Master's Thesis

Title

A Construction Method of an Overlay Network for Scalable P2P Video Conferencing Systems

Supervisor

Professor Masayuki Murata

Author

Hideto Horiuchi

February 14th, 2007

Department of Information Networking

Graduate School of Information Science and Technology

Osaka University

Master’s Thesis

A Construction Method of an Overlay Network for Scalable P2P Video Conferencing Systems

Hideto Horiuchi

Abstract

Recently, video conferencing systems based on Peer-to-Peer (P2P) networking technology have been widely deployed due to their ease of deployment and low cost of operation. However, most of them can only support up to a dozen of participants against the emerging needs for a large scale conference involving hundreds or thousand attendees, e.g., a manager meeting of a company with worldwide branches. In this thesis, we propose a novel method to construct and manage a P2P network for scalable and smooth video conferencing. For smooth conferencing, it is necessary to keep the delay from speakers to participants small. To achieve this goal, we focus on the fact that the number of simultaneous speakers is limited in a conference. Our method consists of three parts: a network construction mechanism, a tree reorganization mechanism, and a failure recovery mechanism. A distribution network has a hierarchical structure and it consists of several distribution trees and a core network connecting them. First, a hierarchical distribution network is eventually constructed as new peers join a conference. Then, the distribution 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. This movement of peers towards the root is called *p*romotion. As a result, the delay from speakers to other participants is reduced. In addition, a peer which has room for more child peers invites a grandchild as a direct child. When a failure occurs, the tree is recovered based on information that an affected node obtains when it joins the distribution network. All of construction, reorganization, and recovery are done through local communication among peers and there is no centralized control. Through simulation experiments, we verify that our method can offer smooth video conferencing for 1000 participants where the delay among speakers is about 40 msecs for 10 speakers to 70 msecs for 500 speakers, and the load of our method is very small.

Keywords

Video Conferencing

P2P (Peer-to-Peer)

Scalability

End-to-end Delay

Contents

List of Figures

List of Tables

1 Introduction

With the proliferation of the Internet, video conferencing systems are getting widely accepted making it possible to have meetings or discussions among people at different and distant locations. However, for supporting a considerable number of participants at an acceptable quality, a conference server must have high processing capability and large network capacity. This apparently comes at a high cost but it can not fully avoid server bottlenecks. IP Multicast is an attractive technology that reduces the bandwidth requirement, but it is not readily available in the current Internet due to slow development and deployment.

Consequently, video conferencing systems using P2P communication have been introduced due to their ease of deployment and low cost of operation [1–6] to answer increasing demand for having a large-scale meeting. For example, a company with worldwide branches may involve hundreds of managers in a business meeting. However, those P2P-based systems still have the scalability problem and most of them can only support at most a dozen of participants. Commercial video conferencing products can support more users, but their scalability depend on Unified Conferencing Bridges (MCUs), which are quite expensive [7, 8]. There have been some research works for P2P video conferencing, but their scalability is not verified [9–11]. There are some excellent algorithms for scalable ALM (Application Level Multicast), but they mainly consider distribution type of applications [12–14] and cannot be directly applied to an interactive online meeting. Therefore, we need a video conferencing system that can accommodate hundreds or thousands of interactive participants with low cost. In video conferencing systems, it is necessary to reduce the delay to deliver and exchange streaming data, i.e., audio and video data for a conference, among participants and to maintain the quality of streaming data in order to provide smooth conferencing.

In this thesis, we propose a novel method for constructing and managing a P2P network for a scalable video conferencing system. 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. To attain the higher scalability, we incorporate two strategies, i.e., the hierarchical network structure and the promotion of active and/or rich peers.

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 conference agenda. It means that the degree of participation is different among peers. Some participants speak very often, but others do not. 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 streaming 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 thesis is organized as follows. First, we describe our proposal in Section 2. Then, we present some simulation results in Section 3 for a single distribution tree and Section 4 for a distribution network. Finally, we summarize the thesis and describe some future works in Section 5.

2 Construction Method of Overlay Network for Video Data Streaming

In this section, we give an overview of scalable P2P video conferencing systems and our proposal for constructing and managing a P2P network for a scalable video conference. In the following, we use the terms peer and participant interchangeably.

2.1 Overview of Scalable P2P Video Conferencing Systems

Our system consists of a login server, peers, and a distribution network constituting of the peers (Fig. 1). Delivery and exchange of streaming data, i.e., video and audio data are done through a distribution network. For low bandwidth requirement and management cost, we adopt a shared-tree architecture to a distribution network. A distribution network consists of a core network and distribution trees whose root is connected to the core network as shown in Fig. 1. Interconnecting distribution trees by a core network, our system can support thousands of participants. 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 thesis, 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 with their available bandwidth. For example, assuming that the capacity of an access link for uploading is 1 Mbps and the rate of streaming data is 256 Kbps, the peer can have one parent and three children, i.e., $f = 3$. 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, but not the structure of each trees. For scalability reasons, the topologies of distribution trees are managed by decentralized control mechanisms.

The overview of system behavior is as follows. First, a newly participating peer requests the login server for authentication. At this time, the participating peer is notified of its role, i.e, 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

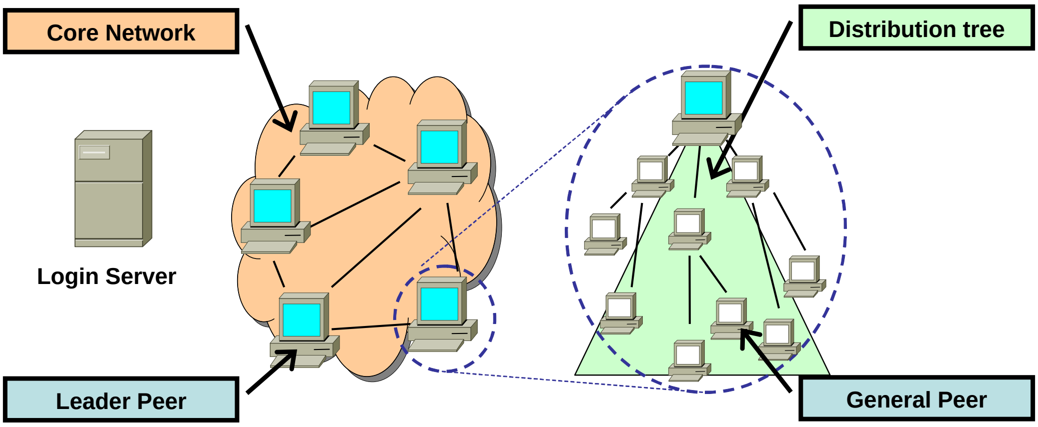


Figure 1: A hierarchical distribution network

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 designated tree to which it belongs, and then broadcasted to the other peers in the tree and peers in the other trees via the core network. Our method makes peers with high fanout, i.e., high bandwidth, and actively speaking move to the root of tree. We call this *promotion*. The promotion reduces the tree height and the delay between actively speaking peers and others. In video conferencing systems, peers may leave the conference because of failures in routers or links, or their will. Our method dynamically recovers from the failure in the distribution network so that peers can continuously receive streaming data.

2.2 Network Construction Mechanism

In our method, a participating peer first gets authenticated by the login server and then connects to the distribution network. 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 addresses of other leader peers, measures delay to them, and connects to the nearest leaders.

If the participating peer is appointed as a general peer, it connects to the designated distribution tree by being introduced temporary parents as shown in Fig. 2 [15]. 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

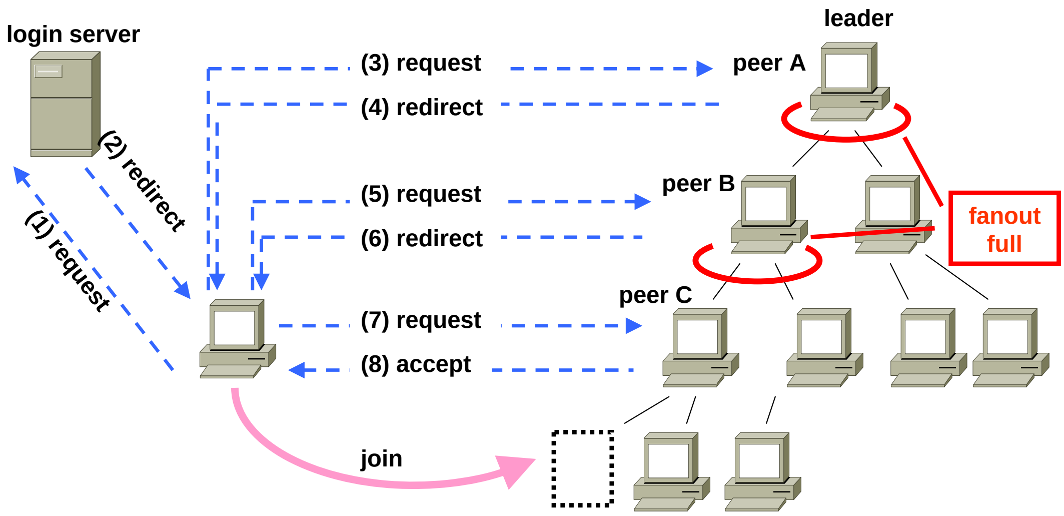


Figure 2: Participation to a tree through redirection

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). The temporary parent which receives the participation request message compares its fanout with the number of children.

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. On the other hand, if the number of children is equal to $f - 1$, 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. In this thesis, we adopt the round-robin scheme for its simplicity and small overhead. On receiving the redirect message, the participating peer adds the IP address of the introduced newly 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,

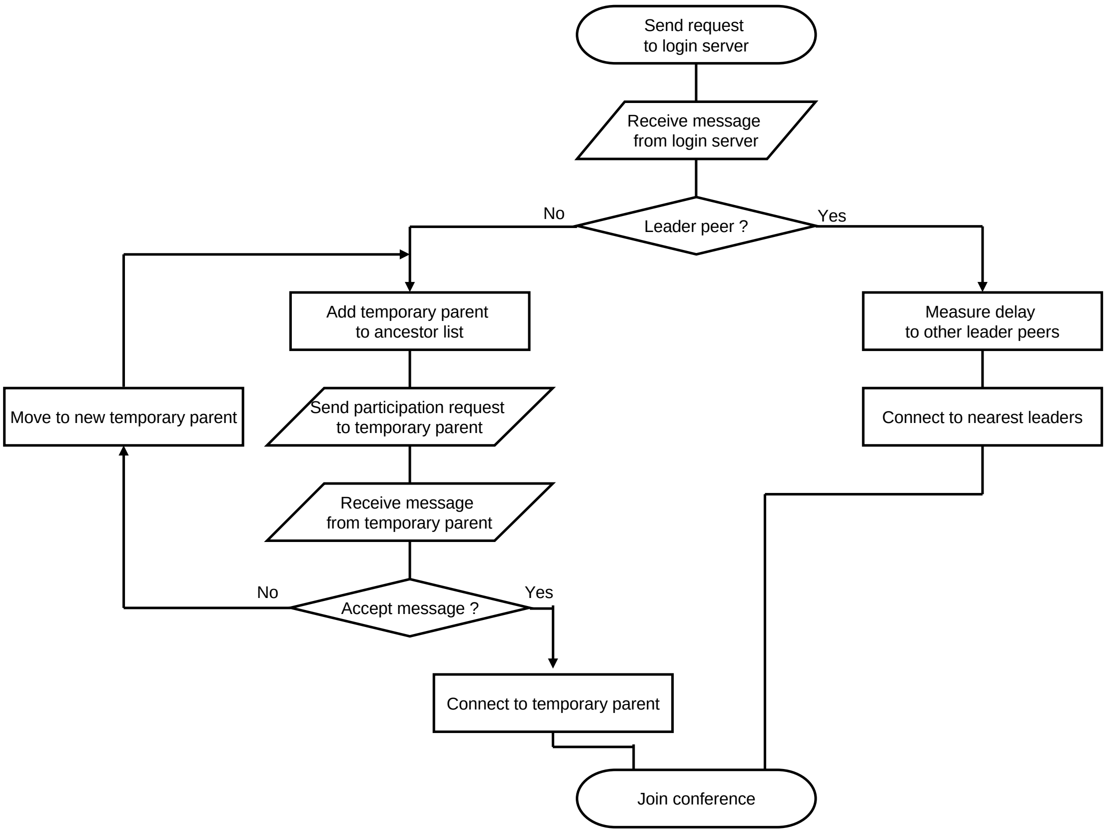


Figure 3: Flowchart: participating peer

e.g., peers A, B, and C in Fig. 2, in the ancestor list at this time. In this mechanism, there is only small overhead at the login server and peers, because no centralized unit manages the distribution tree topology and the additional load of processing messages occurs only at temporary parents. An additional advantage is that a peer can reconfigure the distribution tree during failure with only local interaction because a peer has the knowledge of the complete ancestor list. Figures 3 and 4 show flowcharts of a participating peer and a temporary parent for the tree construction mechanism.

