A NETWORK CONSTRUCTION METHOD FOR A SCALABLE P2P VIDEO CONFERENCING SYSTEM

Hideto Horiuchi, Naoki Wakamiya and Masayuki Murata
Graduate School of Information Science and Technology, Osaka University
1–5 Yamadaoka, Suita-shi, Osaka 565–0871, Japan
{h-horiuti, wakamiya, murata}@ist.osaka-u.ac.jp

ABSTRACT

Recently, video conferencing systems based on peer-to-peer (P2P) networking technology have been widely deployed. However, most of them can only support up to a dozen of participants. In this paper, we propose a novel method to construct and manage a P2P network for scalable video conferencing. Our method consists of three parts: a network construction mechanism, a tree reorganization mechanism, and a failure recovery mechanism. First, a tree-formed network is eventually constructed as new peers join a conference. Then, the tree topology is dynamically reorganized taking into account the heterogeneity of the available bandwidth among peers and their degree of participation so that, those participants, i.e., peers that can have many child peers and/or often speak are located near the root of the tree. As a result, the delay from speakers to other participants is reduced. Through simulation experiments, we verify that our method can offer smooth video conferencing.

KEY WORDS

video conferencing, P2P, scalability, tree construction

1 Introduction

With the proliferation of the Internet, video conferencing systems are getting widely accepted making it possible to have a meeting or a discussion among people at different and distant places. Recently, video conferencing systems based on P2P communication have been introduced due to their ease of deployment and low cost of operation [1–3]. However, they have the scalability problem and most of them can only support at most a dozen of participants. For example, a company with worldwide branches and convenience chain stores may involve hundreds of managers in a business meeting. Commercial video conferencing products can support more users, but their scalability depend on Unified Conferencing Bridges (MCUs), which are quite expensive. There have been some research works for P2P video conferencing, but they still face the scalability issues [4–6]. There are some excellent algorithms for scalable ALM (Application Level Multicast), but they mainly consider distribution type of applications. Therefore, we need a video conferencing system that can accommodate hundreds or thousands of participants with low cost.

In this paper, we propose a novel method for constructing and managing a P2P network for a scalable video conferencing system. To attain the higher scalability, we incorporate two strategies, i.e., the hierarchical network structure and the promotion of active and/or rich peers. We assume that participants, i.e., peers, dynamically join and leave a conference. Peers are heterogeneous in terms of the network capacity available for video conferencing. Our proposed system consists of three parts: a *network construction mechanism*, a *tree reorganization mechanism*, and a *failure recovery mechanism*. The network construction mechanism sets up a hierarchical distribution network, which consists of distribution trees of tens or hundreds of peers, and a core network which interconnects these trees with each other. As far as both of delay in each of distribution trees and that among them, i.e., in the core network, are kept small enough, the number of participants can be easily increased by connecting many distribution trees by the core network. To have a smooth conference, it is necessary to keep the delay from speakers to other participants small. To accomplish this goal, we further focus on the fact that the number of simultaneous speakers is limited whereas speakers dynamically change in accordance with the agenda. Taking into account this, the tree reorganization mechanism dynamically reorganizes a distribution tree so that speakers are located near the root in a distribution tree. In addition, to reduce the height of a distribution tree, the tree reorganization mechanism dynamically moves peers with higher available bandwidth toward the root. Furthermore, in the case of failure in distribution of conference data due to a halt or disappearance of a peer, the failure recovery mechanism reconfigures the distribution network through local interactions among peers using local information acquired during network construction.

The rest of this paper is organized as follows. First, we describe our proposal in Section 2. Then, we present some simulation results in Section 3. Finally, we summarize the paper and describe some future works in Section 4.

2 Network Construction Method for Scalable P2P Video Conferencing

In this section, we give an overview of the scalable P2P video conferencing system. In the following, we use the

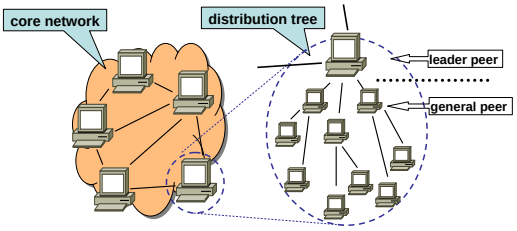


Figure 1. A hierarchical distribution network

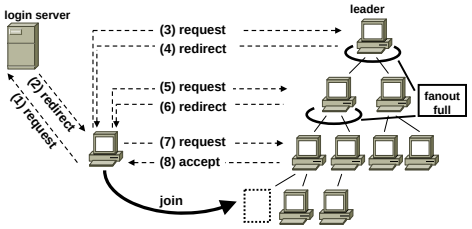


Figure 2. Participation to a tree through redirection

terms peer and participant interchangeably.

