Journal of Mobile Multimedia, Vol. 2, No.4 (2006) 283-296

DISTRIBUTED COORDINATION PROTOCOLS TO REALIZE SCALABLE MULTIMEDIA STREAMING IN PEER-TO-PEER OVERLAY NETWORKS

TOMOYA ENOKIDO
Faculty of Business Administration, Rissho University
eno@ris.ac.jp

YOUHEI TANAKA   SATOSHI ITAYA   MAKOTO TAKIZAWA
Department of Computers and Systems Engineering, Tokyo Denki University
{*youhe, itaya,taki*}*@takilab.k.dendai.ac.jp*

Received May 31, 2006
Revised Sep. 20, 2006

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 treebased 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 coordination protocols DCoP and TCoP in terms of how long it takes and how many messages are transmitted to synchronize multiple contents peers.

Key words: Multimedia Streaming, Multi-source Streaming, Quality of Service, Peer-to-Peer (P2P) Network, Flooding-based Coordination

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. For example, a peer downloads a free movie from some computer and then another peer can obtain the movie content from the peer. In addition, parts of movie contents are cached in peers. Peers supporting other peers with multimedia contents are referred to as contents peers in this paper. Peers which receive multimedia contents are referred to as leaf peers. The contents-leaf relation is relative, i.e. each peer can be not only a contents but also 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 multimedia 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 peer [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 the group communication. In this paper, we take a gossip-based flooding protocol [6, 7] to reduce the communication overhead. First, a leaf peer sends a content request to only some number of contents peers, not all the contents peers. Then, a selected contents peer starts transmitting packets to the leaf peer. The contents peer in turn selects some number of child contents peers and then sends a content request to the selected contents peers. Since each of multiple contents peers selects child contents peers independently of the other peers, a pair of peers may select the same contents peer. There are two algorithms; a child contents peer may be selected by multiple contents peers in one algorithm and is selected by at most one contents peer in the other. The former is a *distributed coordination protocol* (DCoP) and the latter is a *tree-based coordination protocol* (TCoP) in another algorithm. A content request carries information on which packets the parent contents peer has sent to a leaf peer at what rate. Each of the selected child contents 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 multimedia content to contents peers in heterogeneous environment where each contents peer supports different transmission rate. In section 3, we discuss the coordination protocols, DCoP and TCoP. In section 4, we evaluate the coordination protocols DCoP and TCoP.

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 content from a contents peer is a *leaf* peer. Each peer can play any role of contents peer and leaf peer. 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* (SSS) 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 (MSS) 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 multimedia 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 supported by the underlying network. A packet is a unit of data transmission in the underlying network. A content is first decomposed into a sequence of packets. Multiple contents peers $CP_1, \ldots, CP_n$ in parallel transmit packets of a multimedia content to each leaf peer LP_s in the MSS model.

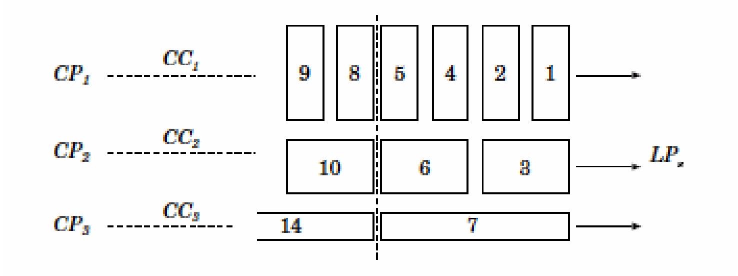


Figure 1 Multi-source streaming(MSS).