2.3 Tree Reorganization Mechanism

In our method, a peer with high activity, i.e., actively participating in the conference by speaking, and high fanout, i.e., much capacity of access link, moves to the root of the tree for low delay and

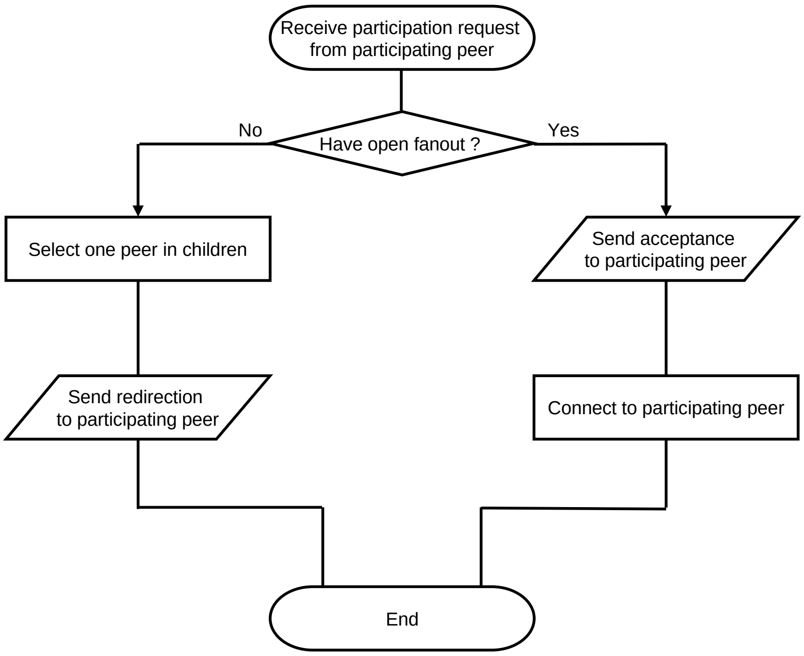


Figure 4: Flowchart: temporary parent of participating peer

smooth conferencing. We call it *promotion*. In addition, to reduce the tree height, a peer with open fanout, that is, a peer with less than $f - 1$ children, then invites one of its grandchildren as a direct child to complete the fanout..

In this section, we describe the details of the reorganization mechanism. 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 promotion when it keeps speaking for a certain period of time. Additionally, a peer compares its fanout with that of its parent periodically. If the fanout is more than its parent, the peer starts a 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. 5. Firstly, peer A which starts the promotion sends a promotion request message to parent peer B and its children (Fig. 5:1). If a peer receiving the request is not locked, it sends back an accept message.

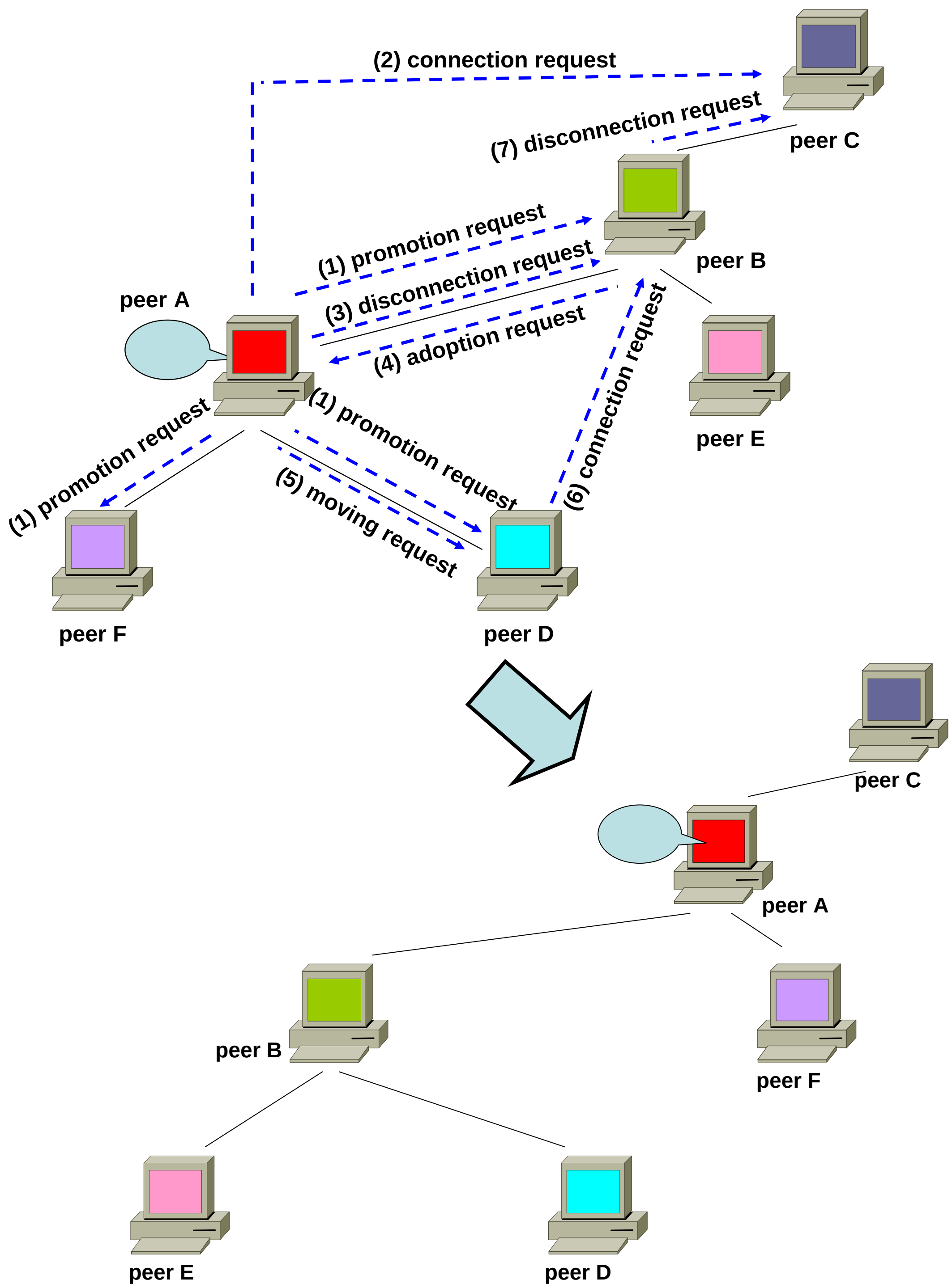


Figure 5: Promotion of peer A for speaking

The accept message from peer B has the IP address of its parent peer C. Otherwise, if a peer receiving the request is locked, it rejects the request and the promotion process is canceled. On receiving accept messages from all peers, peer A sends a connection request message to peer C (Fig. 5:2). If peer C is locked, peer C sends back a reject message, peer A sends a cancel request message to parent peer B and its children and the promotion is canceled. Otherwise peer C 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. 5:3). 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 spare link free on peer C. First, peer B sends an adoption request message to peer A (Fig. 5:4). 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. 5:5). 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. 5:6) and terminating the connection with peer A, then peer A gets open fanout. At the same time, peer B becomes a child of peer A.

After peer B become a child of peer A, peer B disconnects the connection with peer C (Fig. 5:7). As a result, peer C obtains a new spare link. In this way, the promotion is completed. Note that all peers involved in the promotion have at least one link with a peer closer to the leader peer to continuously receive stream data.

Figures 6 through 12 show flowcharts of a promoting peer and other peers for the promotion.

2.3.2 Completing Fanout

Peers periodically compare the number of their children with the fanout. If the number is less than $f - 1$, a peer starts completing the fanout. However, if a peer is involved in other reorganization or failure recovery, the process does not occur. Peer A, which can accommodate more children,