2.1 Overview of Scalable P2P Video Conferencing System

Our system consists of a login server, peers, and a distribution network constituting of the peers. Delivery and exchange of streaming data, i.e., video and audio are done through the distribution network. For low bandwidth requirement and management cost, we adopt a shared-tree architecture to the distribution network. The distribution network consists of the core network and the distribution trees whose root is connected to the core network as shown in Fig. 1. As far as both of delay in each of distribution trees and that among them are small enough, the number of participants can be easily increased by connecting more distribution trees by the core network. In this paper, we call a peer which belongs to the core network *leader peer*, and all other peers *general peers*. A leader peer manages the IP addresses of neighboring leader peers in the core network and all of its direct child peers. A general peer keeps the IP addresses of its parent and children. In addition, it manages the list of the IP addresses, which it knows, in its *ancestor list*. Peers have a limitation on the number of acceptable children called *fanout*, denoted as f, in accordance to their available bandwidth. The login server is responsible for registration and management of the conference, and the authentication of participants. It manages only information of leader peers and the number of general peers in each tree, and not the structure of each tree.

The overview of the system behavior is as follows. First, a newly participating peer requests the login server for authentication. At this time, the participating peer is notified whether it should become a leader peer or general peer. Next, it connects to either the core network or a distribution tree to join the conference. Then, the participant is involved in the conference as a speaker or an audience in accordance with the agenda. Since we do not consider any management of speech coordination in this proposal, all participants can speak whenever they want. Streaming data from a speaker is once transmitted to the root of the tree to which it belongs, and then broadcasted to the other peers in the tree and to peers in the other trees via the core network. Our method makes peers with high fanout, i.e., high bandwidth, and active speaking move to the root of tree. We call this *promotion*. The promotion reduces the tree height and delay between active peers and others. In video conferencing systems, peers may leave because of failures in routers or links. So our method dynamically recovers from the failure in the distribution network so that peers can continuously receive streaming data.

2.2 Network Construction Mechanism

With consideration of the fanout of a newly participating peer and the number of peers in each tree, the login server determines the role of the peer. If a participating peer is determined as a leader peer, the peer gets the IP address list of other leader peers, measures delay to them, and connects to the neighbor peers.

If the participating peer is specified as a general peer, it connects to the designated distribution tree by being introduced temporary parents as shown in Fig. 2 [7]. First, the login server notifies the participating peer of the IP address of an appropriate leader peer as a temporary parent (Fig. 2:1-2). In our mechanism, the leader peer to be introduced is selected in a round-robin fashion. Therefore, without any peer leaving, the number of peers is equal among trees. The participating peer deposits the notified IP address in its ancestor list and sends a participation request message to the temporary parent (Fig. 2:3).

If it has less than $f-1$ children, the temporary parent accepts the request and connects to the peer. The reason for comparing with $f-1$ is that the tree reorganization mechanism requires one spare link as will be explained later. Otherwise, the temporary parent introduces one of its direct children to the participating peer as a new temporary parent. We call this procedure *redirect* (Fig. 2:4). Since a new temporary parent is selected among children in a round-robin fashion, a distribution tree is constructed in breadth-first order, and the delay from the leader peer can be minimized. We can adopt other scheme to have a balanced tree or well-organized tree, such as proposed for ALM, as far as it is worth consuming network resources in exchanging control messages for tree management. The participating peer adds the IP address of the introduced temporary parent to its ancestor list and sends a participation request message (Fig. 2:5). By repeating these try-and-redirect, the participating peer can eventually connect with a temporary parent, which has an available link, and join the distribution tree (Fig. 2:6-8). The participating peer has all IP addresses of its ancestors in the ancestor list at this time.

