Distributed Coordination Protocols to Realize Scalable Multimedia Streaming in Peer-to-Peer Overlay Networks

Satoshi Itaya[1)], Naohiro Hayashibara[1)], Tomoya Enokido[2)], and Makoto Takizawa[1)]
[1)]Dept. of Computers and Systems Engineering, Tokyo Denki University
[2)]Faculty of Business Administration, Rissho University
Email: [1)]{itaya, haya, taki}@takilab.k.dendai.ac.jp, [2)]eno@ris.ac.jp

Abstract

Multimedia contents are distributed to peers in various ways in peer-to-peer (P2P) overlay networks. A peer which holds a content, even a part of a content can provide other peers with the content. Multimedia streaming is more significant in multimedia applications than downloading ways in Internet applications. We discuss how to support peers with multimedia streaming service by using multiple contents peers. In our distributed multi-source streaming model, a collection of multiple contents peers in parallel transmit packets of a multimedia content to a requesting leaf peer to realize the reliability and scalability without any centralized controller. Even if some peer stops by fault and is degraded in performance and packets are lost and delayed in networks, a requesting leaf peer receives every data of a content at the required rate. We discuss a pair of flooding-based protocols, distributed and tree-based coordination protocols DCoP and TCoP, to synchronize multiple contents peers to reliably and efficiently deliver packets to a requesting peer. A peer can be redundantly selected by multiple peers in DCoP but it taken by at most one peer in TCoP. We evaluate the protocols in terms of how long it takes and how many messages are transmitted to synchronize multiple contents peers.

1. Introduction

Multimedia streaming applications [9, 12, 13] like music streaming and movie on demand are getting more significant than downloading service in the Internet applications [1]. Here, multimedia contents have to be reliably delivered to users from providers of the contents while real-time constraints are satisfied. In peer-to-peer (P2P) overlay networks [2,11,16], multimedia contents are in nature distributed to peers in various ways like downloading and caching. Peers which have multimedia contents can support other peers with the contents. Peers supporting multimedia contents are *contents* peers. Peers which receive multimedia contents from contents peers are *leaf* peers. The contents-leaf relation is relative, i.e. each peer can be a contents or leaf peer.

New approaches to realizing multimedia streaming service in P2P overlay networks are discussed in multi-source streaming (MSS) models [5, 8] where multiple contents peers send packets of a content to a leaf peer. A large number of leaf peers are required to be supported and even a low-performance personal computer can support a content. In one approach to synchronizing multiple contents peers in the MSS model, one contents peer is a controller and the other contents peers transmit packets of a content to a leaf peer according to the order of the controller [5, 8]. Itaya *et al.* discuss a centralized coordination protocol [5] similar to the two-phase commitment (2PC) protocol [14]. It takes at least three rounds to synchronize multiple contents peers. Then, the contents peers can start transmitting packets of the content to a leaf peer. Liu and Voung [8] discuss a protocol where a requesting leaf peer sends a transmission schedule of a content to multiple contents peers. Each contents peer synchronously starts transmitting packets according to the schedule. Although it is simple to implement the MSS model in the centralized approach, it takes time to exchange messages to synchronize multiple contents peers and collect states of multiple contents peers.

In the asynchronous multi-source streaming (AMS) models [3–5], each of multiple contents peers asynchronously starts transmitting packets to a leaf peer and sends only a part of a multimedia content different from other contents peers. Here, every contents peer is, possibly periodically exchanging state information on which packets it has sent with all the other contents peers by using a simple type of group communication protocol [10]. The large communication overhead is implied since every contents peer sends state information to all the contents peers. In this paper, we take a gossip-based flooding protocols [6, 7] to reduce the communication overhead. First, a leaf peer sends a content request to some number of contents peers. Then, a contents peer starts transmitting packets to the leaf peer on receipt of the content request. Here, the contents peer selects some number of contents peers and sends a content request to the selected contents peers. There are two algorithms; a contents peer may be selected multiple peers and is selected by at most one peer. The former is a redundant type named *distributed coordination protocol* (DCoP) and the latter is a non-redundant type named *tree-*

based coordination protocol (TCoP). A content request carries information on which packets the contents peer has sent to a leaf peer at what rate. Each of the selected peers makes a decision on which packets to be sent. In addition, parity packets for some number of packets are transmitted so that a leaf peer can receive every data in a content even if some number of packets are lost and contents peers are faulty.

In section 2, we discuss how to allocate packets of a content to contents peers in heterogeneous environment. In section 3, we discuss DCoP and TCoP. In Section 4, we evaluate the coordination protocols in terms of how long it takes to synchronize all the contents peers and how many redundant packets are transmitted.

2. Multi-source Streaming (MSS) Models

Multimedia contents are distributed to peers in various ways like downloading and caching in a peer-to-peer (P2P) overlay network. For example, a peer obtains a free movie from an acquaintance peer by downloading and then supports some part of the movie to other peers. A *contents* peer which holds a multimedia content, even a part of the content can send the content to other peers. A peer receiving a content from a contents peer is a *leaf* peer. Each peer can play any role of contents and leaf ones. A contents peer may not support enough transmission rate due to the limited resource, degradation of quality of service (QoS), and faults in networks.

One contents peer transmits packets of a multimedia content to a leaf peer on request from the leaf peer. This is a traditional *single-source streaming* model but the contents peer is a single point of failure and performance bottleneck. In order to support a large number of leaf peers, a contents peer is required to be realized in a high-performance, expensive server computer. A multi-source streaming model [3–5] is proposed to realize the higher scalability and reliability of streaming service by using personal computers in a P2P overlay network. Here, a system is composed of multiple contents peers $CP_1, \ldots, CP_n$ ($n \geq 1$) supporting a multimedia content C and multiple leaf peers $LP_1, \ldots, LP_m$ ($m \geq 1$) which would like to use the content C, i.e. see the movie content. A pair of a contents peer CP_i and a leaf peer LP_s are interconnected in a logical communication channel CC_i of the underlying network. A packet is a unit of data transmission in an underlying network. A content is decomposed into a sequence of packets. Multiple contents peers $CP_1, \ldots, CP_n$ in parallel transmit packets of a content to each leaf peer LP_s in the MSS model.