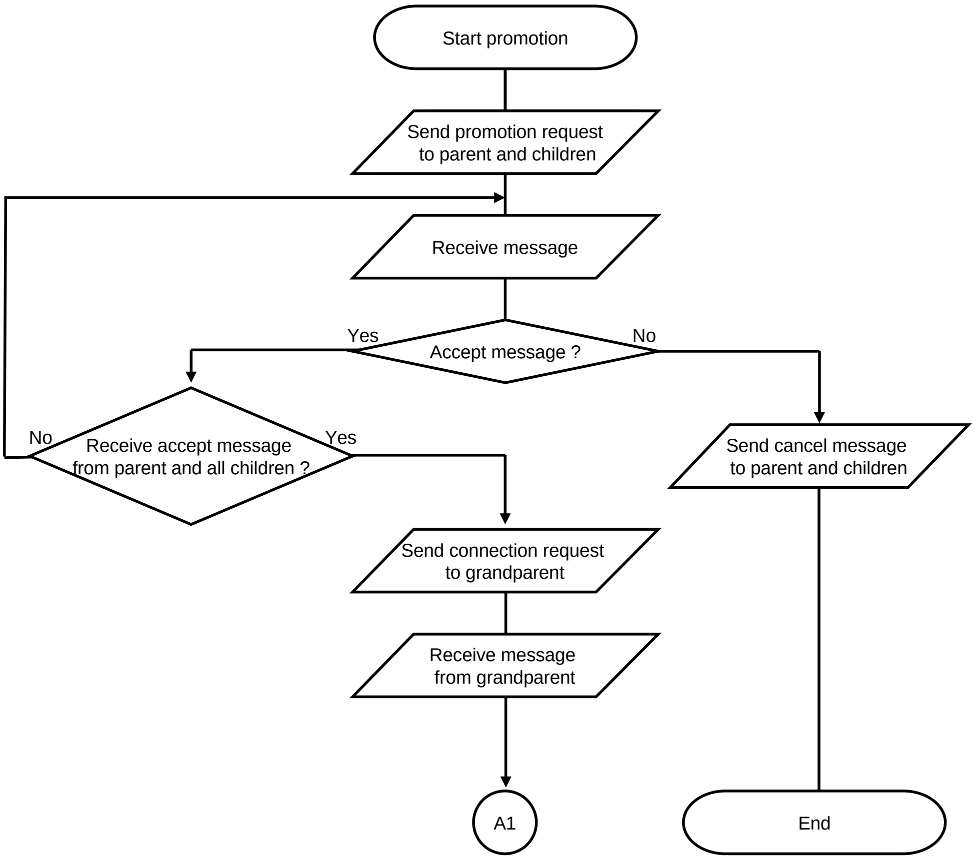


Figure 6: Flowchart: promoting peer

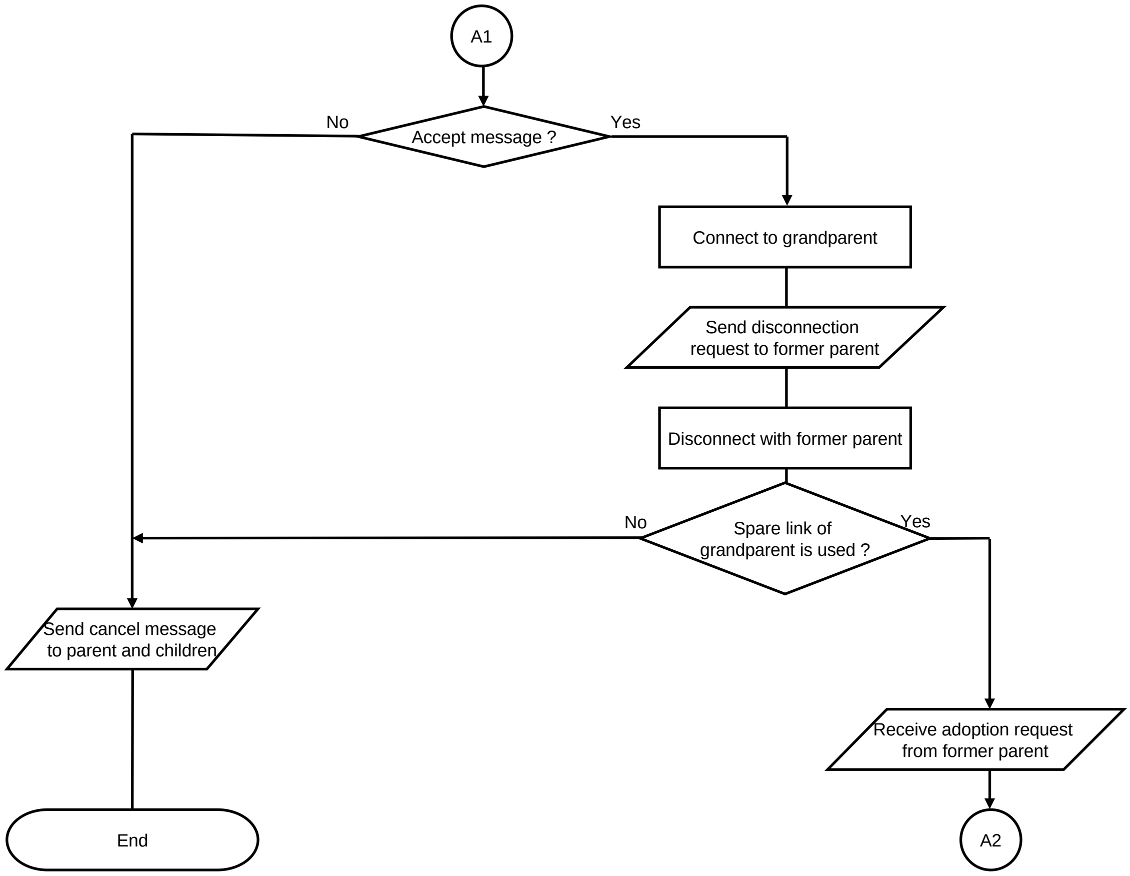


Figure 7: Flowchart: promoting peer (continues from Fig. 6)

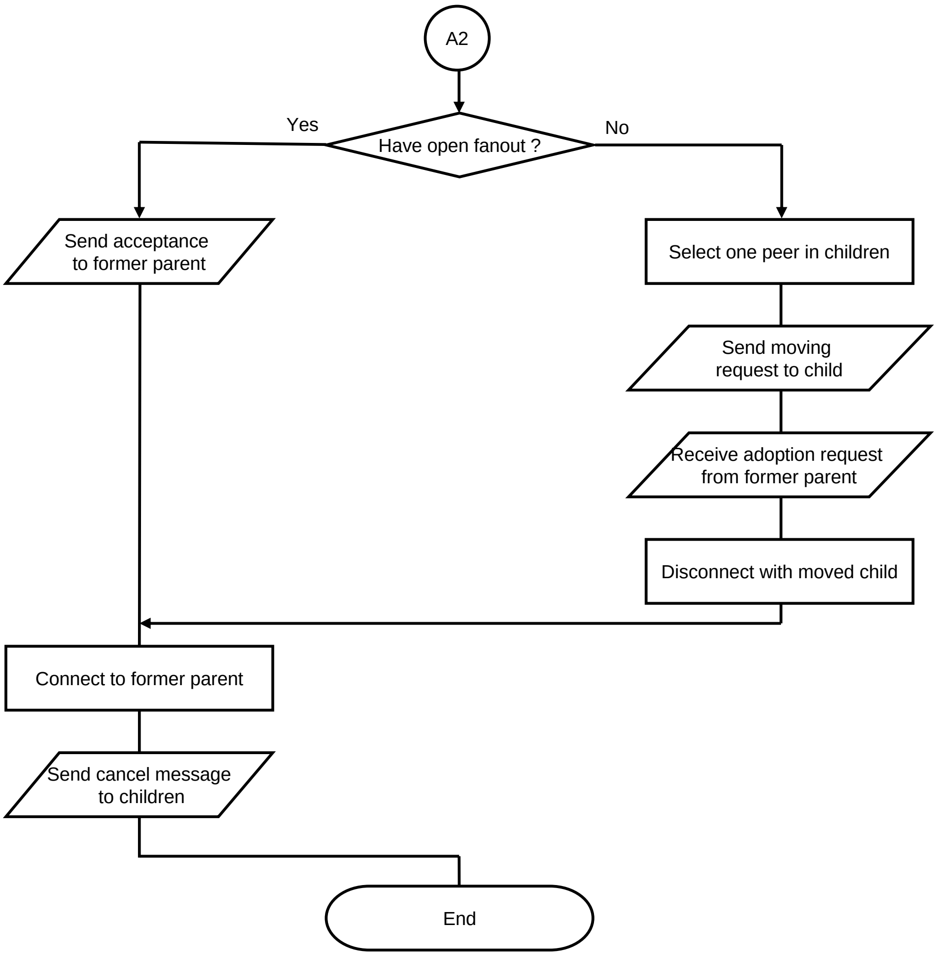


Figure 8: Flowchart: promoting peer (continues from Fig. 7)

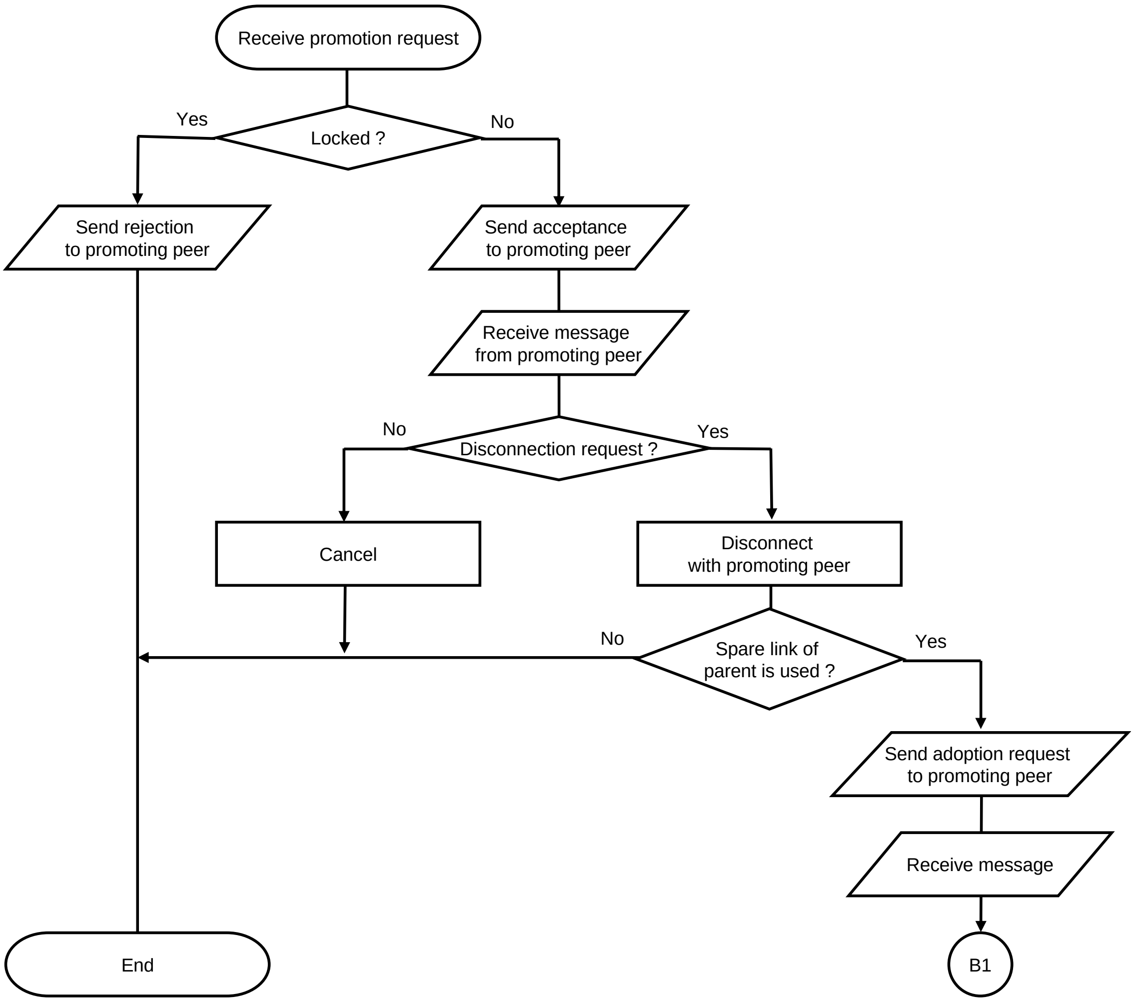


Figure 9: Flowchart: parent peer of promoting peer

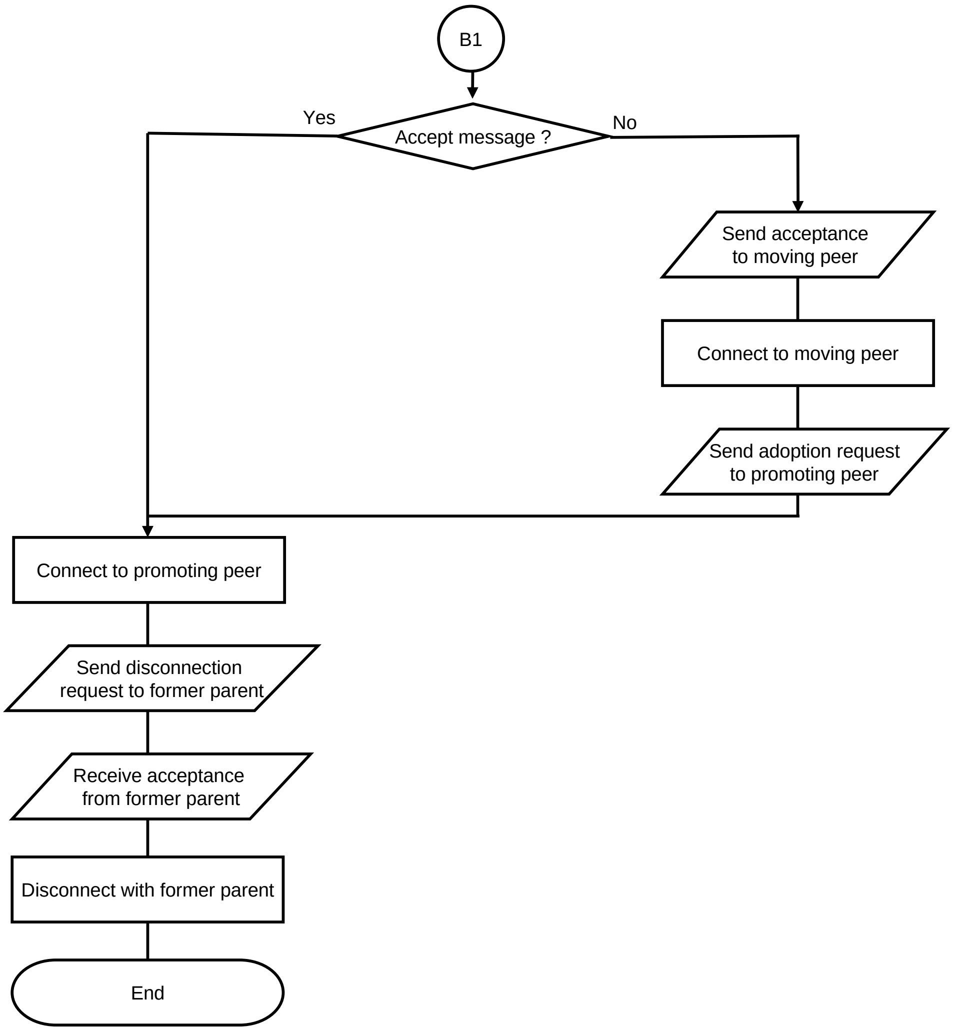


Figure 10: Flowchart: parent of promoting peer (continues from Fig. 9)