Each contents peer CPi ($i = 1, \ldots, n$) 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 LPs. Here, $|pkt| = l$. Suppose three contents peers CP_1, CP_2, and CP_3 transmit packets in a packet sequence $pkt = \langle t_1, \ldots, t_7, \ldots \rangle$ C to a leaf peer LPs where $bw_1 : bw_2 : bw_3 = 4 : 2 : 1$. Each contents peer CPi transmits a subsequence pkt_i of the packet sequence pkt to the leaf peer LP_s. $|pkt_i| \geq |pkt_j|$ if $bw_i \geq bw_j$. For example, the fastest contents peer CP_1 transmits packets t_1, t_2, t_4, and t_5 the second fasted CP_2 transmits t_3 and t_6, and the slowest 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, $pkt_1 \cup pkt_2 \cup pkt_3 = \langle t_1, t_2, t_3, t_4, t_5, t_6, t_7, \ldots \rangle$. 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 communication channel CC_i is modelled 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 communication channels CC_1, CC_2, and CC_3. $st(CL_i^k)$ and $et(CL_i^k)$ show when a contents peer CP_i starts and finishes transmitting the kth packet t_i^k in the packet sequence pkt_i, respectively. $st(CL_i^0)$ is 0 and $et(CL_i^k) = st(CL_i^k) + \tau_i = st(CL_i^{k+i})$ 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, . . . , CC_n. Time slots in **CL** are partially ordered in $\rightarrow$. A time slot CL is *initial* iff (if and only if) there is no time slot *CL'* such that *CL'* $\rightarrow$ *CL* in **CL**. For example, $CL_1^1 \rightarrow CL_2^1$ and $CL_1^2 \rightarrow CL_3^1$ in Figure 2.

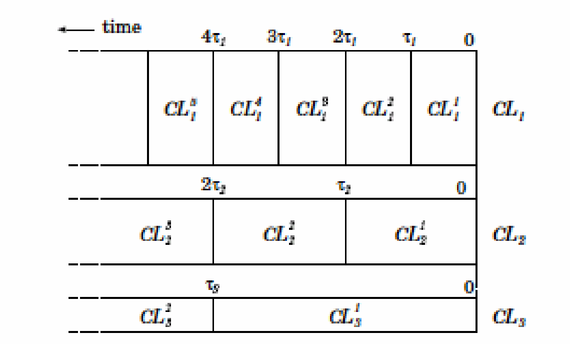


Fig. 2. Time slots.

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, . . . , l),

1. Find an initial time slot *CL* such that *st*(*CL*) ≥ *st*(*CL'*) 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 the precedent relations of time slots shown in Figure 2, 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 . . . , t_{h-1} since the packets t_1, . . . , t_{h-1} preceding packet t_h are surely delivered on receipt of the packet t_h. The allocation algorithm satisfies the following property.

[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* = $\langle t_1 . . . , t_l \rangle$.

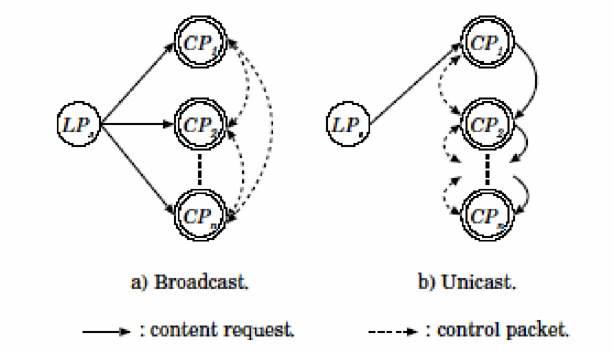
3. Distributed Coordination Protocols

3.1. Types of distributed coordination protocols

Multiple contents peers CP_1, . . . , CP_n are required to cooperate to reliably deliver packets of a multimedia 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 contents peer CP_i supports the same transmission rate to LP_s for simplicity. Let *pkt* be a sequence of packets t_1, . . . , t_l of a content *C*. While transmitting packets to LP_s, 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, . . . , CP_n [Figure 3 a)]. On receipt of the request, every contents peer CP_i starts transmitting packets in the packet sequence *pkt* of the multimedia content *C* to the leaf peer LP_s. Here, the leaf peer 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 the contents peer 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 transmission rate. It takes one round for every contents peer to start transmitting packets to a leaf peer LP_s. However, LP_s may lose packets due to the buffer overrun. In addition, each contents peer 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 of a multimedia content C to only one contents peer, say CP_1 as shown in Figure 3 b). Then, the contents peer CP_1 starts transmitting packets to the leaf peer LP_s. The contents peer CP_1 sends a control packet to another contents peer, say CP_2 to inform what packet CP_1 has sent while transmitting protocols to the leaf peer LP_s. On receipt of the control packet, the contents peer CP_2 starts transmitting packets to LP_s and sends a control packet to CP_3. Finally, a contents peer CP_n starts transmitting packets to the leaf peer LP_s. Here, every contents peer is transmitting packets of the multimedia content C. This implies the minimum redundancy but it takes the longest time all the contents peers to synchronize.



a) Broadcast. b) Unicast.