Each contents peer CP_i ($i = 1, \ldots, n$) sends a part of a sequence pkt of packets $\langle t_1, \ldots, t_l \rangle$ ($l \geq 1$) of a multimedia content C to a leaf peer LP_s. Here, $|pkt| = l$. Suppose three contents peers CP_1, CP_2, and CP_3 transmit packets in a packet sequence $pkt = \langle t_1, \ldots, t_8 \rangle$ of a content C to LP_s where $bw_1 : bw_2 : bw_3 = 4 : 2 : 1$. Each CP_i transmits a subsequence pkt_i of the packet sequence pkt to LP_s. $|pkt_i| \geq |pkt_j|$ if $bw_i \geq bw_j$. For example, the fastest CP_1 transmits

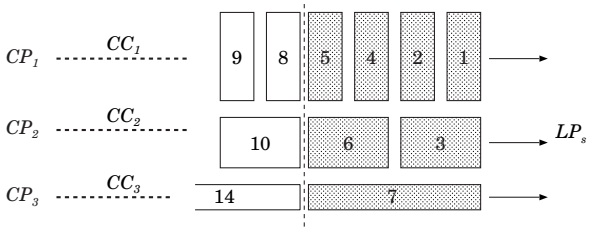

Figure 1. Multi-source streaming (MSS).

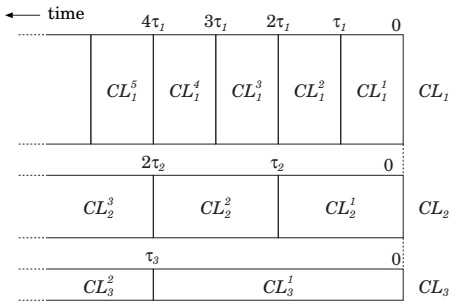

Figure 2. Time slots.

packets t_1, t_2, t_4, and t_5, CP_2 transmits t_3 and t_6, and CP_3 transmits t_7 to LP_s for one time unit, i.e. $pkt_1 = \langle t_1, t_2, t_4, t_5, \ldots \rangle$, $pkt_2 = \langle t_3, t_6, \ldots \rangle$, and $pkt_3 = \langle t_7, \ldots \rangle$ as shown in Figure 1. $|pkt_1| : |pkt_2| : |pkt_3| = 4 : 2 : 1$.

A *union* $pkt_1 \cup pkt_2$ is a packet sequence including every packet in a pair of sequences pkt_1 and pkt_2. For example, $pkt_1 \cup pkt_2 \cup pkt_3 = \langle t_1, t_2, t_3, t_4, t_5, t_6, t_7, t_8 \rangle$. An intersection $pkt_1 \cap pkt_2$ is a sequence of packets which are included in both pkt_1 and pkt_2. 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 $pkt = \langle t_1, \ldots, t_l \rangle$, respectively.

Data transmission in a channel CC_i is modeled to be a sequence of time slots $CL_i^1, CL_i^2, \ldots, CL_i^{c_i}$ ($c_i \geq 1$) where the kth packet t_i^k in a subsequence $pkt_i = \langle t_i^1, t_i^2, \ldots, t_i^{c_i} \rangle$ can be transmitted in the kth time slot CL_i^k where $c_i = |pkt_i|$. Let τ_i be the length of a time slot, which shows time for transmitting a packet in CC_i. $bw_1 : bw_2 : bw_3 = \tau_1 : \tau_2 : \tau_3 = 4 : 2 : 1$. Figure 2 shows time slots of the channels CC_1, CC_2, and CC_3. $st(CL_i^k)$ and $et(CL_i^k)$ show when CP_i starts and finishes transmitting the kth packet t_i^k in pkt_i, respectively. $st(CL_i^0)$ is 0 and $et(CL_i) = st(CL_i^k) + \tau_i = st(CL_i^{k+1})$ for every CC_i. Here, CL_i^k *precedes* CL_j^h ($CL_i^k \rightarrow CL_j^h$) if $et(CL_i^k) < et(CL_j^h)$. Let **CL** be a set of all the time slots in $CC_1, \ldots, CC_n$. Time slots in **CL** are partially ordered in $\rightarrow$. A time slot CL is *initial* iff there is no time slot CL' such that $CL' \rightarrow CL$ in **CL**. Packets are allocated to time slots as follows:

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

1. Find an initial time slot CL such that $st(CL) \geq st(CL')$

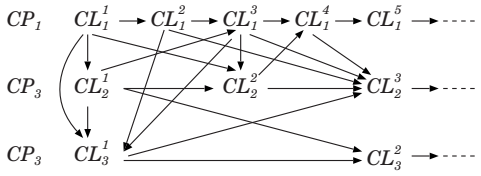

Figure 3. Precedence of time slots.

for every initial time slot CL' in the time slot set **CL**.

2. Allocate the packet t_k with the time slot CL and remove CL from **CL**.

From Figure 3, packets are allocated to time slots as shown in Figure 1. A leaf peer LP_s can deliver a packet t_h without waiting for any packet of $t_1, \ldots, t_{h-1}$ since $t_1, \ldots, t_{h-1}$ preceding t_h are surely delivered on receipt of t_h.

[Packet allocation property] On receipt of a packet t_h, a leaf peer LP_s receives every packet t_k preceding t_h in a packet sequence $pkt = \langle t_1, \ldots, t_l \rangle$.

3. Distributed Coordination Protocols

3.1. Types of distributed coordinations

Multiple contents peers $CP_1, \ldots, CP_n$ are required to cooperate to reliably deliver packets of a content C to a leaf peer LP_s. We take a distributed approach [3, 4] where each contents peer CP_i independently starts transmitting packets on receipt of a content request from LP_s. Here, we assume each CP_i supports the same transmission rate to LP_s. Let pkt be a sequence of packets $t_1, \ldots, t_l$ of a multimedia content C. While transmitting packets of C to LP_s, each CP_i informs the other contents peers of which packets CP_i has sent at what rate and the view showing which contents peer CP_i perceives to be active.