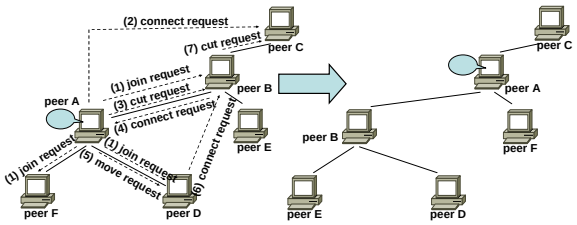


Figure 3. Promotion of peer A for speaking

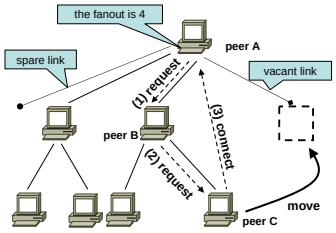


Figure 4. Completing the fanout of peer A

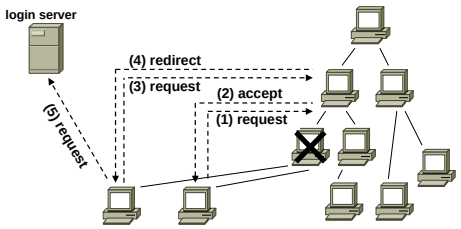


Figure 5. Failure recovery

2.3 Tree Reorganization Mechanism

In our method, a peer with high activity and high fanout moves to the root of the tree for low delay and smooth conferencing. We call it *promotion*. In addition, to reduce tree height by completing the fanout, a peer which has less than $f-1$ children invites its grandchild as a direct child. When a peer has already been involved in tree organization or failure recovery, the peer is considered *locked* and it rejects a request for another tree reorganization or failure recovery.

2.3.1 Peer Promotion

A peer starts the promotion process if it speaks continuously. Additionally, a peer compares its fanout with that of its parent periodically. If the fanout is more than its parent, the peer starts the promotion process. However, the promotion process does not occur if the peer is involved in other tree reorganization or failure recovery, i.e., locked. The promotion means that a peer becomes a child of its grandparent as shown in Fig. 3.

Firstly, peer A which starts the promotion sends a promotion request message to parent peer B and its children (Fig. 3:1). If a peer receiving the request is not locked, it sends back an accept message. The accept message from peer B has the IP address of its parent peer C. On receiving accept messages from all peers, peer A sends a connection request message to peer C (Fig. 3:2). If peer C is not locked, it makes a connection to peer A and sends back an accept message. If the number of children becomes equal to the fanout on peer C, the accept message from peer C includes information indicating that the spare link is used.

After connecting with peer C, peer A sends a disconnection request message to its previous parent B (Fig. 3:3). This request includes information whether the spare link of peer C is used. After receiving the request, peer B terminates the connection with peer A. If the spare link is not used on peer C, the promotion is completed at this time. Now, both peer A and B are children of peer C.

If the spare link is used on peer C, peer B becomes a child of peer A to make one link free on peer C. First, peer B sends an adoption request message to peer A (Fig. 3:4). If the number of children is less than $f-1$ on peer A, peer A accepts peer B as its child. On the other hand if equal, peer A sends a moving request message to peer D which is selected from peer A's children in a round-robin fashion (Fig. 3:5). The request message includes the IP address of peer B. Then peer D becomes a child of peer B by sending a connection request message to peer B (Fig. 3:6) and terminating the connection with peer A. At the same time, peer B becomes a child of peer A and disconnects the connection with peer C (Fig. 3:7). As a result, peer C obtains a new spare link. In this way, the promotion is completed.

2.3.2 Completing the Fanout

Peer A, which can accommodate more children, sends an introduction request message to peer B which is selected in a round-robin fashion from its children (Fig. 4:1). If peer B has any children, it sends a moving request message to peer C which is selected in a round-robin fashion from its children (Fig. 4:2). The moving request includes the IP address of peer A. Peer C sends a connection request to peer A (Fig. 4:3) and makes the connection. Then peer C terminates the connection with peer B and this process is completed. If either of peer B and C or both are locked, peer A receives a reject message and the process is canceled.

2.4 Failure Recovery Mechanism

A peer may become to be unable to receive data due to not being able of accessing its temporary parent in tree construction/reorganization, a halt of links or routers, or a parent peer leaving the conference. We define this event as *failure*. In the failure recovery mechanism, a peer detecting a failure tries to make a new connection with another peer in its ancestor list [7].

If a peer fails in sending a message to a temporary parent, it sends a re-connection request message to the previous temporary parent, which introduced the missing temporary parent. On the other hand, if a peer detects the leaving

or a fault of its parent, it chooses its grandparent in the ancestor list as a new temporary peer to send a re-connection request message (Fig. 5:1,3). In both cases, the IP address of the missing parent is removed from the ancestor list. If the recovering peer fails in sending the re-connection request message to the new temporary parent, it first removes the corresponding IP address from the ancestor list and then moves to the next ancestor at the bottom of the list.