Fig. 3. 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 4. On receipt of a content request from the leaf peer 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, the leaf peer LP_s has to receive every packet of the content at rate ($\leq \tau$). A contents peer CP_i is *active* iff CP_i is sending content packets to the leaf peer LP_s. Otherwise, CP_i is *dormant*. Here, let pkt_i be a subsequence pkt_{is} of packets of a content which a contents peer CP_i sends to the leaf peer 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 the leaf peer LP_s sends the same packets to the leaf peer 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 the leaf peer LP_s at rate $H\tau$. Let ρ_s be the maximum receipt rate of the leaf peer LP_s. If $H\tau \leq \rho_s$, the leaf peer LP_s receives every packet sent by the number H of contents peers. The leaf peer 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 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, the leaf peer LP_s receives every data in presence of packet loss and faults of contents peers but LP_s overruns buffer. In order to reduce the communication overhead and increase the reliability, packets are transmitted as follows:

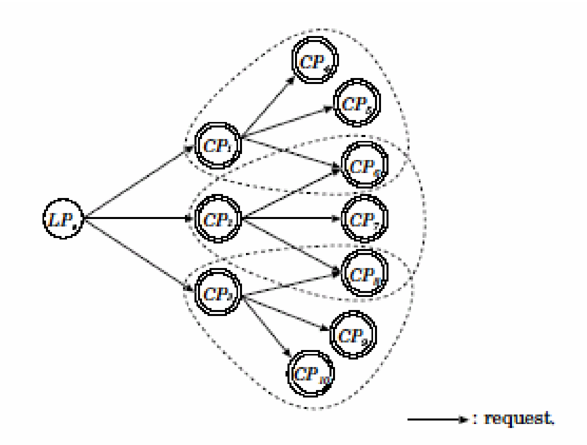


Fig. 4. Flooding-based coordination.

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 contents peer CP_i and CP_j.

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 $t_{<1, 2>}$ is created for a pair of contingent packets t_1 and t_2 as shown in Figure 5. Here, even if either t_1 or t_2 is lost, data in the lost packet can be recovered from the other packet and the parity packet $t_{<1, 2>}$ [15]. Formally speaking, a packet sequence $pkt = <t_1, t_2, \ldots, t_l>$ is separated to subsequences $s_1 = <t_1, \ldots, t_h>$, $s_2 = <t_{h+1}, \ldots, t_{2h}>, \ldots$ for $h \geq 1$. Each subsequence s_i is a *recovery segment* and *h* is *parity interval*. For the $(d + 1)$-*th* recovery segment $s_{d+1} = <t_{1+dh}, t_{2+dh}, \ldots, t_{(d+1)h}>$ $(d \geq 0)$, one parity packet $p_d = t_{<1+dh, (d+1)h>}$ is created by taking the exclusive or (XOR) of data in 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*** $(h + 1)$ as follows;

$<p_d, t_{1+dh}, \ldots, t_{(d+1)h}>$ for $j = 0$.

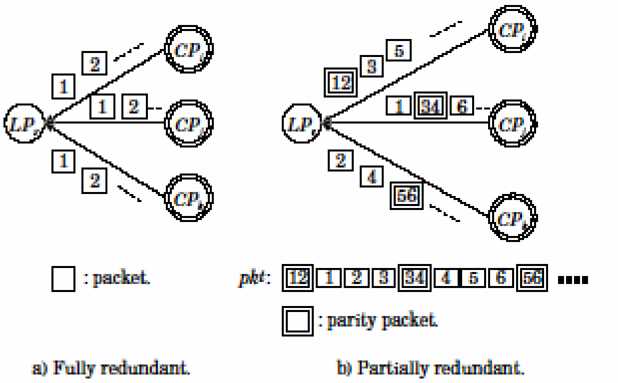
$<\ldots, t_{dh+j}, p_d, t_{dh+j+1}, \ldots>$ for $1 \leq j \leq h - 1$.

$<t_{1+dh}, \ldots, t_{(d+1)h}, p_d>$ for $j = h$.