In the first broadcast way [5], a leaf peer LP_s broadcasts a content request of a multimedia content C to all the contents peers $CP_1, \ldots, CP_n$ [Figure 4(1)]. On receipt of the request, every CP_i starts transmitting packets in the packet sequence pkt of the content C to LP_s. Here, LP_s receives the most redundantly each packet from every contents peer. While transmitting packets to LP_s, each CP_i exchanges control packets with the other contents peers in a simple type of group communication protocol. Control packets carrying service information on CP_i, i.e. which packets CP_i has most recently sent, view showing which contents peers are perceived to be active, and bandwidth to LP_s. On receipt of a control packet, each CP_i changes the transmission schedule on which packets to be sent at what rate. It takes one round for every contents peer to start transmitting packets to LP_s. However, LP_s may lose packets due to the buffer overrun. In addition, CP_i sends a control packet with the service information to every contents peer. This way implies large overhead for communication among contents peers.

In the second unicast way, a leaf peer LP_s sends a content request to only one contents peer, say CP_1 [Figure 4(2)]. Then, CP_1 starts transmitting packets to LP_s and sends a control packet to another contents peer, say CP_2 to inform what packet CP_1 has sent. On receipt of the control packet, CP_2 starts transmitting packets to LP_s and sends a control packet to CP_3. Finally, CP_n starts transmitting packets to LP_s. This implies the minimum redundancy but it takes the longest time all the contents peers to synchronize.

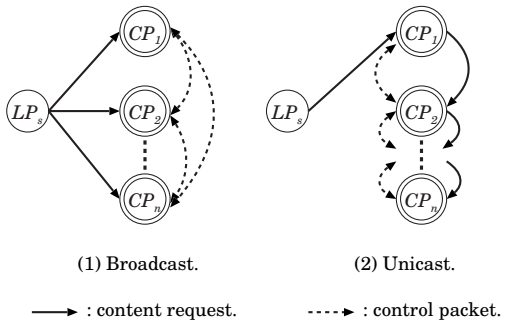

Figure 4. Coordination.

We propose a flooding-based approach similar to the gossip protocols [6, 7]. A leaf peer LP_s first sends a content request to only some number H ($\leq n$) of the contents peers as shown in Figure 5. On receipt of a content request from LP_s, a contents peer CP_i starts transmitting packets at rate τ, where τ shows the transmission rate of a multimedia content, e.g. 30 Mbps for video streaming. That is, LP_s has to receive every packet of the content at rate ($\geq \tau$). A contents peer CP_i is *active* iff CP_i is sending content packets to LP_s. Otherwise, CP_i is *dormant*. Here, let pkt_i be a subsequence pkt_{is} of packets of a content which CP_i sends to LP_s. We assume that every contents peer can transmit packets at the same rate for simplicity in this paper. First, suppose every contents peer CP_i selected by LP_s sends the same packets to LP_s, i.e. $pkt_i = pkt$. Since each of the selected contents peers sends every packet in the sequence pkt to LP_s at the content rate τ, the packets arrive at LP_s at rate $H\tau$. Let ρ_s be the maximum receipt rate of the leaf peer LP_s. If $H\tau \leq \rho_s$, LP_s receives every packet sent by the number H of contents peers. LP_s can surely receive every packet of the sequence pkt even if some contents peers are faulty and packets are lost and delayed in some channel with LP_s. Otherwise, LP_s loses packets due to the buffer overrun.

3.2. Reliable transmission

If each contents peer sends packets different from others, a leaf peer LP_s cannot receive every data of a content even if packets are lost or contents peers are faulty. On the other hand, if every contents peer sends the same sequence of packets, LP_s receives every data in presence of packet loss and faults of contents peers but LP_s overruns buffer. In order to

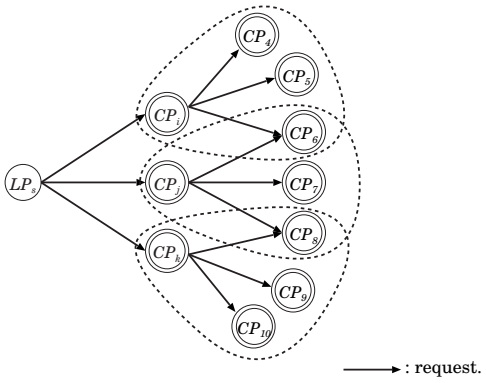

Figure 5. Flooding-based coordination.

reduce the communication overhead and increase the reliability, packets are transmitted as follows:

1. Every contents peer does not send every packet in a packet sequence pkt of a content to a leaf peer LP_s, i.e. $pkt_i \cap pkt_j = \phi$ for every pair of CP_i and CP_j.
2. Parity packets are transmitted so that data of every packet in each subsequence pkt_i can be obtained from packets of other subsequences.

For example, one parity packet $t_{\langle 1,2\rangle}$ is created for a pair of contingent packets t_1 and t_2 as shown in Figure 6. Here, even if either t_1 or t_2 is lost, data in the lost packet can be recovered from the other packet and parity packet $t_{\langle 1,2\rangle}$ [15]. Formally speaking, a packet sequence $pkt = \langle t_1, t_2, \ldots, t_l\rangle$ is separated to subsequences $s_1 = \langle t_1, \ldots, t_h\rangle$, $s_2 = \langle t_{h+1}, \ldots, t_{2h}\rangle, \ldots$ for $h \geq 1$. Each subsequence s_i is a*recovery segment* and h is *parity interval*. For the $(d + 1)$-*th* recovery segment $s_{d+1} = \langle t_{1+dh}, t_{2+dh}, \ldots, t_{(d+1)h}\rangle$ $(d \geq 0)$, one parity packet $p_d = t_{\langle 1+dh,(d+1)h\rangle}$ is created by taking the exclusive or (XOR) of the packets $t_{1+dh}, \ldots, t_{(d+1)h}$. The parity packet p_d is inserted in the recovery segment s_{d+1} for $j = d$ ***mod*** h as follows;

- $\langle p_d, t_{1+dh}, \ldots, t_{(d+1)h}\rangle$ for $j = 0$.
- $\langle \ldots, t_{dh+j}, p_d, t_{dh+j+1}, \ldots\rangle$ for $1 \leq j \leq h-2$.
- $\langle t_{1+dh}, \ldots, t_{(d+1)h}, p_d\rangle$ for $j = h-1$.