On receiving the re-connection request message, the temporary parent compares its distance, i.e., the number of hops from the leader peer, with recovering peer's distance. If the former is more than the latter, it sends back a reject message. Then the recovering peer removes the corresponding IP address from the ancestor list and then sends the re-connection request message to the next ancestor at the bottom of the list. This process prevents a distribution tree making a loop. If the distance of the temporary parent is closer to the leader peer, it establishes a connection with the recovering peer if the number of children is less than $f - 1$ (Fig. 5:2), or introduces a child to the recovering peer as a new temporary parent otherwise (Fig. 5:4). In the latter case, the requesting peer eventually joins the tree and reorganizes its ancestor list by the same process as the initial join. If the list becomes empty, the recovering peer goes to the login server and joins the distribution tree again as a new peer. (Fig. 5:5).

If the failure of the leader peer occures, its child notifies the login server of the failure. The login server appoints the peer which first sends the notification as a new leader peer and updates the information of leader peers. The other children of the missing leader peer also report the failure and are redirected to the new leader peer.

3 Simulation Experiments

In this section, we evaluate our method from a viewpoint of smoothness of video conferencing.

3.1 Simulation Conditions

Conference data originating from a speaking peer are first sent to the leader peer of the designated tree, then distributed to the other peers in the same tree and peers in the other distribution trees through the core network. Since the major contribution of this paper is mainly in the management of a distribution tree, we focus on the performance and effectiveness of our method in managing a distribution tree. Evaluation of the whole system including both of the core network and distribution trees is left as a future work. First, we create a physical network based on the BA model [8] using BRITE. The average degree is set at 2. We also conducted simulation experiments on networks of random topology generated by the Waxman algorithm [9], but we observed similar results and omit them from the paper due to space limitation. A network consists of 101 peers and one among them is designated as a login server. We assume that the ratio of the type of access link is ADSL : FTTH : CATV$= 6 : 3 : 1$ [10]. Taking into account their typical capacity, we define the fanout for each type of access link as 2, 7, and 4, respectively. The delay between an arbitrary pair of peers is computed as the product of the physical hops of the shortest path by the Dijkstra algorithm and the one-hop delay of 1 msec. We do not consider transmission delay and processing delay.

Peers participate in the conference at random time. The first participating peer becomes a leader peer. After all of 100 peers participate in the conference and construct the tree, no further peer joins.

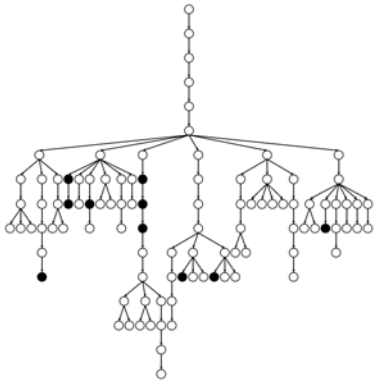
After all peers participate in the conference, a peer begins to speak and the tree reorganization is conducted. We define peers to speak as *candidates*. Ten candidates are randomly chosen at the start and are fixed during the simulation. The duration of each speech is exponentially distributed with a mean value of 6 seconds and the minimum duration is 1 msec. Any one of the candidates is always speaking during the simulation. In other words, when a candidate stops speaking, the next speaker is randomly chosen among candidates and starts speaking immediately. The same candidate would be chosen as the next speaker, but only one candidate speaks at the same time. In the following figures, time zero corresponds to the instant when the first speech starts.

A speaking peer starts the promotion when it continuously speaks for more than 5 seconds, and as long as it is speaking, it tries the promotion every 5 seconds. However, as described in Section 2.3, if the preceding promotion is not completed, the next promotion is not triggered. All peers compare its fanout with its parent every 24 seconds for possible promotion. To distribute the timing of the promotion among peers, the first comparison occurs at a random time with uniform distribution from 0 to 24 seconds. Peers compare the number of their children with their fanout every 7 seconds and they may start completing the fanout depending on the result. To distribute the timing, the first comparison occurs at random time with uniform distribution from 0 to 7 seconds.