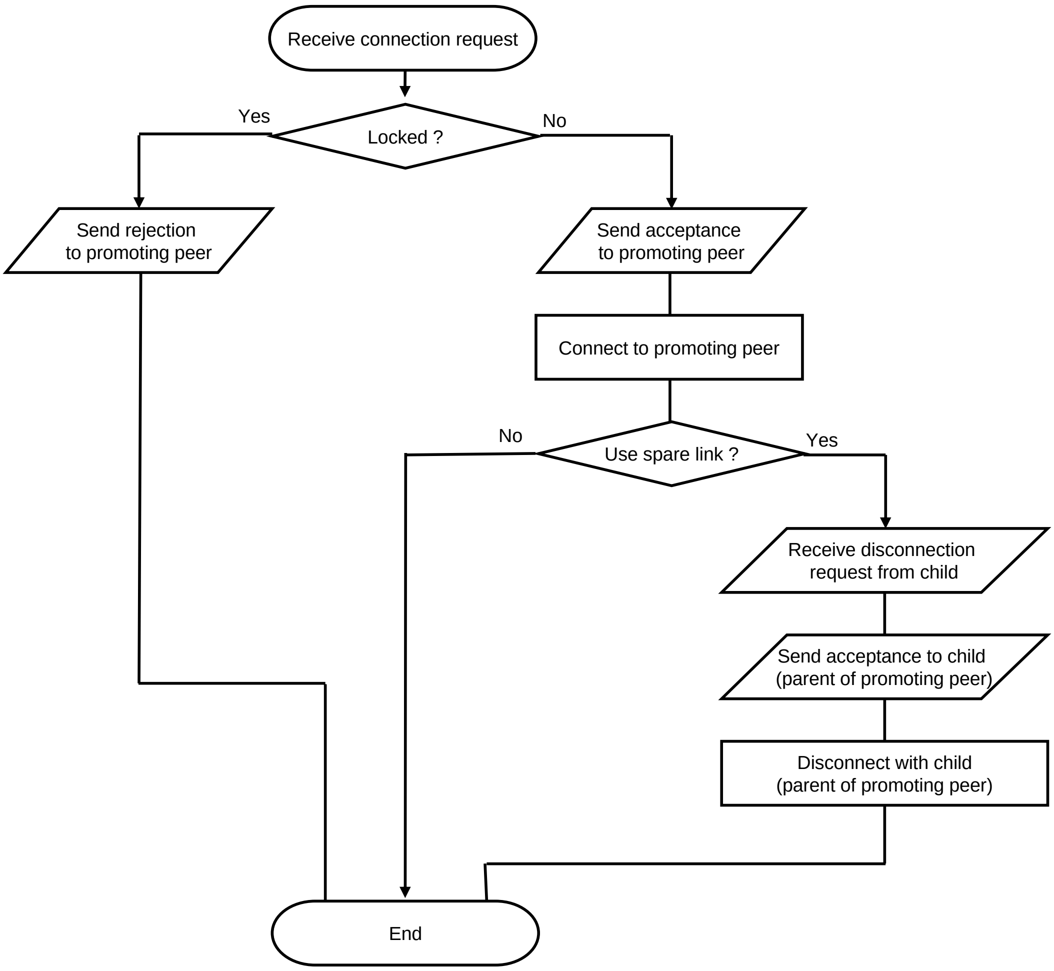


Figure 11: Flowchart: grandparent of promoting peer

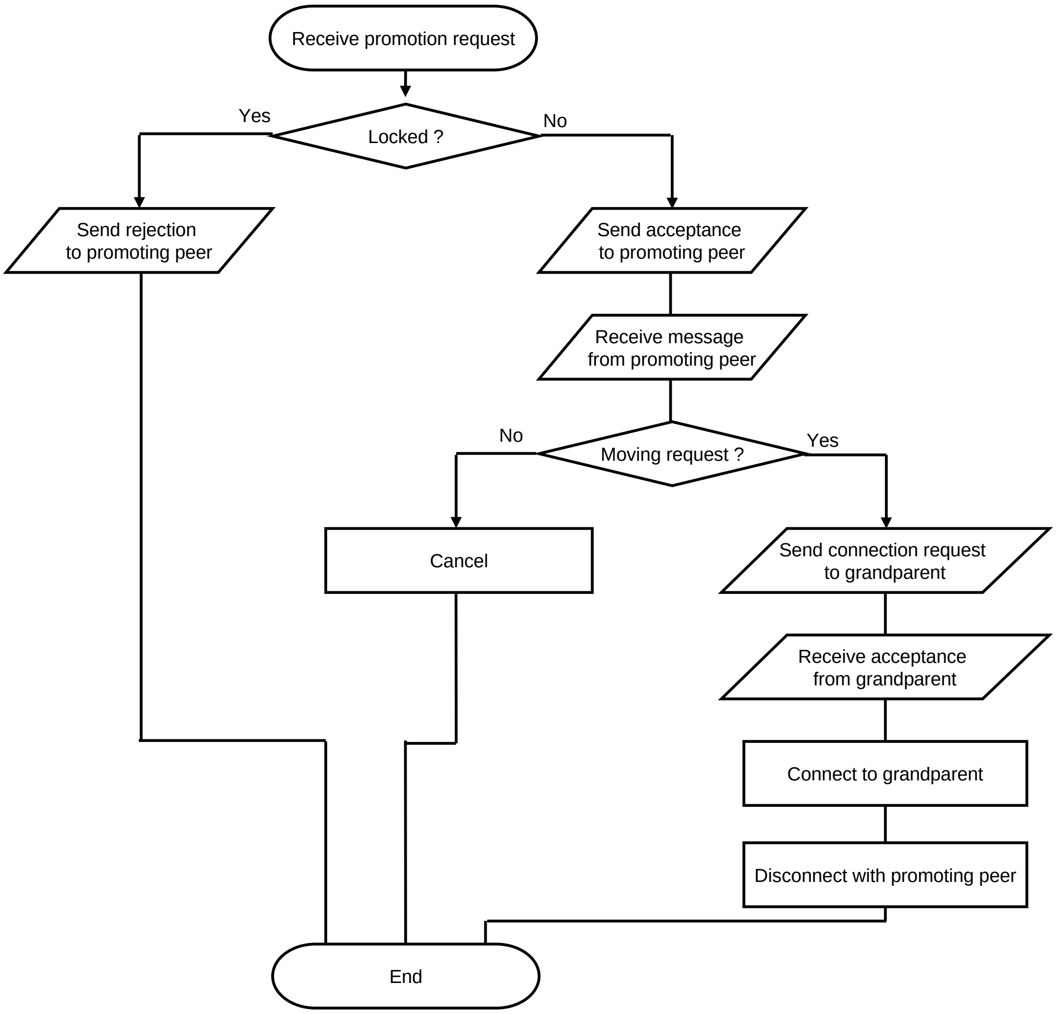


Figure 12: Flowchart: child of promoting peer

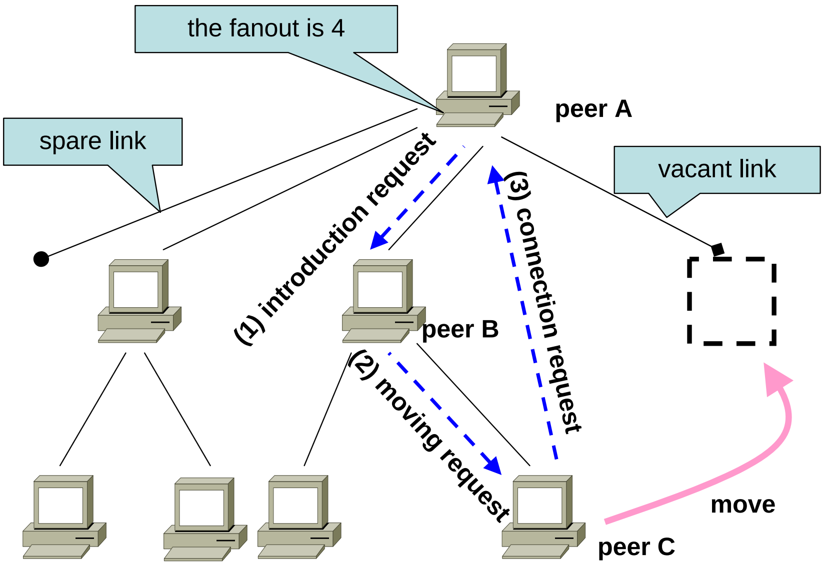


Figure 13: Completing the fanout of peer A

sends an introduction request message to peer B which is selected in a round-robin fashion from its children (Fig. 13:1). 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. 13:2). The moving request includes the IP address of peer A. Peer C sends a connection request to peer A (Fig. 13:3) and makes the connection. Then peer C terminates the connection with peer B and this process is completed. During the process, if either of peer B and C or both are locked, or peer B has no child peer, peer A receives a reject message and the process is canceled.

Figures 14, 15, and 16 show flowcharts of a completing peer and other peers for the fanout completion.

2.4 Failure Recovery Mechanism

A peer may become to be unable to receive streaming data due to not being able of to send a message to 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 [15].

If a peer fails in sending a message to a temporary parent, it sends a re-connection request