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 $[pkt]^2 = [<t_1, t_2, t_3, t_4, t_5, t_6, \ldots>]^2 = <t_{<1, 2>}, t_1, t_2, t_3, t_{<3, 4>}, t_4, t_5, t_6, t_{<5, 6>}, \ldots>$ is created for a sequence $pkt = <t_1, t_2, t_3, t_4, t_5, t_6, \ldots>$ and parity interval $h = 2$ as shown in Figure 5 b). Even if one packet in a recovery segment $s_{d+1} = <t_{1+dh}, \ldots, t_{(d+1)h}>$ 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 $pkts_1, \dots, pkts_H$ ($s_u \in \{1, \dots, n\}$ and $1 \le u \le H$) as follows:

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

For example, the enhanced sequence $[pkt]^2 = <t_{<1,2>}, t_1, t_2, t_3, t_{<3,4>}, t_4, \dots>$ is divided into three subsequences $[pkt]_1^2 = <t_{<1,2>}, t_3, t_5, \dots>$, $[pkt]_2^2 = <t_1, t_{<3,4>}, t_6, \dots>$, and $[pkt]_3^2 = <t_2, t_4, t_{<5,6>}, \dots>$ as shown in Figure 5 b). Since the number H of contents peers $CP_{s1}, \dots, CP_{sH}$ ($CP_{si} \in \{CP_1, \dots, CP_n\}$) transmit packets to the leaf peer LP_s, each CP_{si} sends packets in a subsequence $[pkt]_{s_i}^h$ 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 burst manner, LP_s can receive every data of a multimedia content. For $h = H - 1$, each CP_{si} sends packets at rate $\tau / (H - 1)$ and the receipt rate of the leaf peer LP_s is $\tau H / (H - 1)$.



a) Fully redundant. b) Partially redundant.

Fig. 5. Packet sequence with parity packets.

3.3. Selection of contents peers

Each active contents peer CP_j randomly selects H ($\le n$) other contents peers out of $(n - 1)$ contents peers except for the contents peer CP_j while transmitting packets to a leaf peer LP_s on receipt of a content request from another peer. The contents 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 the contents peer CP_j is taken by another contents peer, CP_j may not take H contents peers since some of the H contents peers may be taken by another contents peer. Hence, the contents peer CP_j may obtain only H_j ($\le H$) contents peers. One is a *redundant* approach where one child contents peer may be selected by multiple parents as shown in Figure 4. The other is a *non-redundant* approach where each child 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* contents peers, respectively. The parent CP_j sends a control packet c to each child contents peer CP_i to make CP_i start transmitting packets to a leaf peer LP_s. Here, the control packet c carries the view VW_j, the sequence number SEQ_j of a packet which the parent 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 contents peer CP_j, a child contents peer CP_i knows by what transmission schedule the parent CP_j is transmitting packets. Based on the information on the parent CP_j, the child CP_i makes the transmission schedule and starts transmitting packets to the leaf peer LP_s according to the transmission schedule. Each child contents peer 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, 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 contents peer CP_j informs a child CP_i that the CP_j had most recently sent a packet t at the transmission rate τ_j when CP_j sent the control packet c to the child 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 6]. The parent contents peer 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 contents peer 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 $pkt_j\,[t>$ 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 child contents peers of the parent 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 $\tau_j / (H_j + 1)$ on δ time units after CP_j sends to control packet as the child contents peer. Hence, the parent CP_j and H_j child contents peers transmit packets according to the transmission schedule, i.e. totally $H_j + 1$ ($\leq H$) contents peers.

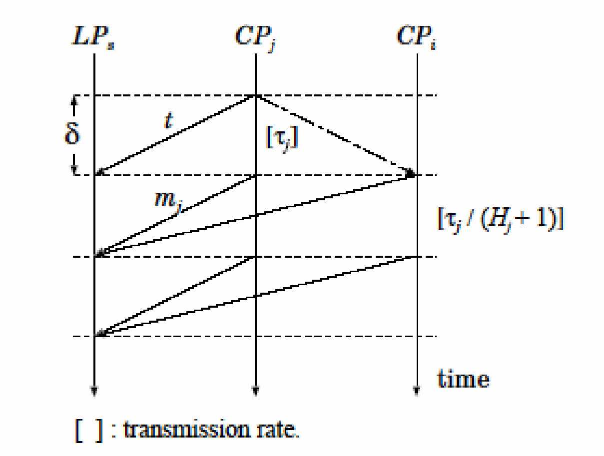


Fig. 6. Transmission.