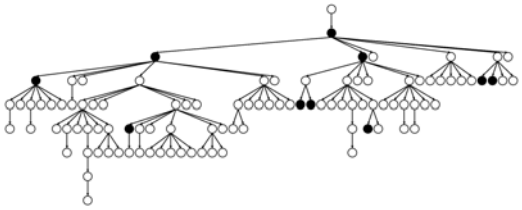
We evaluate our method from the viewpoint of the delay from all peers to the leader peer and from all candidates to the leader peer, the number and duration of freezes, and the number of messages per physical link and per peer. In the figures, we also show results of the case that a distribution tree does not change during a simulation experiment, denoted as *static*, to compare with results of the case with the tree reorganization mechanism, denoted as *dynamic*. Following results are the average over 1000 simulation experiments, each of which lasts for 30 minutes in simulation time unit after the first speaker begins to speak.

3.1.1 Simulation Results

First, we evaluate the effectiveness of the tree reorganization mechanism without failures. In Fig. 6, how a tree was reorganized in a certain simulation run is illustrated.

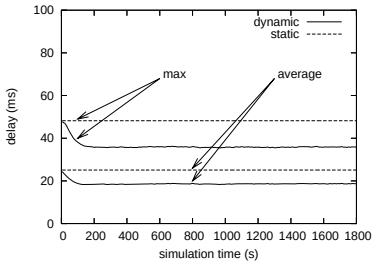


(a) The initial tree topology

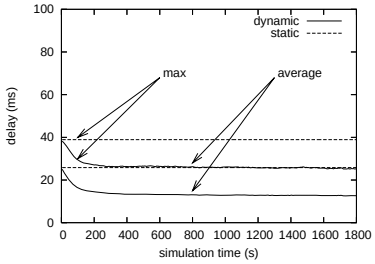


(b) The reorganized tree topology

Figure 6. Result of tree reorganization



(a) All peers



(b) Candidates

Figure 7. Delay between the leader peer and peers

In these figures, filled circles correspond to the candidates and open circles indicate other peers. The figures show that the tree reorganization mechanism reduces the height of the tree. With the 1000 simulation experiments, the average distance from the leader peer to all peers is reduced from about 5.3 hops to about 4.1 hops, and the maximum distance changes from about 9.9 hops to 7.5 hops by promotion related to fanout comparison and completion. Furthermore, we can see that the candidates have moved near the root of the tree. With 1000 simulation experiments, the average distance from the leader peer and candidates decreases from about 5.6 hops to about 2.8 hops, and the maximum distance changes from about 8.4 hops to 5.4 hops by promotion for speaking. However, all candidates are not necessarily located near the root depending on the timing of speaking or the duration of speaking, as shown in Fig. 6(b).

Fig. 7(a) illustrates the average and maximum delay between the leader peer and all peers. The figure shows that the tree reorganization mechanism can effectively reduce both delay. Among promotions, those invoked by fanout comparison and completion mainly contribute to the initial reduction of delay. When the maximum delay between the leader peer and all peers in a distribution tree is D and that between leader peers in the core network is L, the maximum end-to-end delay among all peers can be derived as $D \times 2 + L$. Except for the initial stage, D is about 35 msecs in Fig. 7(a). Therefore, if we can construct a core network in which the delay among leader peers is less than 30 msecs, we can offer video conferencing with the end-to-end delay less than 100 msecs, which is smaller than the recommended one way delay for voice communication [11]. Fig. 7(b) shows the average and maximum delay between the leader peer and candidates. When comparing to Fig. 7(a), the delay for candidates is less than that for all peers. It means that speakers have better and smoother conversation.

Table 1. Summary of freezes

		Number	Duration
Average	All peers	0.104	1.33
	Candidates	0.067	0.41
Max	All peers	1.35	6.01
	Candidates	1.4	6.11

Next, we consider the case with failures. The probability that a peer leaves the conference is defined as one leaving in 5 minutes for 100 peers. Candidates and locked peers do not leave. If a peer does not receive any response from a temporary parent for the timeout period, which is set at 3 seconds in the simulation experiments, it considers that the temporary parent disappeared or failed. A peer cannot receive streaming data when a path from the leader peer is lost or disconnected. We define this as *freeze*.

