldots, V_n \rangle$ where $V_j = 1$ if CP_i perceives CP_j to be active, otherwise $V_j = 0$ ($j = 1, \ldots, n$). Here, $VW_i.V_j$ shows the jth bit V_j in VW_i. $|VW_i|$ is $|\{CP_j \mid VW_i.V_j = 1\}|$, i.e. the number of active contents peers which CP_i perceives. In each CP_i, the following variables are manipulated to send content packets ($j = 1, \ldots, n$):

- SQ_j = sequence number of a content packet where CP_i knows that CP_j has sent every content packet t where $t.SQ \leq SQ_j$ to LP_s, initially 0.
- VW_j = view $\langle V_1, \ldots, V_n \rangle$ of CP_j.
- MVQ_{jk} = sequence number of a content packet where CP_j has known that CP_k sent every data packet t where $t.SQ \leq MVQ_{ik}$, initially 0.
- $MVQ = \{MVQ_{jk} \mid j, k = 1, \ldots, n\}$.
- $MinMVQ_j$ = sequence number where CP_j has known that every active contents peer sent every content packet t where $t.SQ \leq MinMVQ_j$.
- $MinMVQ = min(MinMVQ_1, \ldots, MinMVQ_n)$.
- BW_j = bandwidth of CP_j which CP_i knows.

The contents peer CP_i knows that every CP_j has transmitted every content packet t where $t.SQ \leq MinMVQ$. "$MVQ_{jk} = \top$" means that CP_j does not perceive CP_k to be active. $MinMVQ_i = min(SQ_1, \ldots, SQ_n)$. Each control packet c sent by CP_i carries informations; $c.SQ$ = vector of sequence numbers $\langle SQ_1, \ldots, SQ_n \rangle$ where each SQ_j is a sequence number of a content packet most recently sent by a contents peer CP_j which CP_i knows, $c.VW$ = view VW_i of CP_i, and $c.BW$ = bandwidth BW_i of CP_i.

4.3. Transmission of content and control packets

Every active contents peer knows that every content packet t where $t.SQ \leq MinMVQ$ has been surely sent by some contents peer. Here, even if CP_i had not sent some packet t where $t.SQ \leq MinMVQ$, CP_i does not need to send t since t has been surely sent by another contents peer. Here, CP_i can only send a content packet t where $t.SQ > MinMVQ$. Let $MaxBW$ show the maximum one in $MaxBW_1, \ldots, MaxBW_n$. CP_i is assumed to know the maximum bandwidth $MaxBW_j$ of every CP_j ($j = 1, \ldots, n$).

The faster contents peer CP_i is, the more number of packets CP_i transmits. The number of packets to be sent by each CP_i should be decided to be proportional to the bandwidth BW_i. BW_i may change due to congestions of the communication channel and overload of CP_i. It spends computation and communication resource to reallocate packets to each contents peer each time the bandwidth of some contents peer is changed. In order to reduce the overhead of the packet allocation, the contents peers are classified with respect to BW_i ($\leq MaxBW_i$) in each CP_i as follows:

[Classification of contents peers]

1. CP_j is classified into a class 0 if $BW_j = MaxBW$.
2. CP_j is classified into the class k if $2^{-k+1} > BW_j/MaxBW \geq 2^{-k}$ ($k \geq 1$).

Let K be the total number of classes of the contents peers. Let $class(CP_i)$ denote a class of a contents peer CP_i ($\in \{0, 1, \ldots, K-1$ ($K \geq 1$)\}). Let $\mathbf{C}_k$ be a set of contents peers of a class k ($k = 0, 1, \ldots, K-1$). If there are multiple contents peers in each class k, the contents peers in $\mathbf{C}_k$ are sorted in an ascending order of the identifiers. Let CPN_k (≥ 0) be the number $|\mathbf{C}_k|$ of active contents peers in a class k ($< K$). For each class k, there is a sequence BK_k of buckets $BK_{k0}, BK_{k1}, \ldots, BK_{kc_k}$ ($c_k = CPN_k - 1$). Each bucket B_{ki} ($i = 1, \ldots, c_k$) includes CPN_k (≥ 1) of content packets, where one content packet from each active contents peer of a class k. Let $MaxSQ$ shows the sequence number of the last content packet. For the sequence number $IniSQ$ of some content packet, content packets in a postfix $\langle t_{IniSQ}, t_{IniSQ+1}, \ldots, t_{MaxSQ} \rangle$ of the sequence *pkt* are allocated to the buckets as follows:

[Packet allocation PAlloc($IniSQ$, K, $MaxSQ$)]

```
$c_i := b_i := 0$ for every $i$; $k := 0$;
for $t_h$ in pkt
    ($h = IniSQ, IniSQ+1, \ldots, MaxSQ$){
    store $t_h$ in the bucket $BK_{kb_k}$; $c_k := c_k + 1$;
    if $c_k > CPN_k${ $b_k := b_k + 1$;
        if $k = K - 1, k := 0$;
        else if $b_k$ is even, $k := k + 1$;
            else if $k > 0, k := k - 1$; } }
```

According to the packet allocation algorithm **PAlloc**($IniSQ$, K, $MaxSQ$), content packets are first allocated to buckets of the fastest channel. Lastly, content packets are allocated to the slowest one. Here, a subsequence of the content packets allocated is a *segment*. Initially, $IniSQ = 1$. If VW_i is changed, content packets are reallocated to the buffers. A contents peer CP_i takes a sequence $BK_k = \langle BK_{k0}, BK_{k1}, \ldots, BK_{kc_k} \rangle$ of buckets if $k = class(CP_i)$. If $CPN_k = 1$, each bucket in BK_k includes one packet. CP_i sends a content packet for each bucket BK_{kr} ($r = 0, 1, \ldots,$

c_k). If $CPN_k > 1$, CPN_k of content packets are included in each bucket since each of CPN_k active contents peers in the class k sends one content packet in each bucket. Content packets in each bucket are sorted in the sequence number. The contents peers in $\mathbf{C}_k$ are sorted in the identifies. CP_i takes the vth packet in every bucket in BK_k if CP_i is the vth in $\mathbf{C}_k$.

By exchanging control packets among the contents peers, each contents peer CP_i detects whether every other contents peer is active or dormant. A control packet c sent by CP_j carries the view $c.VW$ (= VW_j) to CP_i. CP_i has a *consistent* view VW_i iff $VW_i = VW_j$ for every CP_j such that $VW_i.V_j = 1$. Even if another CP_j perceives CP_k to be active, CP_i may perceive CP_k to be dormant since CP_i has not received any control packet from CP_k.

[View change] Each time VW_i changes from inconsistent state to consistent state, CP_i changes the transmission procedure as follows:

1. Every content packet t where $t.SQ > IniSQ$ in *pkt* is allocated to the buckets BK_0, BK_1, $\ldots$, BK_{K-1} according to the **PAlloc**($IniSQ$, K, $MaxSQ$).
2. CP_i sends content packets from the buckets in the bucket sequence BK_k where k is a class of CP_i.

Even if some number of contents peers are dormant, the other active contents peers can deliver every data of a multimedia content to a leaf peer as presented in the preceding subsection. However, if more number of contents peers get dormant, the leaf peer cannot receive some content packets. Hence, a collection of active contents peers reallocate content packets to buckets. If VW_i is consistent, every active contents peer has the same view and bandwidth information. Next, each active CP_i has to find the sequence number SQ of the content packet on which every active contents peer makes an agreement. As discussed, every content packet t where $t.SQ \leq MinMVQ$ is surely sent by some contents peer. However, $MinMVQ$ may not be the same in every active contents peer. Hence, we take the following action in each CP_i:

1. Every CP_j in VW_i is classified to a class *class*(CP_j) by the classification algorithm. $M := \sum_{k=0}^{K-1} 2^{K-1-k} CPN_k$. M gives the size of a segment.
2. CP_i takes a content packet s named *synchronization* point, where $t.SQ = \gamma M$ such as an integer γ that $\gamma M \leq MinMVQ < (\gamma + 1)M$.
3. $IniSQ$ is a sequence number $s.SQ$ of a synchronization point s. CP_i reallocates every content packet t where $t.SQ \geq IniSQ$ to the buckets by **PAlloc**($IniSQ$, K, $MaxSQ$).