Let $[pkt]^h$ show an *enhanced* packet sequence obtained by inserting parity packets to a sequence pkt for parity interval h (≥ 1). Here, $|[pkt]^h| = |pkt|$ $(h + 1) / h$. For example, an enhanced packet sequence $[pkt]^2 = [\langle t_1, t_2, t_3, t_4, t_5, t_6, \ldots\rangle]^2 = \langle t_{\langle 1,2\rangle}, t_1, t_2, t_3, t_{\langle 3,4\rangle}, t_4, t_5, t_6, t_{\langle 5,6\rangle}, \ldots\rangle$ is created for a sequence $pkt = \langle t_1, t_2, t_3, t_4, t_5, t_6, \ldots\rangle$ and parity interval $h = 2$ as shown in Figure 6 b). Even if one packet in a recovery segment $s_{d+1} = \langle t_{1+dh}, \ldots, t_{(d+1)h}\rangle$ with a parity packet p_d is lost, data in the lost packet can be recovered from the other packets. An enhanced sequence $[pkt]^h$ is divided into H subsequences $pkt_{s_1}, \ldots, pkt_{s_H}$ $(s_u \in \{1, \ldots, n\}$ and $1 \leq u \leq H)$ as follows:

- For the jth packet t in an enhanced subsequence $[pkt]^h$, t is allocated to a subsequence pkt_{s_i} where $i = j$ ***mod*** H + 1.

For example, the enhanced sequence $[pkt]^2 = \langle t_{\langle 1,2\rangle}, t_1, t_2, t_3, t_{\langle 3,4\rangle}, t_4, \ldots\rangle$ is divided into three subsequences $[pkt]^2_1 = \langle t_{\langle 1,2\rangle}, t_3, t_5, \ldots\rangle$, $[pkt]^2_2 = \langle t_1, t_{\langle 3,4\rangle}, t_6, \ldots\rangle$, and $[pkt]^2_3 = \langle t_2, t_4, t_{\langle 5,6\rangle}, \ldots\rangle$ as shown in Figure 6 b). Since H contents peers $CP_{s_1}, \ldots, CP_{s_H}$ $(CP_{s_i} \in \{CP_1, \ldots, CP_n\})$ transmit packets to the leaf peer LP_s, each CP_{s_i} sends packets in a subsequence $[pkt]^h_{s_i}$ at rate $\tau(h + 1) / (hH)$. The leaf peer LP_s receives packets at rate $\tau(h + 1) / h$. Here, even if $(H - h)$ contents peers are faulty, LP_s can receive every data of a content from the other h operational contents peers. In addition, even if packets are lost with $(H - h)$ channels in a bursty manner, LP_s can receive every data of a content. For $h = H - 1$, each contents peer CP_{s_i} sends packets at rate $\tau / (H - 1)$ and the receipt rate of LP_s is $\tau H / (H - 1)$.

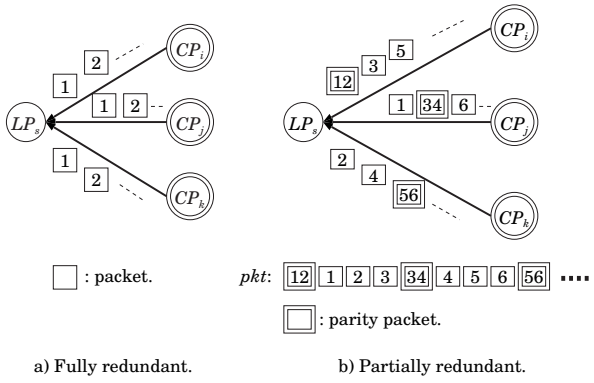

Figure 6. Packet sequence with parity packets.

3.3. Selection of contents peers

Each active contents peer CP_j randomly selects H $(\leq n)$ other contents peers out of $(n - 1)$ contents peers except for CP_j while transmitting packets to a leaf peer LP_s on receipt of a content request from another peer. CP_j and the selected contents peers send packets in a subsequence pkt_j to a leaf peer, i.e. totally H contents peers send packets. Here, a contents peer may be selected by multiple contents peers. If a contents peer selected by CP_j is taken by another contents peer, CP_j may not take H ones. Hence, CP_j may obtain only H_j $(\leq H)$ contents peers. One is a *redundant* approach where one contents peer may be selected by multiple parents as shown in Figure 5. The other is a *non-redundant* approach where each contents peer can be selected by at most one parent. We discuss how to select contents peers later.

Suppose a contents peer CP_i is selected by CP_j. Here, CP_j and CP_i are referred to as *parent* and *child*, respectively. The parent CP_j sends a control packet c to each child CP_i to make CP_i start transmitting packets to LP_s. Here, the control packet c carries the view VW_j, the sequence number SEQ_j of a packet which CP_j has most recently sent to

LP_s, the transmission rate τ_j, and the number H_j of child contents peers. On receipt of a control packet c from a parent CP_j, a child CP_i knows by what transmission schedule CP_j is transmitting packets. Based on the information on the parent CP_j, CP_i makes the transmission schedule and starts transmitting packets to LP_s according to the schedule. Each child CP_i transmits a subsequence of the packet subsequence pkt_j of the parent CP_j.