Since a parent contents peer CP_j randomly selects child contents peers, a child peer CP_i may be selected by multiple parents, say CP_j and CP_k as a child. One way is that the parent contents peer CP_i takes both the contents peers CP_j and CP_k as the parents. The parent peer CP_i creates subsequences pkt_{ji} from pkt_j of the child contents peer CP_j and pkt_{ki} from pkt_k of the child CP_k as presented here. Then, the subsequences pkt_{ji} and pkt_{ki} are merged into a subsequence $pkt_{<<jk>i>} = pkt_{ji} \cup pkt_{ki}$. The parent contents peer CP_i sends packets in the subsequence $pkt_{<<jk>i>}$ 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 contents peer 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 $|VW_i| = n$, the parent 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 parent 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. **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 which is obtained by dividing a sequence 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 pkt which is to be sent by a content peer CP_i on δ time units after the contents peer 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 a content peer 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 the leaf peer 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 the leaf peer LP_s at transmission 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 the set **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 2 **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_1, 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 contents peer 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 where m_j is a marked packet for a packet t, where $t.SEQ = c_1.SEQ$;
     $m_j :=$ **Mark**$(CP_j, pkt_j, t, \delta, \tau_j)$;
     $epkt_{ji} :=$ **Esq**$(pkt_j\,[m_j\,\mathrm{i}, \mathrm{h})$;
   - transmits packets in an enhanced subsequence pkt_{ji} from $epkt_{ji}$ to the leaf peer LP_s;
     $pkt_{ji} :=$ **Div**$(epkt_{\mathrm{j}}, H + 1, CP_i)$; $\tau_{\mathrm{i}} := c_1.\tau\,(h + 1) / (h(H + 1))$;
     Psend$(CP_i, pkt_{\mathrm{ji}}, \tau_i, LP_s)$;
   - if $|VW_i| < n$, selects H contents peers and then sends a control packet c to the selected contents peers;
     C := **Select**(**CP**, CP_j, H);
     if **C** $= \phi$, CP_i stops selecting child peers.
     $VW_{\mathrm{ik}} := 1$ if $CP_k \in$ **C**; $c_1.VW := VW_i$;
     $t :=$ **Current**(CP_{i}); $c_1.SEQ := t.SEQ$; $c_1.\tau := \tau_i$;
     Csend(CP_i, c_1, CP_k) for every $CP_k \in$ **C**;

3.5. Non-Redundant coordination protocol

In another *tree-based coordination protocol* (TCoP), each contents peer CP_i takes either one of contents peers CP_j and CP_k as the parent if both the contents peers CP_j and CP_k select the contents peer CP_i as a child. For example, a contents peer CP_i takes a child contents peer CP_j since the contents peer CP_i receives a control packet from the parent CP_j before another parent CP_k. Hence, the parent contents peer CP_j has to know which child contents peer selected can be a child of the parent 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 TCoP procedure [Figure 7] 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. Each selected peer CP_j randomly selects H contents peers and sends a control packet c_1 to the selected contents peers while sending packets in a subsequence pkt_j to the leaf peer LP_s;
   $pkt_i :=$ **Div**(**Esq**$(pkt, t), H + 1, CP_i)$; $\tau_{\mathrm{i}} := c_1.\,\tau\,(h + 1) / (h(H + 1))$;
   Psend$(CP_j, pkt, \tau_{\mathrm{i}}, LP_s)$;
   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**; $c_1.\tau := \tau_{\mathrm{i}}$;
   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, a contents peer 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. The parent 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); $c_2.SEQ := t.SEQ$;
$c_2.\tau := \tau_j$; $c_2.n := |\boldsymbol{H}_j|$;

5. On receipt of c_2 from the parent CP_j , a contents peer 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**($pkt_j\,(m_{ji}, c_2.n\rangle$; $\tau_i := \tau_j / c_2.n$;
   Psend(CP_j , pkt_{ji}, τ_i, LP_s);
6. The parent CP_j also makes a subsequence pkt_{ij} as presented in CP_i. On δ time units after sending the control packet c_2, CP_j sends packets in pkt_{ij} at rate $\tau_i / c_2.n$.