Suppose CP_i takes the synchronization sequence number $IniSQ_i$. Content packets where sequence numbers are larger than or equal to $IniSQ_i$ are reallocated to buckets according to the **PAlloc**($IniSQ_i$, K, $MaxSQ$). Here, in another CP_j, $IniSQ_j \neq IniSQ_i$. A condition $|IniSQ_j - IniSQ_i| = \alpha \cdot M$ surely holds for some integer constant α (≥ 0) and every pair of CP_i and CP_j. Suppose $IniSQ_j < IniSQ_i$. CP_j allocates every content packet t where $t.SQ \geq IniSQ_j$ by the **PAlloc**($IniSQ_j$, K, $MaxSQ$). Every packet t where $t.SQ \geq IniSQ_i$ is surely allocated to the buckets in CP_j in a same way as CP_i because M content packets are a unit of packet allocation. In Figure 2, $M = 7$. Hence, $t_1, t_8, t_{15}, \ldots$ can be synchronization points. Thus, each CP_i reallocates packets to buckets in the same way even if some packets which have been sent by another contents peer might be transmitted again. LP_s continuously receives packets from active contents peers without packet loss while the membership and performance of contents peers are changed.

5. Evaluation

We evaluate the HAMS model compared with the SSS model and the AMSS model. In this evaluation, three contents peers CP_1, CP_2, and, CP_3 transmit content packets of a multimedia video content C of one Gbytes to a leaf peer LP_s. We assume that the delay time of each channel CL_{is} between a pair of CP_i and LP_s is reliable and constant (i = 1, 2, 3). On the other hand, each channel CL_{is} between CP_i and LP_s may support different bandwidth bw_{is}. We consider three configurations c1, c2, and c3 of channels CL_{1s}, CL_{2s}, and CL_{3s} with the ratio $|bw_{1s}| : |bw_{2s}| : |bw_{3s}| = 4 : 2 : 1, 2 : 2 : 1$, and $1 : 1 : 1$, respectively [Figure 4]. The minimum bandwidth is denoted by 1 which means 10 [Mbps] in each configuration.

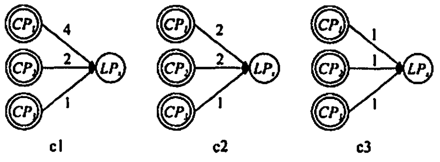


Figure 4. Configurations.

In the evaluation, a peer is realized in one process and processes are interconnected with logical channels in one computer (DELL Precision 650 with Linux 2.6.11-kernel OS, dual Intel Xeon 2.0 GHz CPU, and 1.5 GB main memory). Each CP_i transmits some number of packets for one time unit. The transmission rate [packet/time unit] of CP_i is given by $1/BW_{is}$. One content packet is 500 bytes long. CP_i transmits content packets of the video contents to a leaf peer LP_s. In the SSS model, one contents peer sends all the content packets to LP_s through the fastest channel in each configuration. In the AMSS model, each contents peer transmits content packets at the same rate. The rate is decided by the minimum bandwidth 10 [Mbps] in every channel. Each CP_i transmits content packets at the rate of the channel CL_{is} in the HAMS model.

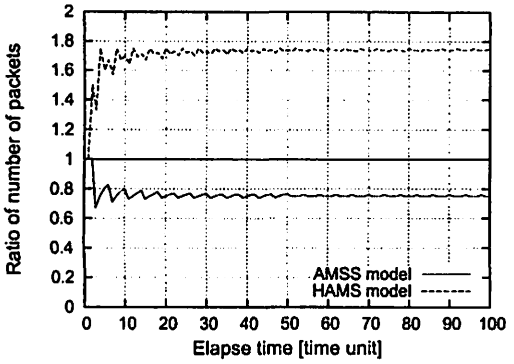


Figure 5. Ratio of number of packets (c1).

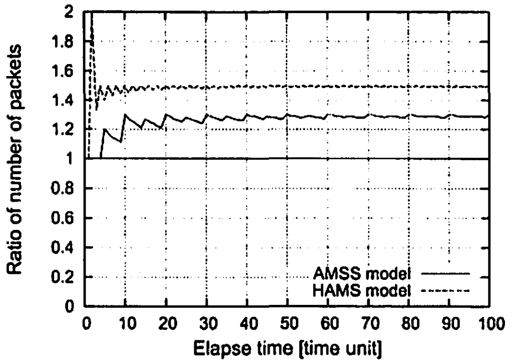


Figure 6. Ratio of number of packets (c2).