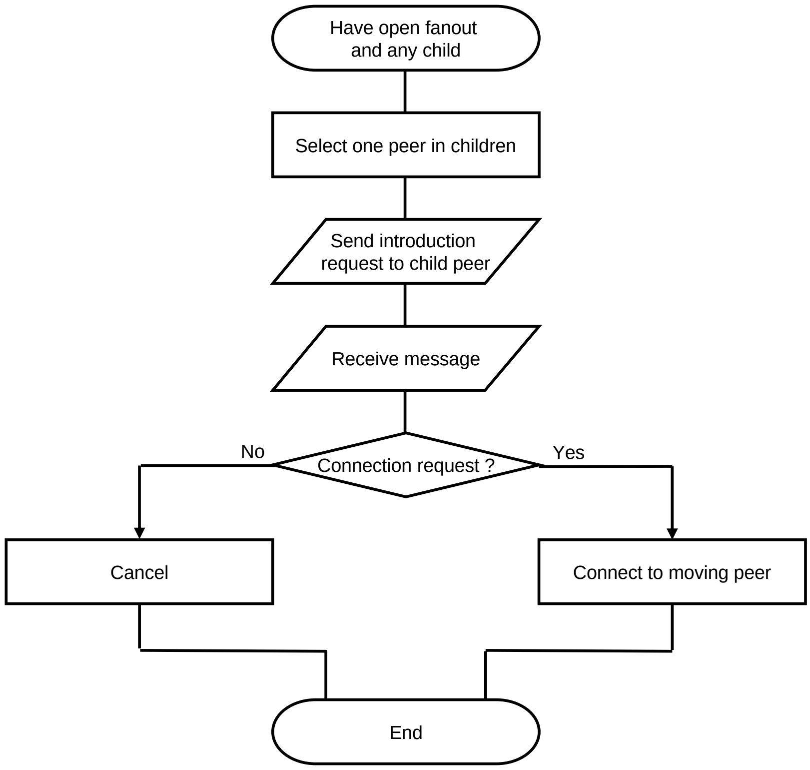


Figure 14: Flowchart: completing peer

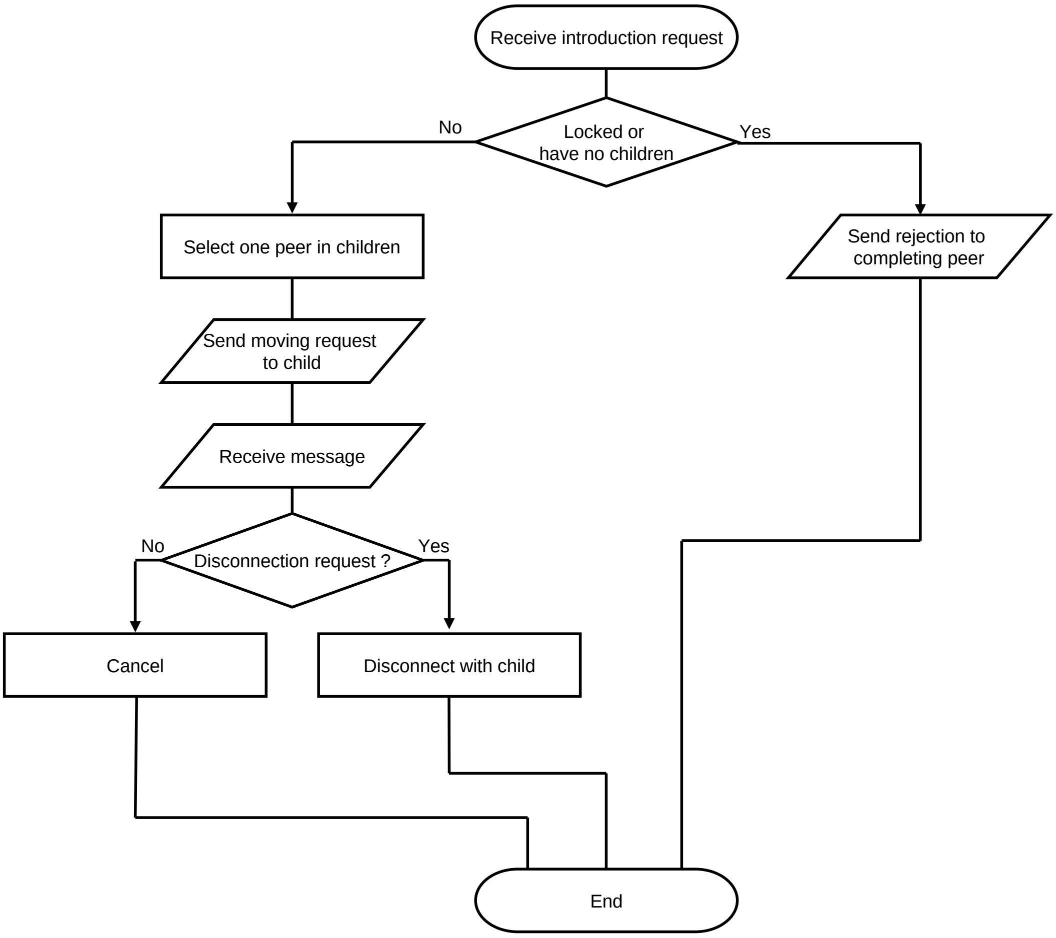


Figure 15: Flowchart: child of completing peer

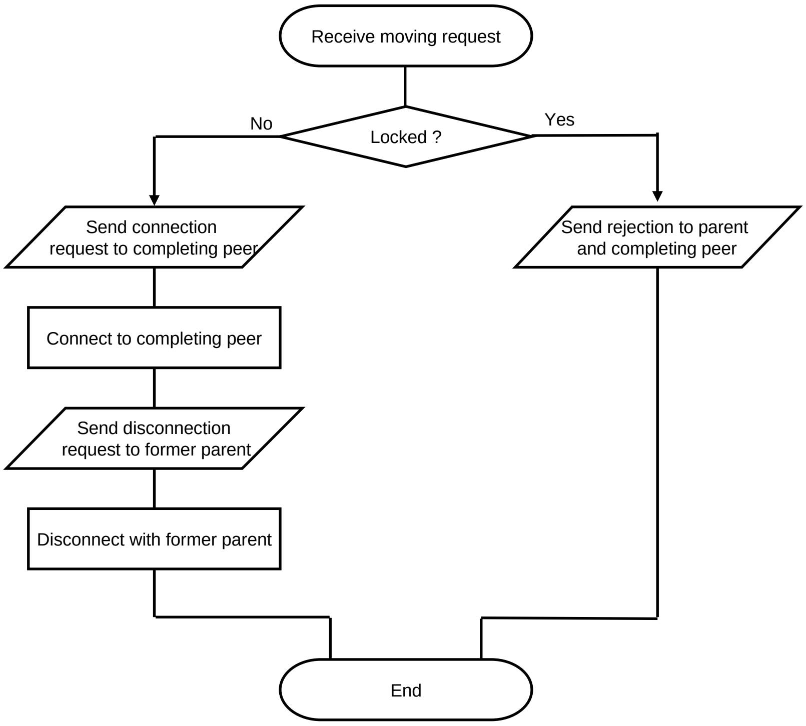


Figure 16: Flowchart: grandchild of completing peer

message to the previous temporary parent, which introduced the missing temporary parent. 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 and send a re-connection request message (Fig. 17:1,3). In both cases, the IP address of the missing parent is removed from the ancestor list. If the recovering peer fails in sending a re-connection request message to a 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 a 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. 17:2), or introduces a child to the recovering peer as a new temporary parent otherwise (Fig. 17:4). In the latter case,

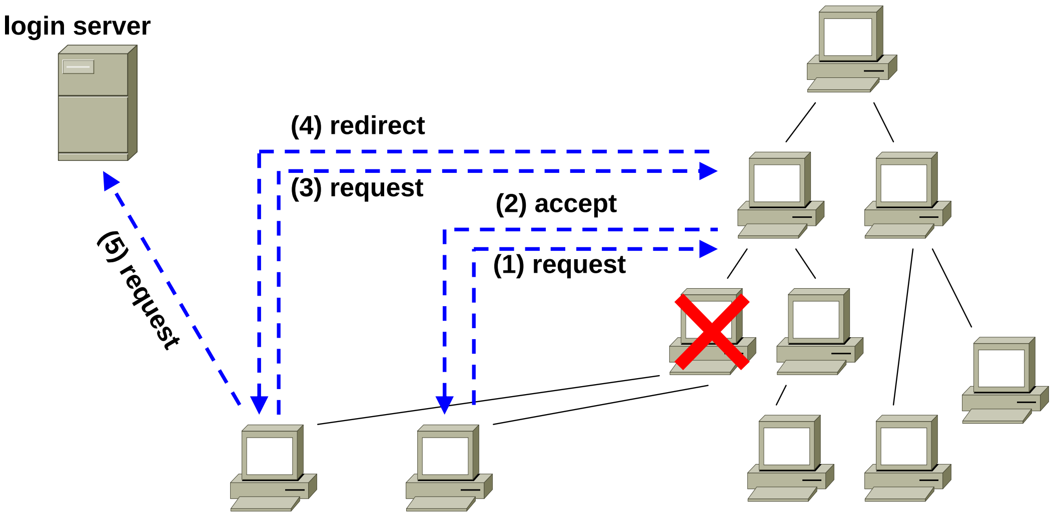


Figure 17: Failure recovery

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. 17:5).

If the failure of the leader peer occurs, 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.

Figures 18 and 19 show flowcharts of a recovering peer and a temporary parent for the failure recovery mechanism.

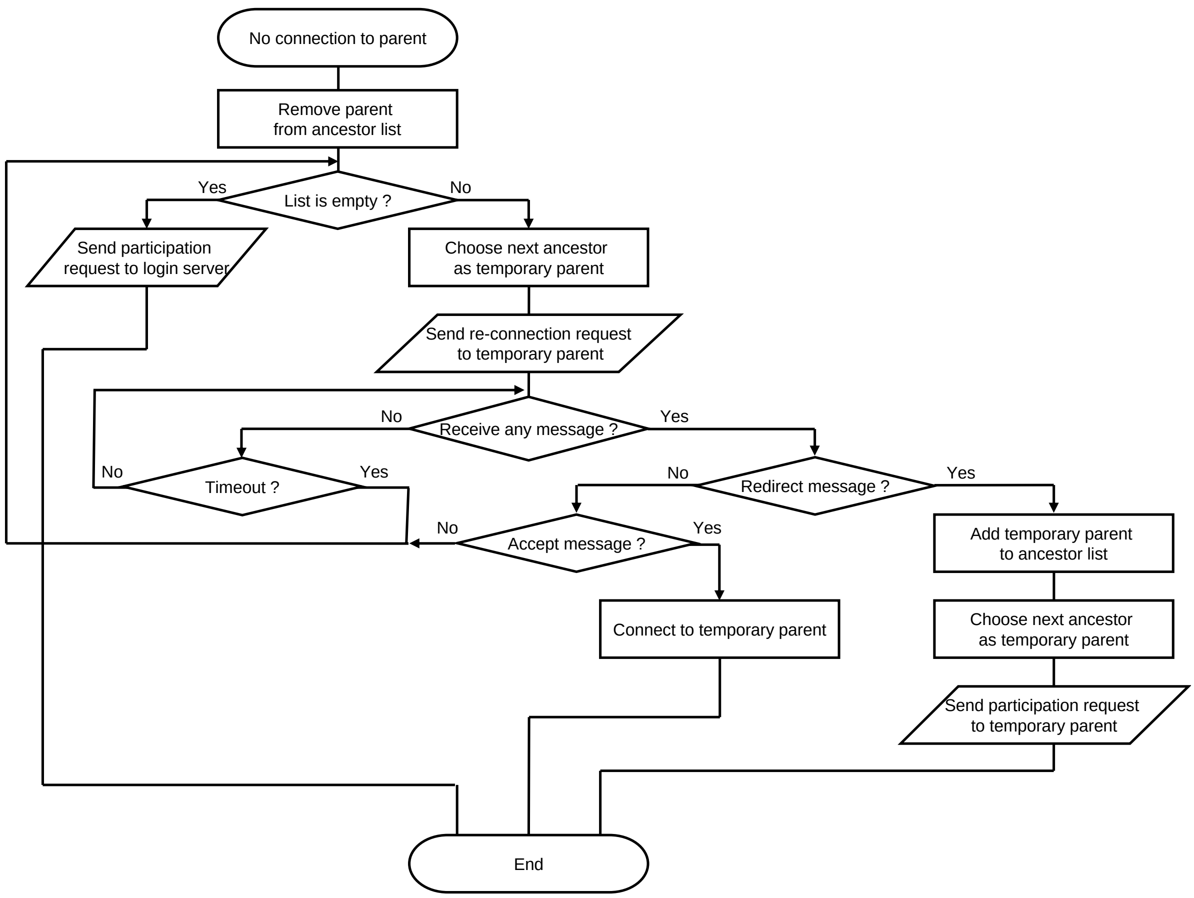


Figure 18: Flowchart: recovering peer

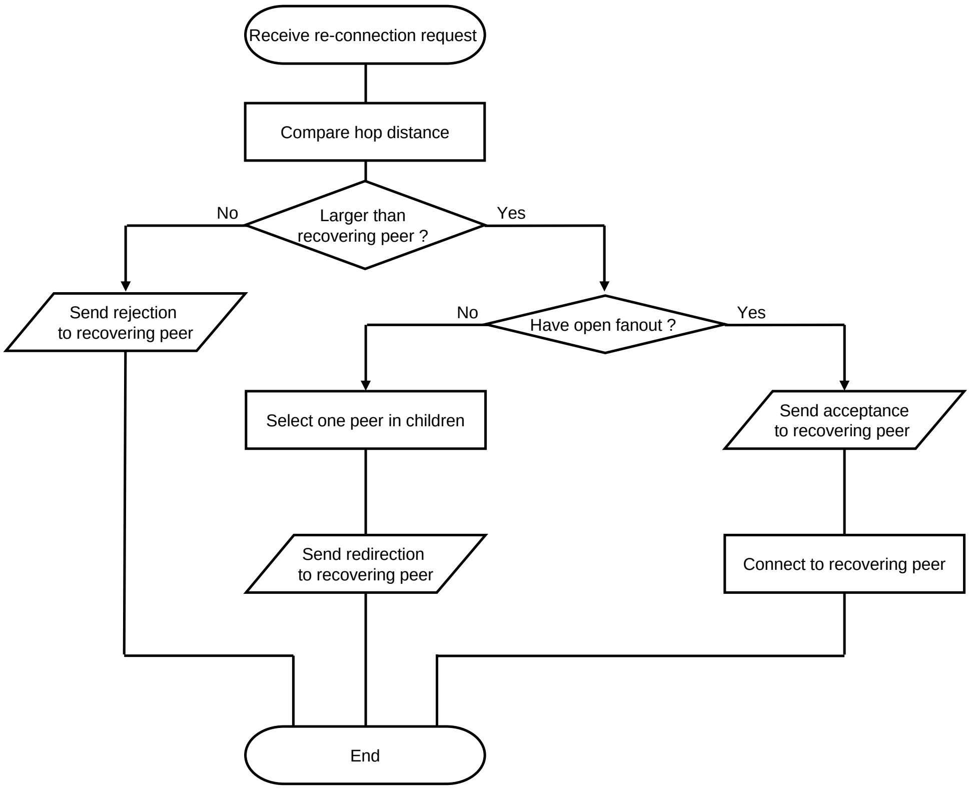


Figure 19: Flowchart: temporary parent of recovering peer

3 Evaluation for Single Distribution Tree

In this section, we evaluate our method from a viewpoint of smoothness of video conferencing. The simulation program for experiments is made by C++ language.

Streaming 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 thesis is mainly in the management of a distribution tree, we firstly focus on the performance and effectiveness of our method in managing a single distribution tree in this section. Evaluation of the whole system constituting both of the core network and several distribution trees will be given in section 4.