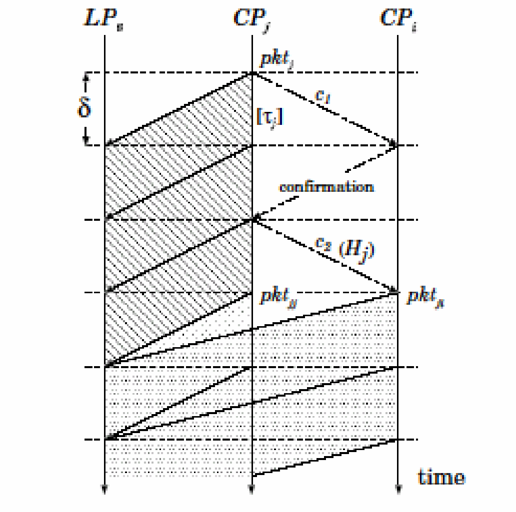


Fig. 7. Transmission.

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 8 is obtained from Figure 4. Compared with DCoP, we can remove the redundancy but it takes three rounds for each selection of child contents peers.

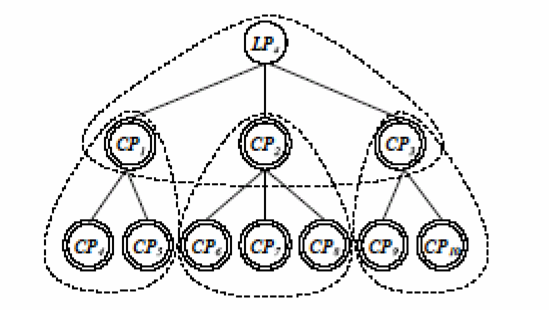


Fig. 8. Transmission tree in TCoP.

3.6. Examples

First, a leaf peer LP_s sends a control packet to three child contents peers randomly selected, say CP_1, CP_2, and CP_3 of a multimedia content C. Let pkt be a packet sequence $<t_1, t_2, \ldots>$. Each contents peer CP_i (i = 1, 2, 3) of the contents peers sends an enhanced packet subsequence $[pkt]_i^h$ as shown in Figure 5. Here, $[\mathrm{pkt}]_1^2 = <t_{<1,2>}, t_3, t_5, t_{<7,8>}, t_9, \ldots>$, $[pkt]_2^2 = <t_1, t_{<3,4>}, t_6, t_7, t_{<9,10>}, \ldots>$, and $[pkt]_3^2 = <t_2, t_4, t_{<5,6>}, t_8, t_{10}, \ldots>$ for parity interval $h = 2$. Then, each contents peer 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 contents peer CP_i sends two packets for δ time units. In the DCoP, CP_6 is a child of two parents CP_1 and CP_2. A pair of enhanced subsequences $[[pkt]_1^2]^3 = <t_{<<1,2>,3,5>}, t_{<1,2>}, t_3, t_5, t_{<7,8>}, t_{<<7,8>,9,11>}, t_9, t_{11}, \ldots>$ and $[[pkt]_2^2]^3 = <t_{<1,<3,4>,6>}, t_1, t_{<3,4>}, t_6, t_7, t_{<7,<9,11>,12>}, t_{12}, \ldots>$ are obtained for CP_1 and CP_2, respectively, each of which is divided to four subsequences. CP_1 takes an enhanced subsequence $[[pkt]_1^2]_1^3 = <t_{<<1,2>,3,5>}, t_{<7,8>}, \ldots>$. CP_6 takes a pair of enhanced subsequences $[[pkt]_1^2]_6^3 = <t_5, t_{11}, \ldots>$ from CP_1 and $[[pkt]_2^2]_6^3 = <t_1, t_{<7,<9,11>,12>}, \ldots>$ from CP_2 and merges them to $pkt_6 = <t_1, t_5, t_{11}, t_{<7,<9,11>,12>}, \ldots>$. Then, CP_6 sends packets in pkt_6.