Figure 5 shows the configuration c1. About 70 % of the throughput is increased in the HAMS model for the SSS model since at most 70 [Mbps] rate is taken in the HAMS model while 40 [Mbps] in the SSS model. However, about 20 % of the throughput is decreased in the AMSS model since only the minimum bandwidth bw_{3s}, i.e. 10 [Mbps] of the slowest channel CL_{3s} can be used in each channel. Figure 6 shows the configuration c2. Here, both the HAMS and AMSS models imply the higher throughput than the SSS model. In the AMSS model, three channels are used to in parallel transmit content packets and the total bandwidth 30 [Mbps] used in the channels is larger than 20 [Mbps] of the fastest CL_{1s}. Figure 7 shows c3. Here, the HAMS and AMSS models support the same throughput. The HAMS and AMSS models imply three times higher bandwidth than the SSS model.

The AMSS model can support the higher throughput than the SSS model for the configurations c2 and c3 but the lower for c1. In c3, the AMSS and HAMS models support the same throughput since every channel supports the same bandwidth. In conclusion, the HAMS model can support multimedia streaming applications with the high throughput in heterogeneous environment.

6. Concluding Remarks

In this paper, we newly discussed the *heterogeneous asynchronous multi-source streaming* (HAMS) model for transmitting continuous multimedia contents from multiple contents peers to a leaf peer. The peers may not support enough computation power to distribute contents and enough QoS may not be supported in networks. In addition, each channel between contents peers and leaf peers may support different QoS. While transmitting content packets to leaf peers and exchanging control packets among contents peers, every active contents peer sends a different subsequence of content packets from the other contents peers to a leaf peer. In the evaluation, we showed that the HAMS model implies high-performance communication than the AMSS model [1] and the SSS model.

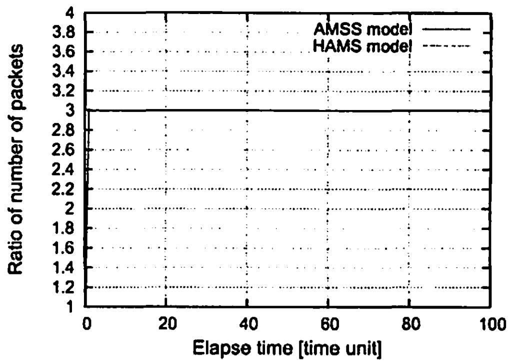


Figure 7. Ratio of number of packets (c3).

Acknowledgment

This research is partially supported by Research Institute for Science and Technology [Q05J-04] and Frontier Research and Development Center [16-J-6], Tokyo Denki University.

References

[1] S. Itaya, T. Enokido, and M. Takizawa. A High-performance Multimedia Streaming Model on Multi-source Streaming Approach in Peer-to-Peer Networks. In *Proc. of IEEE AINA-2005*, volume 1, pages 27–32, 2005.

[2] S. Itaya, N. Hayashibara, T. Enokido, and M. Takizawa. HAMS: Scalable Peer-to-Peer Multimedia Streaming Model in Heterogeneous Networks. *to appear in Journal of Computer and System Sciences*, 2005.

[3] Microsoft Windows Media Technology. http://www.microsoft.com/windows/windowsmedia/. 1998.

[4] J. Postel. Transmission Control Protocol. *RFC793*, 1981.

[5] Project JXTA. http://www.jxta.org/. 2001.

[6] P. V. Rangan, H. M. Vin, and S. Ramanathan. Designing an On-Demand Multimedia Service. *IEEE Communications Magazine*, 30(7):56–65, 1992.

[7] Real Networks. http://www.realnetworks.com/. 1996.

[8] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A Transport Protocol for Real Time Applications. *RFC1889*, 1996.

[9] T. Tojo, T. Enokido, and M. Takizawa. Notification-Based QoS Control Protocol for Multimedia Group Communication in High-Speed Networks. In *Proc. of IEEE ICDCS-24*, pages 644–651, 2004.