We have to discuss which packets each child contents peer CP_i starts transmitting on receipt of a control packet c from a parent contents peer CP_j. Suppose a parent CP_j is sending packets in a packet subsequence pkt_j. As discussed before, a contents peer CP_j creates an enhanced sequence $[pkt_j]^{h_j}$ from the sequence pkt_j for parity interval h_j. Each child CP_i is assigned with a subsequence pkt_{ji} obtained by dividing the enhanced subsequence $[pkt_j]^{h_j}$ to the number H_j of child contents peers. The parent CP_j informs a child CP_i that CP_j had most recently sent a packet t at the transmission rate τ_j when CP_j sent the control packet c to CP_i. On receipt of the control packet c from the parent CP_j, the child CP_i perceives that CP_j sent the packet t to the leaf peer LP_s δ time units before [Figure 7]. The parent CP_j has sent the number δ / τ_j of packets for δ time units since CP_j sent the packet t until the child CP_i receives the control packet c. The child CP_i marks the (δ / τ_j)-th packet m_j following the packet t in a subsequence pkt_j. Here, m_j is referred to as *marked* for the packet t. The child CP_i is required to send packets following the marked packet m_j. From the postfix $pkt_j\langle t]$ of the subsequence pkt_j for the packet t, the child CP_i constructs a subsequence pkt_{ji} of packets by inserting parity packets for the number H_j of the children of CP_j and parity instance h_j. The child CP_i sends packets in pkt_{ji} to LP_s. The parent CP_j also changes the packet subsequence to pkt_{ii} and the rate to τ_j / (H_j + 1) on δ time units after CP_j sends to control packet as the child contents peer. Hence, the parent CP_j and H_j children transmit packets according to the transmission schedule, i.e. totally H_j + 1 ($\leq H$) contents peers.

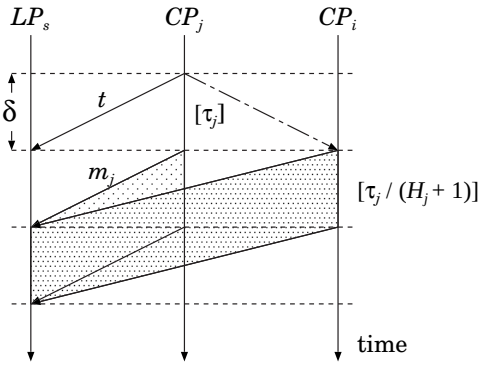

Figure 7. Transmission.

Since a parent contents peer CP_j randomly selects child contents peers, a contents peer CP_i may be selected by multiple parents, say CP_j and CP_k. One way is that CP_i takes both CP_j and CP_k as the parents. CP_i creates subsequences pkt_{ji} from pkt_j of CP_j and pkt_{ki} from pkt_k of CP_k as presented here. Then, the subsequences pkt_{ji} and pkt_{ki} are merged into a subsequence $pkt_{\langle\langle jk\rangle i\rangle} = pkt_{ji} \cup pkt_{ki}$. CP_i sends packets in the subsequence $pkt_{\langle\langle jk\rangle i\rangle}$ to the leaf peer LP_s. On CP_j's selecting CP_i as a child, the child CP_i might have been taken already as a child of another parent and been sending packets in a subsequence pkt_i to LP_s. On receipt of a content request from a parent CP_j, a pair of the subsequences pkt_i and pkt_{ji} are merged to $pkt_i = pkt_i \cup pkt_{ji}$. Here, a parent CP_j surely takes the number H of child contents peers while some of the children may have multiple parents. That is, $H_j = H$. Question is when each contents peer CP_i can stop selecting child contents peers. A control packet c sent by a parent CP_j carries the view VW_j. On receipt of the control packet c, VW_i is updated to be $VW_i \cup c.VW$ (= VW_j). Here, if $|VW_i| = n$, CP_i does not send a control packet to selected child contents peers. An enhanced subsequence pkt_{ji} (= $[pkt_j]_i^H$) is obtained by adding parity packets to pkt_j, i.e. obtaining an enhanced sequence $[pkt_j]^H$ and dividing $[pkt_j]^H$ to H subsequences, i.e. $[pkt_j]_i^H$ to CP_i.

3.4. Redundant coordination protocol

We discuss the *distributed coordination protocol* (DCoP) where a child contents peer may be selected by multiple parents. Let **CP** be a set of contents peers $CP_1, \ldots, CP_n$. We introduce the following procedures to present the coordination protocol. A function **Select**(**CP**, CP_i, m) gives a set of at most m different child contents peers for a contents peer CP_i, which are selected in a set **CP** $- \{CP_k \mid CP_k \in VW_i\}$. If $VW_i = \langle 1, \ldots, 1\rangle$, ϕ is returned. A function **Esq**(pkt, h) gives an enhanced subsequence $[pkt]^h$ obtained by inserting parity packets to a sequence pkt for parity interval h. **Div**(pkt, H, CP_i) outputs a subsequence pkt_i of a sequence pkt which is obtained by dividing pkt into H subsequences and assigning one of them to a contents peer CP_i. **Mark**(CP_i, pkt, t, δ, τ) shows a marked packet m in a sequence pkt which is to be sent by CP_i on δ time units after CP_i sent a packet t in pkt where τ is the transmission rate of CP_i. **Psend**(CP_i, pkt, τ, LP_s) means that CP_i sends packets in a sequence pkt to a leaf peer LP_s at rate τ. **Csend**(CP_i, c, CP_j) shows that a contents peer CP_i sends a control packet c to CP_j. **Current**(CP_i) shows a packet which CP_i has most recently sent. We show the protocol DCoP for the number H, parity interval h, leaf peer LP_s, and content rate τ as follows:

[DCoP(CP, LP_s, H, h, n, τ**)]**

1. First, a leaf peer LP_s selects H ($\leq n$) contents peers in **CP** and sends a content request c of a multimedia content C to the selected contents peers;
   C := **Select**(**CP**, ϕ, H);
   $c.\tau := \tau$;
   Csend(LP_s, c, CP_k);