3.1 Simulation Conditions

First, a physical network used for a simulation run is generated based on the BA model [16] or the Waxman model [17] using BRITE [18]. The average degree is set at 2. A network consists of 101 or 1001 nodes, i.e., routers. One router is connected with the login server and each of the other routers has one peer which participates in the conference. We define the fanout for a peer with two approaches. In the first approach, we assume that the fanout, i.e., the acceptable number of logical links for conferencing, depends on the capacity of an access link of a peer. The ratio of the type of access link is ADSL : FTTH : CATV$= 6 : 3 : 1$ [19]. We first randomly assign the type of access link to peers keeping the ratio. Then, taking into account their typical capacity, we define the fanout for each type of access link as 2, 7, and 4, respectively. In the second approach, the fanout is set equal to the degree, i.e., the number of neighboring routers, of the designated router plus 1.

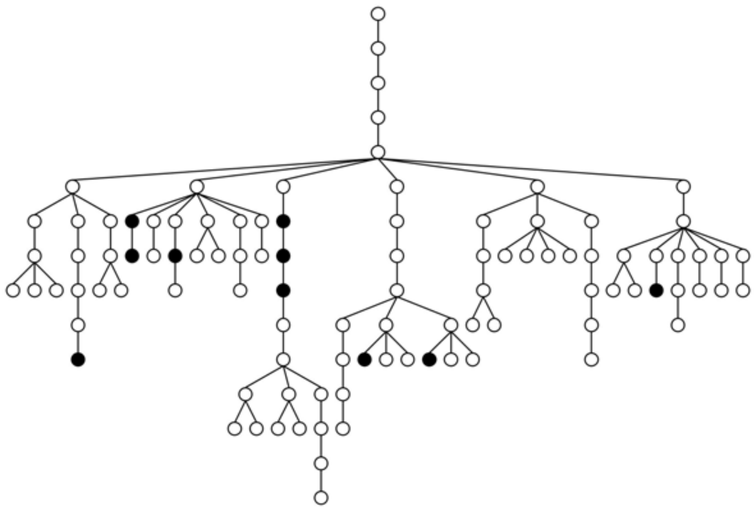
The delay of a logical link between an arbitrary pair of peers is given as the product of the number of hops of the shortest path obtained by the Dijkstra algorithm [20] for the physical network ignoring the access link and the one-hop delay of 1 msec. We do not consider transmission delay and processing delay.

A simulation run consists of two phases, i.e., the initialization and conference. In the initialization phase, peers to participate in the conference emit a participation request to the login server at random time. A distribution tree is gradually constructed as a new peer joins, where the first participating peer becomes a leader peer. When all peers have joined the distribution tree and the tree is completed, the phase moves to the conference. In the conference phase, no further peer joins.

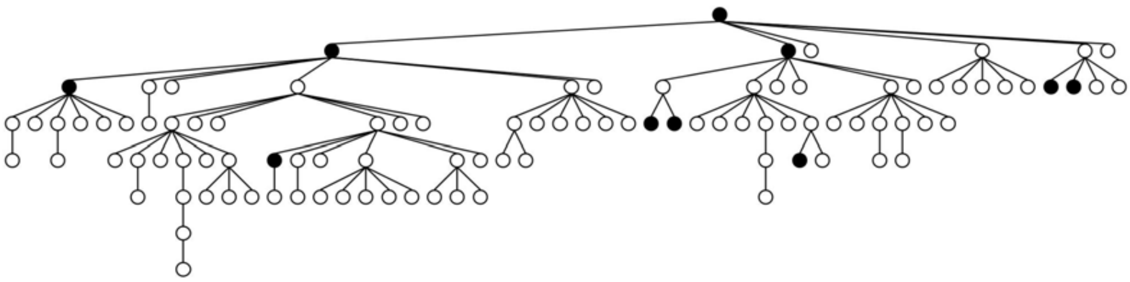
As the initialization phase is finished and the conference phase starts, a peer begins to speak and tree reorganization is conducted. We define peers to speak as *candidates*. Candidates are randomly chosen among participating peers at the beginning of the conference phase and fixed during the simulation run. The duration of speech is exponentially distributed with a mean value of 6 seconds [21] and the minimum duration is 1 msec. Any one of 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 beginning of the conference phase 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 and the peer is locked, 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 promotion among peers, the first comparison at a peer 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 at a peer occurs at random time with uniform distribution from 0 to 7 seconds.

We evaluate our method from the viewpoint of smoothness of the conference. For this purpose, we use the following measures; 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. To have a smooth conference, all of the delay, the number and duration of freezes, and the load must be kept small even if the number of participants increases. In the following figures, we also show results of the case that a distribution tree does not change during a simulation run, 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 for 100 peers or 60 minutes for 1000 peers in simulation time unit after the first speaker begins to speak.



(a) The initial tree topology



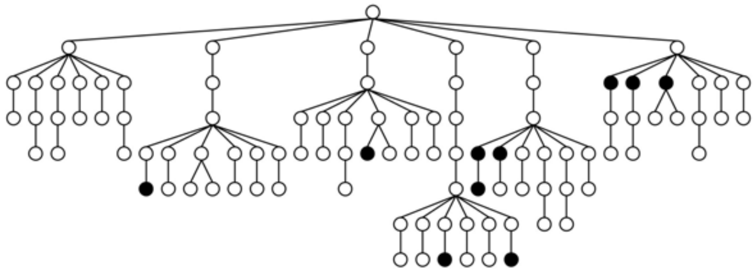
(b) The reorganized tree topology

Figure 20: Result of tree reorganization (BA model, fanout defined by link type, 100 peers, 10 candidates)

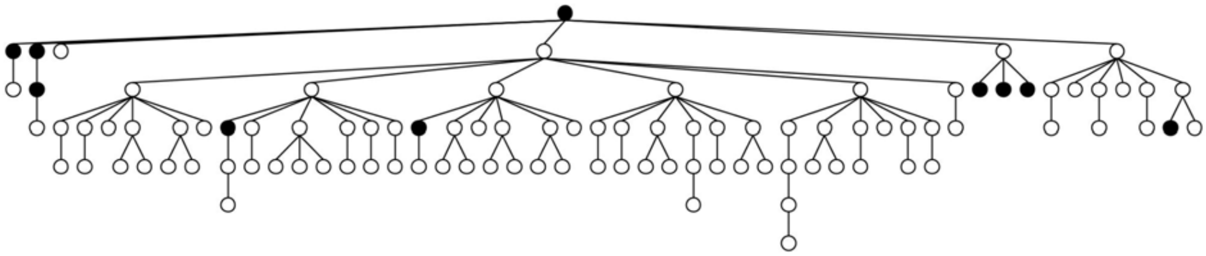
3.2 Results on Tree Topology

First, we evaluate the effectiveness of the tree reorganization mechanism without failures.

In Figs. 20, 21 and 22, how a distribution tree was reorganized in a certain simulation run is illustrated. The model used for Figs. 20 and 22 is the BA model, and the model for Fig. 21 is the Waxman model. The fanout for a peer of Figs. 20 and 21 is defined by the type of access link, and that for Fig. 22 is defined by the degree of the designated router. In Table 1, those simulation conditions are summarized. The number of peers is 100 and that of candidates is 10 in all figures. In these figures, filled circles correspond to candidates and open circles indicate other peers.



(a) The initial tree topology

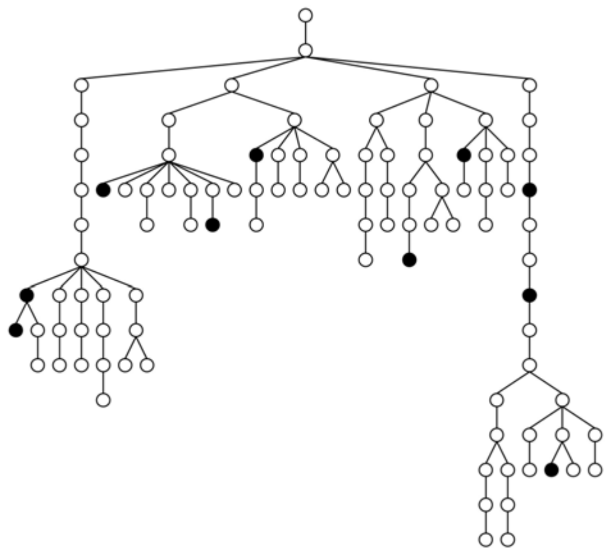


(b) The reorganized tree topology

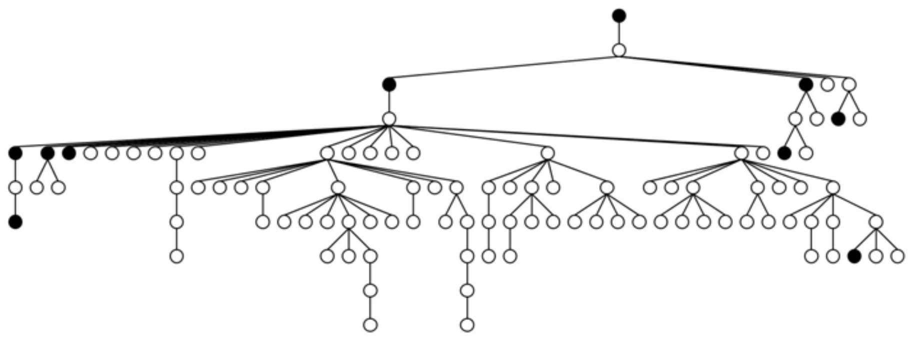
Figure 21: Result of tree reorganization (Waxman model, fanout defined by link type, 100 peers, 10 candidates)

Table 1: Summary of conditions

	Physical network topology	fanout
Condition A	BA model	the type of access link
Condition B	Waxman model	the type of access link
Condition C	BA model	the degree of the designated router



(a) The initial tree topology



(b) The reorganized tree topology

Figure 22: Result of tree reorganization (BA model, fanout defined by router degree, 100 peers, 10 candidates)

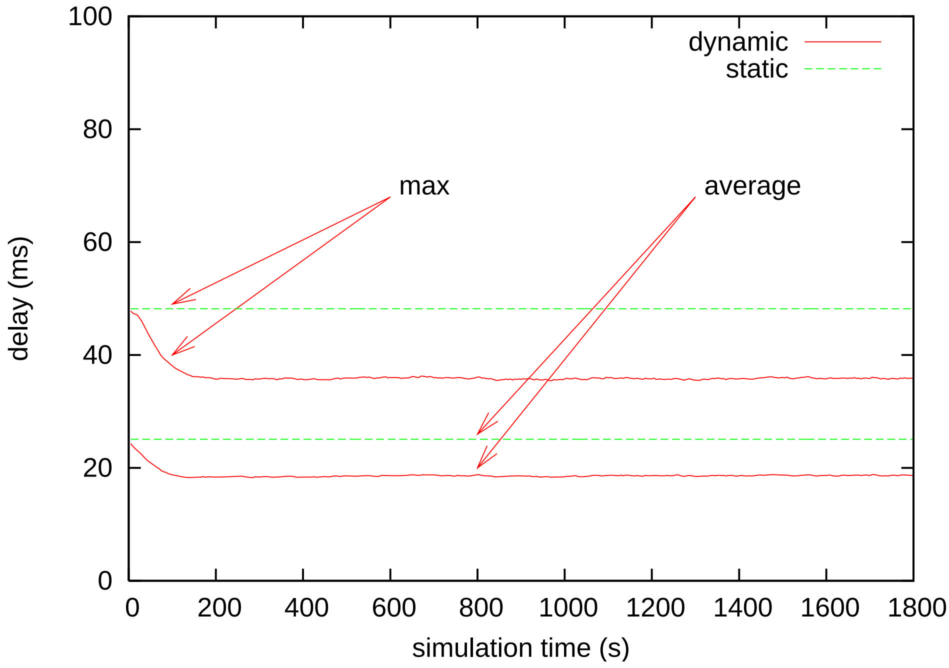
The figures show that tree reorganization mechanism reduces the height of the tree. With 1000 simulation experiments for each condition, the average distance from the leader peer to all peers is reduced from about 5.3 hops to about 4.1 hops under condition A, about 5.3 hops to about 4.2 hops under condition B, and about 7.4 hops to about 4.0 hops under condition C, and the maximum distance changes from about 9.9 hops to 7.5 hops, about 9.9 hops to 7.4 hops, and about 14 hops to 8.2 hops. It can be seen that the initial hop distance of condition C is larger than that of the other conditions. It is because a fanout is relatively small for most of peers with condition C due to the skewed degree distribution of the BA model. For its majority, peers with a small fanout are likely to exist near the root in the initial tree. However, as a result of reorganization, the average hop distance is almost the same among conditions in reorganized trees. This explains that our tree reorganization mechanism is effective in reducing the height of tree independently of the topology of physical network.