In the other coordination protocol TCoP, contents peers CP_6 and CP_8 are selected by a pair of parents CP_1 and CP_2 and a pair of parents CP_2 and CP_3, respectively. Suppose CP_6 and CP_8 take contents peers CP_2 as the parent. CP_4 and CP_5 start transmitting packets following the packet t_3. The subsequence $[pkt]_1^2[t_5> = <t_5, t_{<7,8>}, t_9, t_{11}, t_{<13,14>}, \ldots>$ is enhanced by adding parity packets for parity interval $h = 2$. Here, a subsequence $<t_{<5,<7,8>>}, t_5, t_{<7,8>}, t_9, t_{<9,11>}, t_{11}, t_{<5,<13,14>>}, t_{15}, t_{<<13,14>,15>}, \ldots>$ is obtained. Here, the contents peers CP_1, CP_4, and CP_5 take subsequences $<t_{<5,<7,8>>}, t_9, t_{<13,14>}, \ldots>$, $<t_5, t_{<9,11>}, t_{15}, \ldots>$, and $<t_{<7,8>}, t_{11}, t_{<<7,8>,15>}>$, 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 ($\geq$ 1) 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 a contents peer CP_i and a leaf peer LP_s supports reliable high-speed communication like 10 Gbps Ethernet.

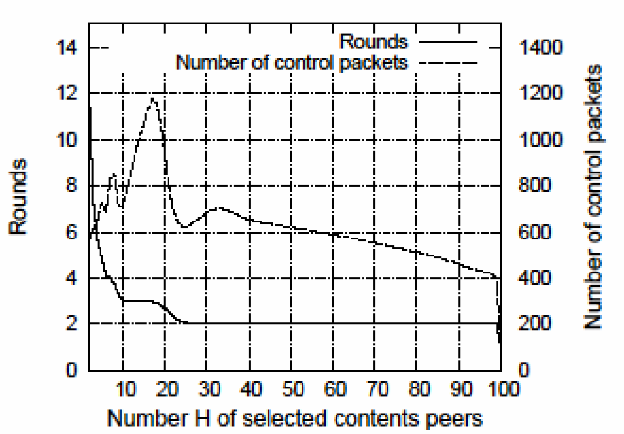


Fig. 9. Rounds and number of control packets in DCoP.

Figure 9 shows the number of control packets transmitted and how many rounds it takes to synchronize 100 contents peers in DCoP for each H ($2 \lesssim H \lesssim 100$). 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 10 shows the numbers of control packets and 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.

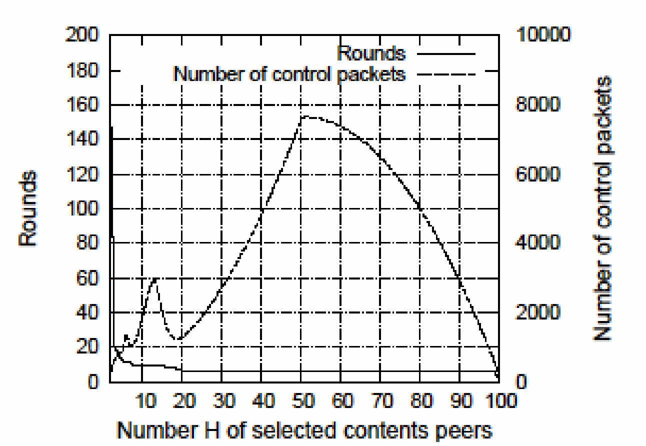


Fig. 10. Rounds and number of control packets in TCoP.

In DCoP and TCoP, one parity packet is transmitted for every $H - h$ packets. Figure 11 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.

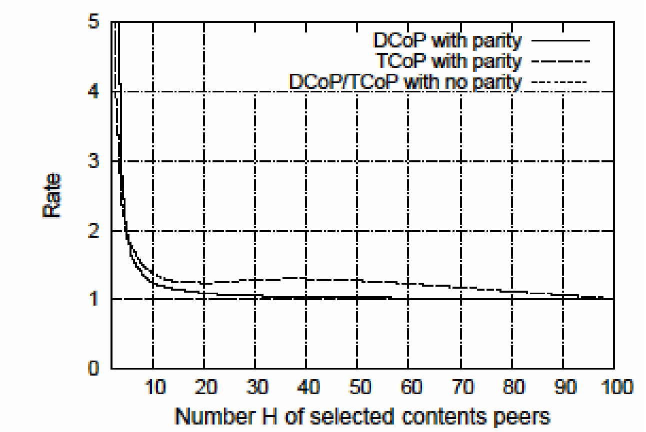


Fig. 11 Receipt rate of leaf peer.

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

Acknowledgements

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 Multisource 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. Massouli´e, 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.