2. On receipt of a content request c_1 from LP_s, a contents peer CP_i does the following actions:
   - creates an enhanced sequence $[pkt]^h$ from a packet sequence pkt and then obtains a subsequence pkt_i from $[pkt]^h$;
     pkt_i := **Div**(**Esq**(pkt, h), H, CP_i);
   - starts transmitting packets in pkt_i to LP_s at rate τ_i;
     $\tau_i := c_1.\tau(h + 1) / (hH)$;
     Psend(CP_i, pkt, τ_i, LP_s);
   - selects $(H - 1)$ contents peers from **CP**;
     C := **Select**(**CP**, CP_i, H);
   - sends a control packet c to the selected contents peers;
     For every CP_k, $VW_{ik} := 1$ if $CP_k \in$ **C**, otherwise $VW_{ik} := 0$;
     $c_1.VW := VW_i$; $c_1.\tau := \tau_i$;
     t := **Current**(CP_i); $c_1.SEQ := t.SEQ$;
     Csend(CP_i, c, CP_k) for every $CP_k \in$ **C**;
   - After it takes δ time units, CP_i does the actions of step 3.
3. On receipt of a control packet c_1 from a parent CP_j, a contents peer CP_i does the following actions:
   - $VW_i := VW_i \cup c.VW$;
   - creates an enhanced subsequence $epkt_{ji} = [pkt_j[m_j\rangle]^h$ from a postfix $pkt_j[m_j\rangle$ of the subsequence pkt_j of CP_j where m_j is a marked packet for a packet t, where $t.SEQ = c_1.SEQ$;
     m_j := **Mark**(CP_j, pkt_j, t, δ, τ_j);
     $epkt_{ji}$:= **Esq**($pkt_j[m_j\rangle$, h);
   - transmits packets in an enhanced subsequence pkt_{ji} from $epkt_{ji}$ to LP_s;
     pkt_{ji} := **Div**($epkt_j$, $H + 1$, CP_i);
     $\tau_i := c_1.\tau(h + 1) / (h(H + 1))$;
     Psend(CP_i, pkt_{ji}, τ_i, LP_s);
   - if $|VW_i| < n$, selects H contents peers and sends a control packet c to the contents peers;
     C := **Select**(**CP**, CP_j, H);
     if **C** = ϕ, CP_i stops selecting child peers.
     $VW_{ik} := 1$ if $CP_k \in$ **C**; $c_1.VW := VW_i$;
     $c_1.\tau := \tau_i$;
     t := **Current**(CP_i); $c_1.SEQ := t.SEQ$;
     Csend(CP_i, c, CP_k) for every $CP_k \in$ **C**;

3.5. Non-redundant coordination protocol

In another non-redundant way named *tree-based coordination protocol* (TCoP), each contents peer CP_i takes either one of CP_j and CP_k as the parent if CP_j and CP_k select CP_i as a child. For example, CP_i takes CP_j since CP_i receives a control packet from CP_j before CP_k. Hence, a parent CP_j has to know which contents peer selected can be a child of CP_j. **Aselect**(**CP**, CP_i, H) selects H different contents peers in **CP** $- \{CP_i\} - \{CP_k \mid VW_{ik} =$ ON$\}$, i.e. selects contents peers in **CP** excluding the parent CP_i and contents peers which CP_i knows to have been selected. Here, |**Aselect**(**CP**, CP_i, H)| $\leq H$. **Aselect**(**C**, CP_i) collects a contents peer which sends the positive acknowledgment in **C**. The following procedure [Figure 8] is taken:

[TCoP(CP, LP_s, H, n, τ**)]**

1. First, a leaf peer LP_s selects H ($\leq n$) contents peers and sends a content request c of a multimedia content C to the selected contents peers as DCoP where $c.\tau = \tau$.
2. CP_j randomly selects H contents peers and sends a control packet c_1 to each of the selected contents peers while sending packets in a subsequence pkt_j to LP_s;
   pkt_i := **Div**(**Esq**(pkt, t), $H + 1$, CP_i);
   $\tau_i := c_1.\tau(h + 1) / (h(H + 1))$;
   Psend(CP_j, pkt, τ_i, LP_s);
   C := **Aselect**(**CP**, CP_j, H);
   t := **Current**(CP_i); $c_1.SEQ := t.SEQ$;
   $c_1.VW_{ii} := 1$, $c_1.VW_{ik} := 1$ if $CP_k \in$ **C**;
   $c_1.\tau := \tau_i$;
   Csend(CP_j, c_1, CP_k) for every CP_k in **C**;
3. On receipt of the control packet c_1 from CP_j, CP_i sends a confirmation cc_1 to CP_j if CP_i takes CP_j as the parent.
   Csend(CP_i, cc_1, CP_k);
4. CP_j collects the confirmations from the selected contents peers. Then, CP_j sends a control packet c_2 to each of the confirmed contents peers.
   $\mathbf{H}_j$:= **Areceive**(**C**, CP_j);
   $c_2.VW := VW_j$;
   t := **Current**(CP_j); $c_2.SEQ := t.SEQ$;
   $c_2.\tau := \tau_j$;
   $c_2.n := |\mathbf{H}_j|$;
5. On receipt of c_2 from CP_j, CP_i decomposes the subsequence $pkt_j[t\rangle$ to a subsequence pkt_{ji};
   $t := (\delta / \tau_j)$-th packet from $c_2.SEQ$ in pkt_j;
   m_j := **Mark**(CP_i, pkt_j, t, δ, τ_j) for a packet t such that $t.SEQ = c_2.SEQ$;
   pkt_{ji} := **Esq**($pkt_j[m_j\rangle$, $c_2.n$);
   $\tau_i := \tau_j / c_2.n$;
   Psend(CP_j, pkt_{ji}, τ_i, LP_s);
6. CP_j also makes a subsequence pkt_{jj} as presented in CP_i. On δ time units after sending the control packet c_2, CP_j sends packets in pkt_{jj} at rate $\tau_j / c_2.n$.

If a contents peer CP_j could find no child, CP_j stops selecting child contents peers. Here, a set of contents peers are structured in a tree whose root is a leaf peer LP_s. A tree of Figure 9 is obtained from Figure 5. Compared with DCoP, we can remove the redundancy but it takes three rounds for each selection of child contents peers.

3.6. Examples

First, a leaf peer LP_s sends a control packet to three contents peers randomly selected, say CP_1, CP_2, and CP_3 of a multimedia content C. Let pkt be a packet sequence $\langle t_1, t_2, \ldots\rangle$. Each CP_i of the contents peers sends an enhanced packet subsequence $[pkt]_i^h$ as shown in Figure 6. Here, $[pkt]_1^2 = \langle t_{\langle 1,2\rangle}, t_3, t_5, t_{\langle 7,8\rangle}, t_9, \ldots\rangle$, $[pkt]_2^2 = \langle t_1, t_{\langle 3,4\rangle}, t_6, t_7,$

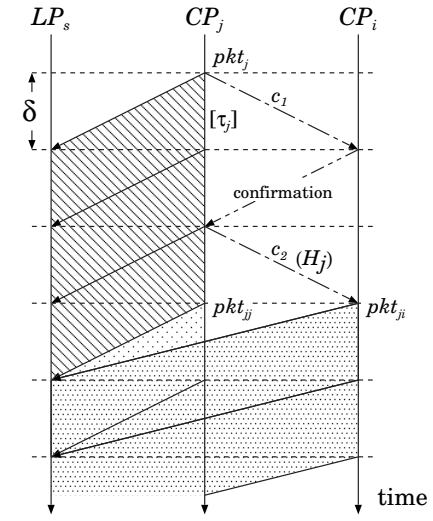

Figure 8. Transmission.