Furthermore, we can see that candidates have moved near the root of the tree by reorganization. With 1000 simulation experiments of each condition, the average distance from the leader peer to candidates decreases from about 5.6 hops to about 2.8 hops under condition A, about 5.6 hops to about 2.8 hops under condition B, and about 7.5 hops to about 3.1 hops under condition C, and the maximum distance changes from about 8.4 hops to 5.4 hops, about 8.4 hops to 5.5 hops, and about 11 hops to 5.5 hops. Although all candidates are not necessarily located near the root depending on the timing and duration of speaking, moving active peers towards the root helps in having a smooth conference among interactive participants, as will be shown in the next subsection.

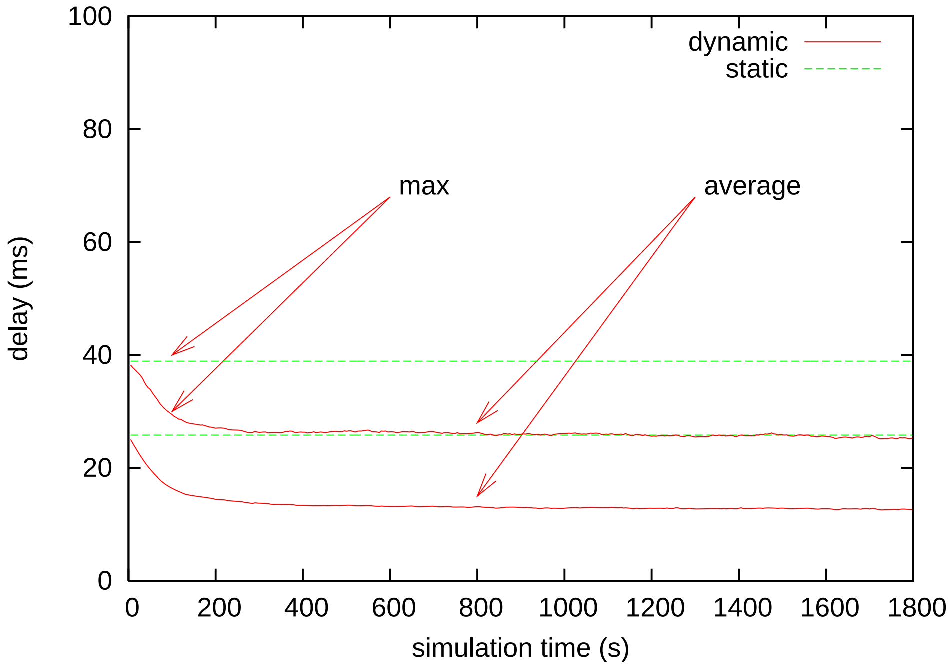
3.3 Results on Delay

Figures 23(a), 24(a), and 25(a) illustrate how the average and maximum delay between the leader peer and all peers change under conditions A, B, and C, respectively. Independently of conditions, the delay between the leader peer and all peers is successfully decreased as can be seen by comparing results of dynamic to those of static. Among promotions, those invoked by fanout comparison and completion mainly contribute to the initial reduction of delay.

Figures 23(b), 24(b), and 25(b) show the average and maximum delay between the leader peer and candidates. When comparing to Fig. 23(a), 24(a), and 25(a), the delay for candidates is less than that for all peers. It means that speakers have better and smoother conversation. We should note here that the delay for candidates remains constant after the initial reduction. This is because

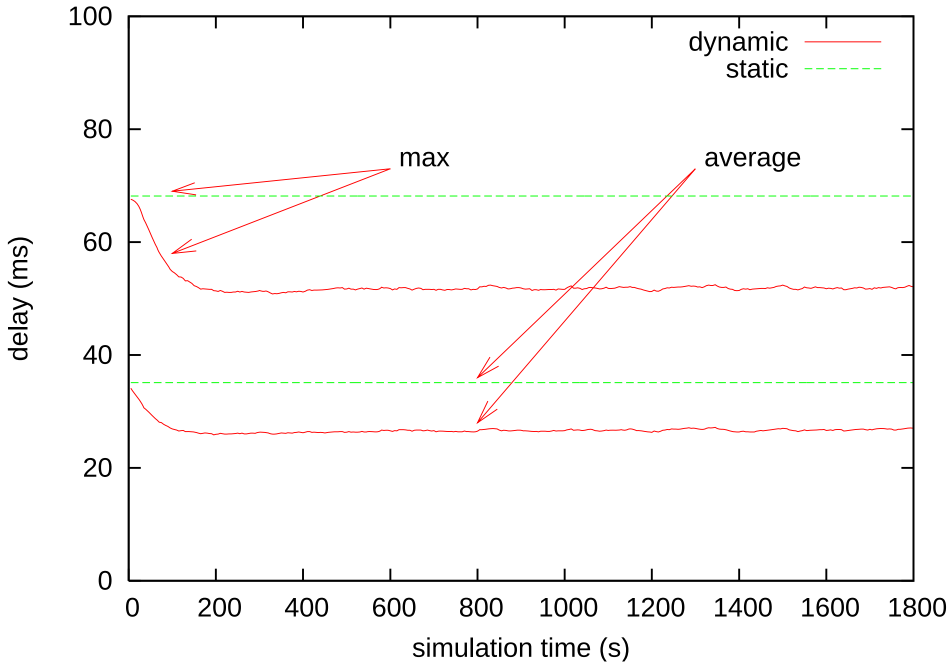


(a) All peers

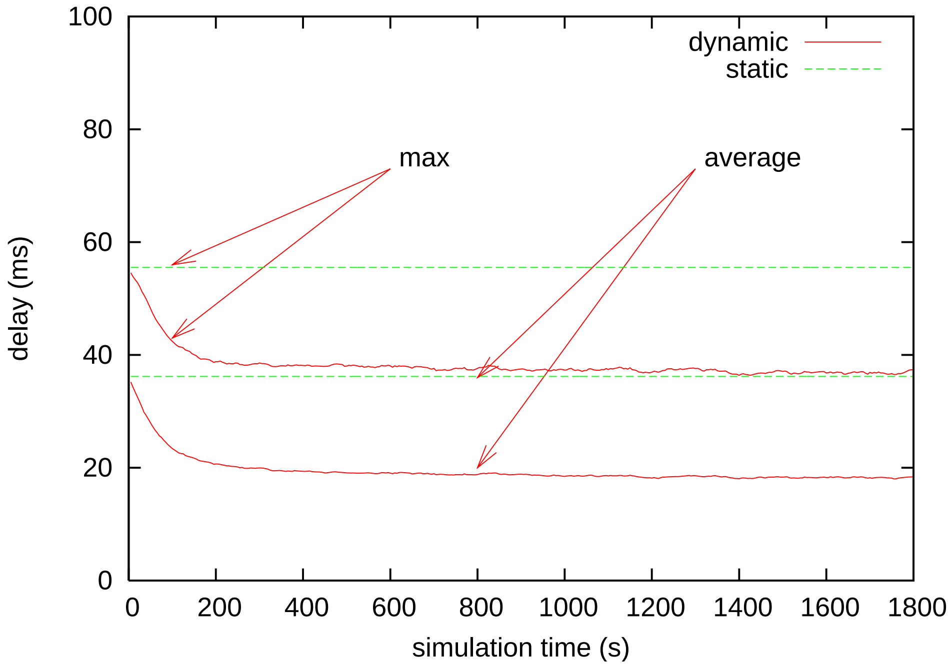


(b) Candidates

Figure 23: Delay between the leader peer and peers (BA model, fanout defined by link type, 100 peers, 10 candidates)

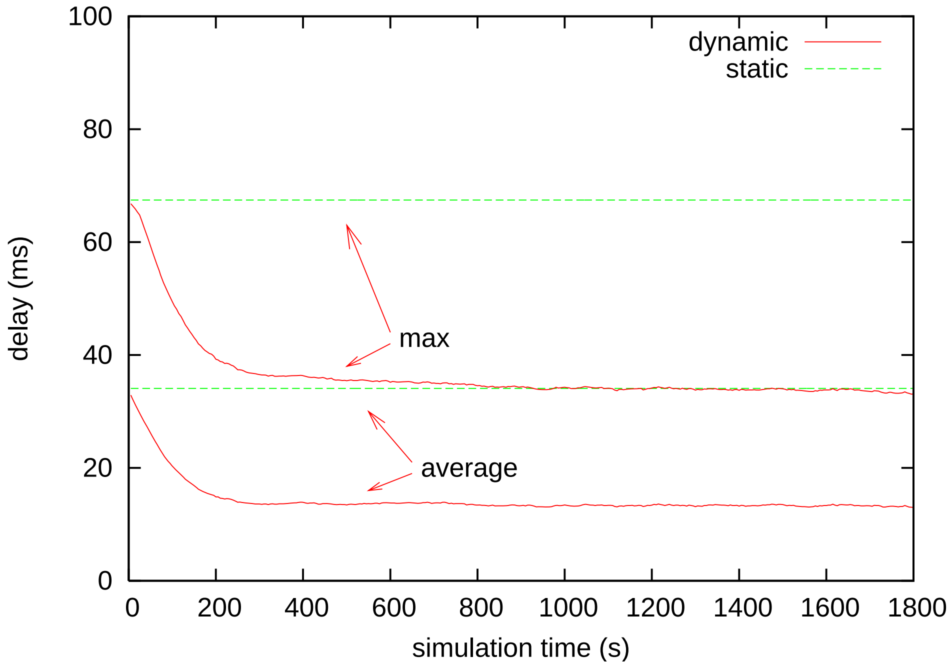


(a) All peers

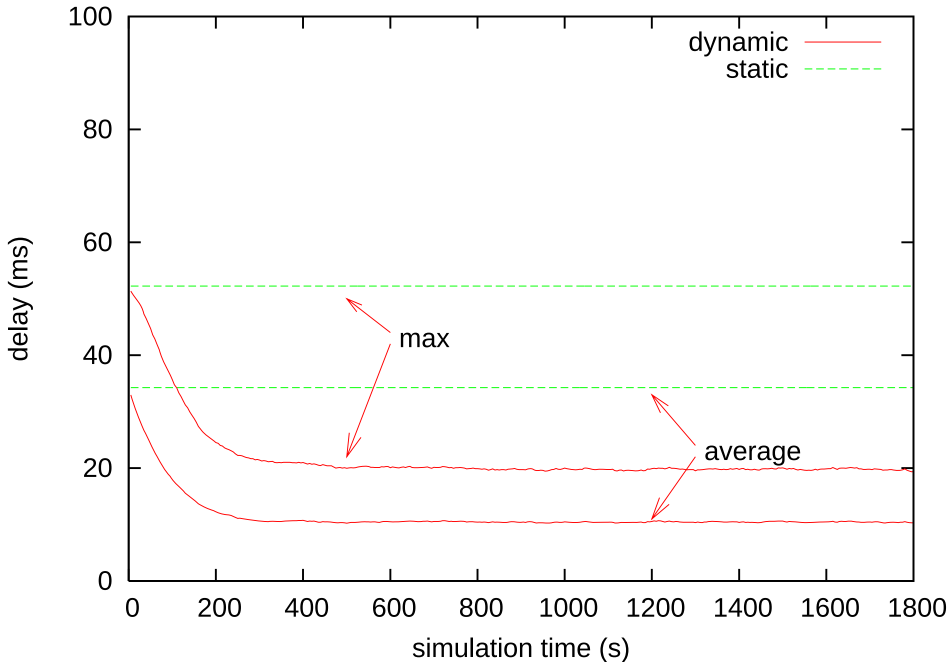


(b) Candidates

Figure 24: Delay between the leader peer and peers (Waxman model, fanout defined by link type, 100 peers, 10 candidates)



(a) All peers



(b) Candidates

Figure 25: Delay between the leader peer and peers (BA model, fanout defined by router degree, 100 peers, 10 candidates)

that the most of speakers have moved near the root during the initial reduction.