Table 1 summarizes the results on freeze that all remaining peers at the end of the simulation experiments experienced. On average, one peer among ten experiences a freeze of about 1.3 seconds during a 30 minutes-long conference. The most of freezes last only for several tens of msecs, except that some peers wait for the timeout of 3

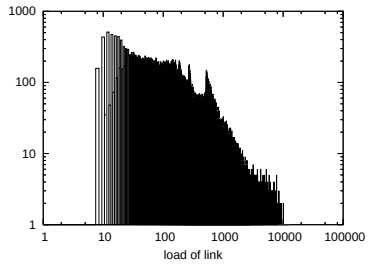


Figure 8. Frequency distribution of the load on link

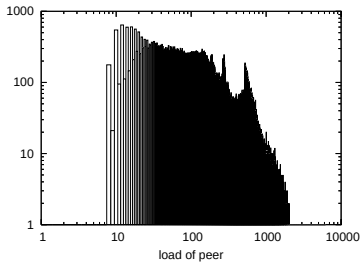


Figure 9. Frequency distribution of the load on peer

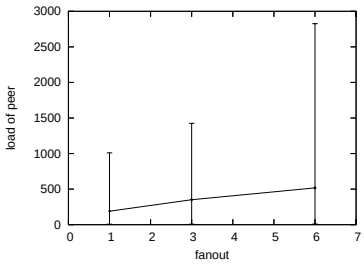


Figure 10. Relationship between the load and fanout

seconds to detect the disappearance of a temporary parent peer. Since candidates are moved close to the leader peer, they experience less loss or disconnection of a path to the leader peer. Consequently, the probability and duration of freeze are smaller than those of all peers. Among 1000 simulation experiments, the maximum number of freeze is more than one and a freeze is as long as 6 seconds. However, this seldom occurs.

Fig. 8 illustrates the frequency distribution of load on links. The average is 701 messages and the maximum is 6924 messages. Assuming that the size of a message is 5 Bytes and the packet size including the header is 33 Bytes, the bandwidth consumed by control messages per link is 103 bps on average and 1016 bps at maximum. This is considerably smaller than the typical rate of streaming data, i.e., from 64 Kbps to 8 Mbps.

Fig. 9 illustrates the frequency distribution of load on peers. The number of messages that a peer handles is 305 on average and 1554 at maximum. It consumes 45 bps on average and 228 bps at maximum. Most of commercially available equipment can afford this. At the worst case scenario with 1000 peers, the maximum load on link is about 10 Kbps and that on peer is about 2 Kbps assuming that the load is proportional to the number of peers. They are small enough and thus our system is scalable.

Finally, Fig. 10 shows the relationship between the load on peer and its fanout. A solid line connecting error bars stands for the average. There is a tendency that a peer with larger fanout hundles more control messages, but it does not always hold depending on the initial location in the tree and whether it is a candidate or not.

4 Conclusion

In this paper, we proposed a network construction method for a scalable P2P conferencing system. We showed that our mechanism can offer smooth video conferencing with low delay and seldom and short freezes. In addition, we showed that the load of control messages was very low. By connecting distribution trees by the core network, our system can accommodate hundreds or thousands of participants with the acceptable delay.

Although we verified that a distribution tree is well organized and maintained, we plan to extend our experiments to the whole distribution network consisting of a core network and several distribution trees.

References

[1] "Skype." available at `http://skype.com/`.

[2] "iChat." available at `http://www.apple.com/macosx/features/ichat/`.

[3] "NetMeeting." available at `http://www.microsoft.com/windows/netmeeting/`.

[4] F. Lanubile, F. Calefato, and T. Mallardo, "Peer-to-peer remote conferencing," in *Proceedings of the ICSE Workshop on Global Software Development*, pp. 34–38, May 2004.

[5] H. K. Kim and J. N. Hwang, "Design and implementation of desktop video conference system based on client-server and P2P," in *Proceedings of International Conference on Communications in Computing*, pp. 158–161, June 2006.

[6] M. R. Civanlar, O. Ozkasap, and T. Celebi, "Peer-to-peer multipoint videoconferencing on the Internet," *Signal Processing: Image Communication*, vol. 20, pp. 743–754, Sept. 2005.

[7] S. Suetsugu, N. Wakamiya, and M. Murata, "A hybrid video streaming scheme on hierarchical P2P networks," in *Proceedings of EuroIMSA 2005*, pp. 240–245, Feb. 2005.

[8] A. Barabasi and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, pp. 509–512, Oct. 1999.

[9] B. M. Waxman, "Routing of multipoint connections," *IEEE Journal on Selected Areas in Communications*, vol. 6, pp. 1617–1622, Dec. 1988.

[10] I. A. Japan, ed., *Internet White Paper*. Impress R & D, 2006.

[11] ITU-T, "Recommendation G.114 - one-way transmission time," *Switzerland*, Feb. 2003.