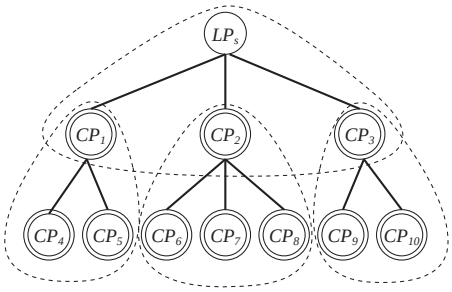

Figure 9. Transmission tree.

$t_{\langle 9,10\rangle}, \ldots\rangle$, and $[pkt]_3^2 = \langle t_2, t_4, t_{\langle 5,6\rangle}, t_8, t_{10}, \ldots\rangle$ for parity interval $h = 2$. Then, each CP_i randomly selects three contents peers, say CP_1 selects three contents peers CP_4, CP_5, and CP_6, CP_2 selects CP_6, CP_7, and CP_8, and CP_3 selects CP_8, CP_9, and CP_{10} for $H = 3$. Suppose that each CP_i sends two packets for δ time units. In DCoP, CP_6 is a child of two parents CP_1 and CP_2. A pair of enhanced subsequences $[[pkt]_1^2]^3 = \langle t_{\langle\langle 1,2\rangle,3,5\rangle}, t_{\langle 1,2\rangle}, t_3, t_5, t_{\langle 7,8\rangle}, t_{\langle\langle 7,8\rangle,9,11\rangle}, t_9, t_{11}, \ldots\rangle$ and $[[pkt]_2^2]^3 = \langle t_{\langle 1,\langle 3,4\rangle,6\rangle}, t_1, t_{\langle 3,4\rangle}, t_6, t_7, t_{\langle 7,\langle 9,11\rangle,12\rangle}, t_{12}, \ldots\rangle$ are obtained for the parents CP_1 and CP_2, respectively, each of which is divided to four subsequences. CP_1 takes an enhanced subsequence $[[pkt]_1^2]_1^3 = \langle t_{\langle\langle 1,2\rangle,3,5\rangle}, t_{\langle 7,8\rangle}, \ldots\rangle$. CP_6 takes a pair of enhanced subsequences $[[pkt]_1^2]_6^3 = \langle t_5, t_{11}, \ldots\rangle$ from CP_1 and $[[pkt]_2^2]_6^3 = \langle t_1, t_{\langle 7,\langle 9,11\rangle,12\rangle}, \ldots\rangle$ from CP_2 and merges them to $pkt_6 = \langle t_1, t_5, t_{11}, t_{\langle 7,\langle 9,11\rangle,12\rangle}, \ldots\rangle$. Then, CP_6 sends packets in pkt_6.

In TCoP, contents peers CP_6 and CP_8 are selected by a pair of parents CP_1 and CP_2 and a pair of CP_2 and CP_3, respectively. Suppose CP_6 and CP_8 take CP_2 as the parent. CP_4 and CP_5 start transmitting packets following the packet t_3. The subsequence $[pkt]_1^2[t_5\rangle = \langle t_5, t_{\langle 7,8\rangle}, t_9, t_{11}, t_{\langle 13,14\rangle}, \ldots\rangle$ is enhanced by adding parity packets for parity interval $h = 2$. Here, a subsequence $\langle t_{\langle 5,\langle 7,8\rangle\rangle}, t_5, t_{\langle 7,8\rangle}, t_9, t_{\langle 9,11\rangle}, t_{11}, t_{\langle 5,\langle 13,14\rangle\rangle}, t_{15}, t_{\langle\langle 13,14\rangle,15\rangle}, \ldots\rangle$ is obtained. Here, CP_1, CP_4, and CP_5 take subsequences $\langle t_{\langle 5,\langle 7,8\rangle\rangle}, t_9\ t_{\langle 13,14\rangle}, \ldots\rangle$, $\langle t_5, t_{\langle 9,11\rangle}, t_{15}, \ldots\rangle$, and $\langle t_{\langle 7,8\rangle}, t_{11}, t_{\langle\langle 7,8\rangle,15\rangle}\rangle$, respectively.

4. Evaluation

We evaluate a pair of the coordination protocols DCoP and TCoP for synchronizing multiple contents peers in terms of the synchronization time and the number of redundant parity packets. Suppose there are n contents peers $CP_1, \ldots, CP_n$ which transmit packets of a content to a leaf peer LP_s. Let H be the number of child contents peers to be selected by each parent ($H \leq n$). $(H - h)$ shows packet interval. Suppose each channel CC_i between CP_i and LP_s supports reliable high-speed communication like 10 Gbps Ethernet.

Figure 10 shows the number of control packets transmitted and how many rounds it takes to synchronize 100 contents peers in DCoP for each H ($2 \leq H \leq 100$). Here, h = 1, i.e. one parity packet is sent for every 99 packets. The straight line shows the number of rounds and the dotted line indicates the number of control packets. For example, it takes two rounds and about 600 control packets are transmitted until all the contents peers start transmitting packets to a leaf peer in two rounds for H = 60. Figure 11 shows the number of control packets and the number of rounds in TCoP. About 7400 control packets are transmitted in six rounds for H = 60. More number of packets are transmitted in TCoP than DCoP.

In DCoP and TCoP, one parity packet is transmitted for every $H - h$ packets. Figure 12 shows the receipt rate of a leaf peer from 100 contents peers for each H. Here, "rate = 1" shows the content rate, for example, 30 Mbps for video content. If no parity packet is transmitted in DCoP and TCoP, the leaf peer receives the content rate, i.e. rate = 1. For example, rate = 1.019 in DCoP and rate = 1.226 in TCoP for H = 60. In DCoP, the fewer number of parity packets are transmitted than TCoP. The smaller H is, the more number of parity packets are transmitted.

5. Concluding Remarks

In this paper, we discussed the *multi-source streaming* model for transmitting continuous multimedia contents from multiple contents peers to a leaf peer. In P2P overlay networks, peers on various types of computers can support other peers with multimedia contents. We discussed two types of the distributed coordination protocols, DCoP and TCoP for multiple contents peers to transmit packets to a leaf peer. In order to reduce the communication overheads, only a subset of the contents peers start transmitting packets and then each of the contents peers initiates some number of other contents peers. In the evaluation, DCoP shows better performance than TCoP. We are now discussing heterogeneous environ-

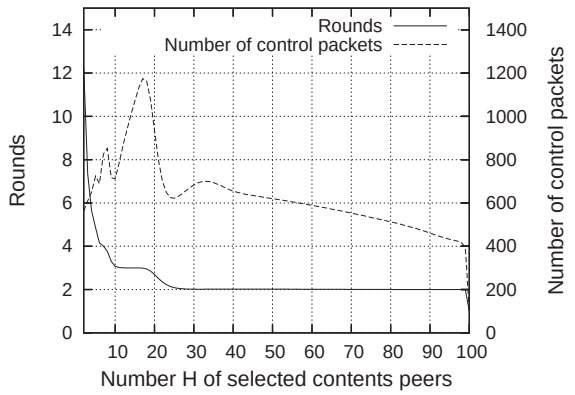

Figure 10. Rounds and number of control packets in DCoP.

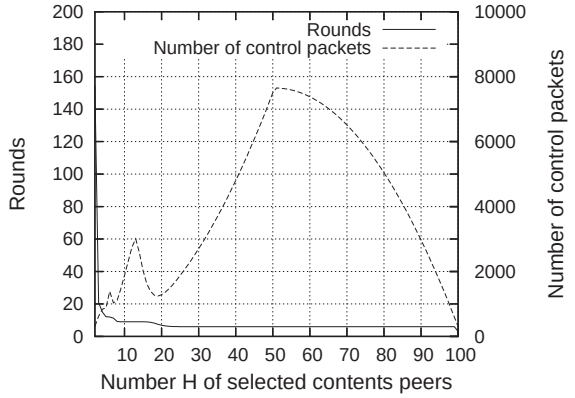

Figure 11. Rounds and number of control packets in TCoP.

ment where each contents peer may support different transmission rate and even change the rate.

Acknowledgment

This research is partially supported by Research Institute for Science and Technology [Q05J-04] and Frontier Research and Development Center [18-J-6], Tokyo Denki University.

References

[1] Apple Computer, Inc. *iTunes*. http://www.apple.com/itunes/.

[2] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong. Freenet: A Distributed Anonymous Information Storage and Retrieval System. In *Proc. of the Workshop on Design Issues in Anonymity and Unobservability*, pages 311–320, 2000.

[3] 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 the 19th International Conference on Advanced Information Networking and Applications (AINA-2005)*, volume 1, pages 27–32, 2005.

[4] S. Itaya, T. Enokido, M. Takizawa, and A. Yamada. A Scalable Multimedia Streaming Model Based on Multi-source Streaming Concept. In *Proc. of the IEEE 11th International Conference on Parallel and Distributed Systems (ICPADS-2005)*, volume 1, pages 15–21, 2005.

[5] S. Itaya, N. Hayashibara, T. Enokido, and M. Takizawa. Scalable Peer-to-Peer Multimedia Streaming Model in Heterogeneous Networks. In *Proc. of the 7th IEEE International Symposium on Multimedia (ISM'05)*, pages 208–215, 2005.

[6] A.-M. Kermarrec, L. Massoulié, and A. J. Ganesh. Probabilistic Reliable Dissemination in Large-Scale Systems. *IEEE Trans. on Parallel and Distributed Systems*, 14(3):248–258, 2003.

[7] M.-J. Lin and K. Marzullo. Directional Gossip: Gossip in a Wide Area Network. *Technical Report: CS1999-0622*, 1999.

[8] X. Liu and S. T. Vuong. Supporting Low-Cost Video-on-Demand in Heterogeneous Peer-to-Peer Networks. In *Proc. of the 7th IEEE International Symposium on Multimedia (ISM'05)*, pages 523–530, 2005.

[9] Micorosoft. *Windows Media Technology*. http://www.microsoft.com/windows/windowsmedia/.

[10] A. Nakamura and M. Takizawa. Causally Ordering Broadcast Protocol. In *Proc. of IEEE the 14th International Conference on Distributed Computing Systems (ICDCS-14)*, pages 48–55, 1994.

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

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

[13] RealNetworks. *Real.com*. http://www.realnetworks.com/.

[14] D. Skeen. Nonblocking Commitment Protocols. In *Proc. of ACM SIGMOD*, pages 133–147, 1981.

[15] 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.

[16] D. Xu, M. Hefeeda, S. Hambrusch, and B. Bhargava. On Peer-to-Peer Media Streaming. In *Proc. of IEEE the 22nd International Conference on Distributed Computing Systems (ICDCS-22)*, pages 363–371, 2002.

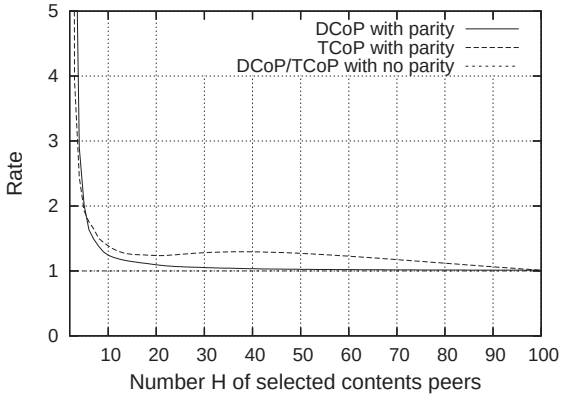

Figure 12. Receipt rate of leaf peer.