When the maximum delay between the leader peer and all peers in a distribution tree is D, the maximum delay between the leader peer and candidates is C, and that between leader peers in the core network is L, the maximum end-to-end delay for distributing streaming data from any candidate to all peers can be derived as $D + C + L$. By reorganization, D is decreased to about 50 msecs and C becomes about 35 msecs in Fig. 24. Therefore, if we can construct a core network in which L is less than 15 msecs, we can offer video conferencing with the end-to-end delay less than 100 msecs from candidates to all peers, which is smaller than the recommended one-way delay for voice communication [22].

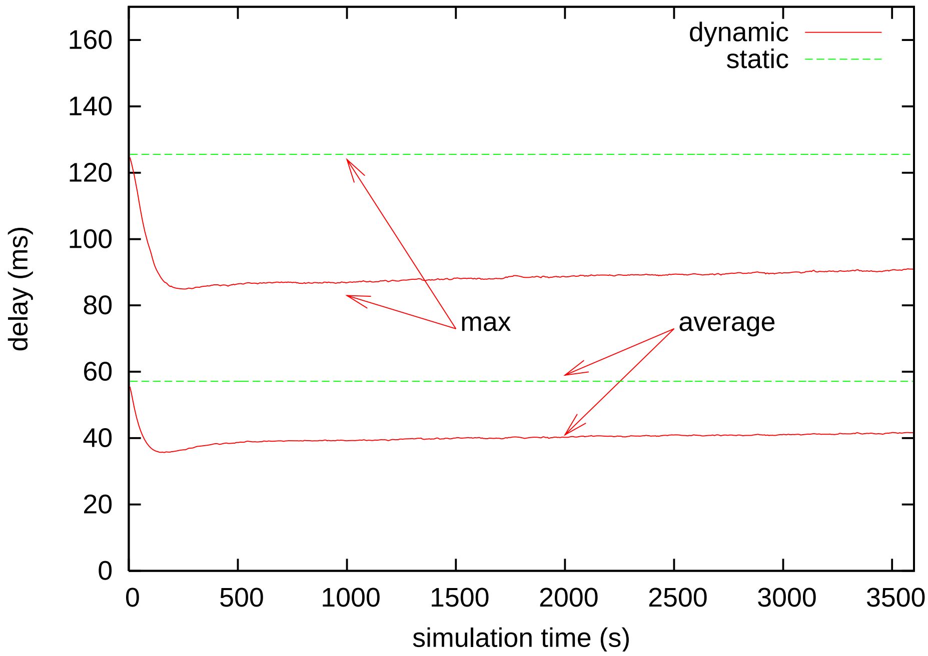
Among Figs. 23, 24, and 25, the initial delay of Figs. 24 and 25 is larger than Fig. 23. It is because the physical hop distance among peers under condition B becomes high due to the Waxman model and the tree height under condition C becomes high as explained in subsection 3.2.

From these figures, we can say that our method is effective for a distribution tree of 100 peers under any of three conditions. In the following, we only show results under condition A, where a physical network topology is generated by the BA model and the fanout of a peer is defined with the type of access link.

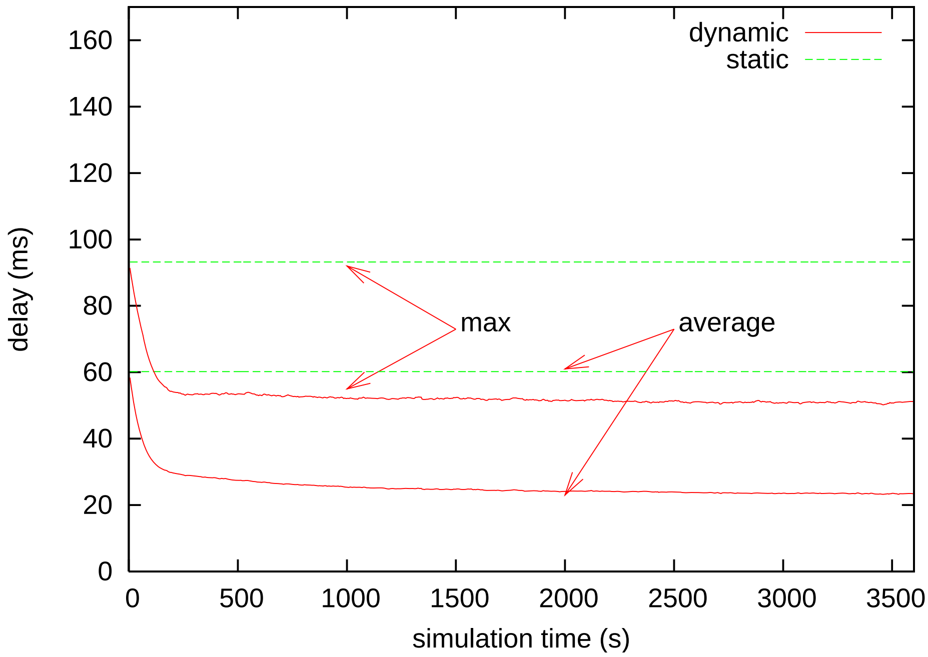
To see the scalability of our method, the number of peers in a tree is increased from 100 to 1000. Results are shown in Fig. 26 for the average and maximum delay between the leader peer and all peers and Fig. 27 for the standard deviation. The number of candidates is kept at 10. Results for the case of 100 candidates are shown in Figs. 28 and 29.

Figure 26 shows that the delay is effectively reduced by reorganization. However, the absolute value of the delay is higher than the case with 100 peers for the increased number of peers. The maximum delay from candidates to all peers is about $C + D + L = 50 + 90 + L$ msecs and larger than 100 msecs. This delay is too large to have interactive communication between a candidate and the most distant peer. However, when we focus on candidates, who mainly speak and communicate, the end-to-end delay is derived as $C \times 2 + L$, which is about $100 + L$ msecs at maximum and $50 + L$ msecs on average. Therefore, if we can construct a core network with sufficiently low delay, we can offer smooth video conferencing among actively speaking participants.

As shown in Fig. 27, reorganization leads to a more balanced tree. The reason why the standard deviation gradually increases as time passes is promotion of speaking peer. The initial decrease of the delay is mainly due to fanout-related reorganization. As a candidate speaks, it moves towards



(a) All peers



(b) Candidates

Figure 26: Delay between the leader peer and peers (BA model, fanout defined by link type, 1000 peers, 10 candidates)

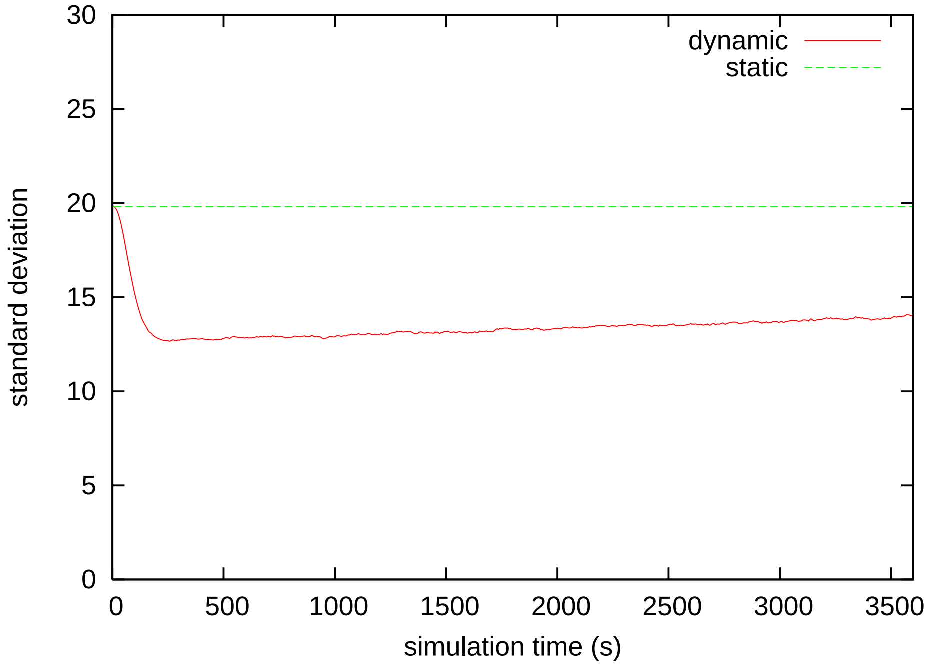
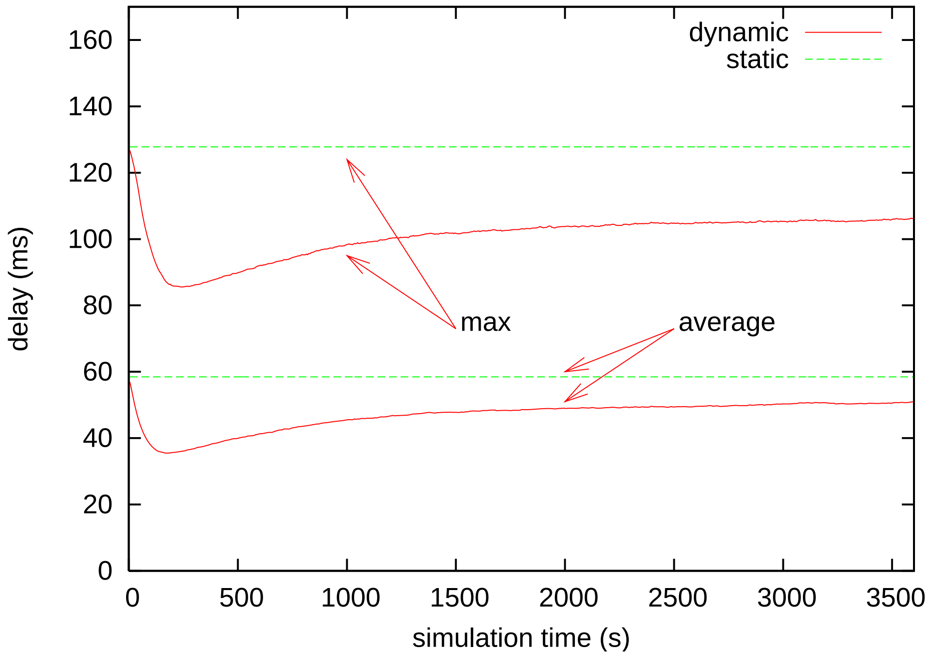


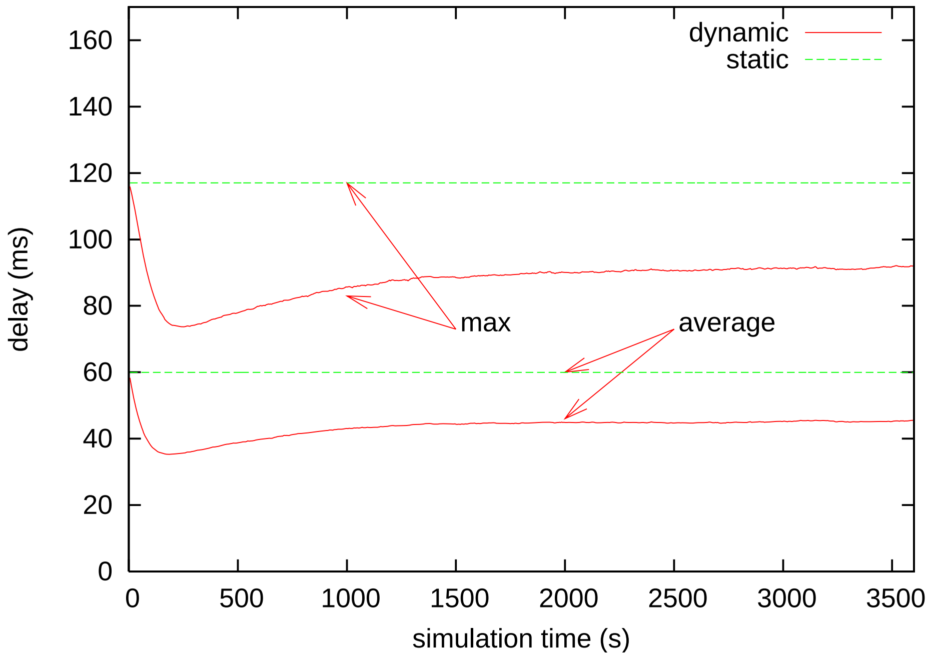
Figure 27: Standard deviation of delay between the leader peer and all peers (BA model, fanout defined by link type, 1000 peers, 10 candidates)

the root. Since a candidate does not necessarily have a large fanout, the promotion would make a tree higher and unbalanced. This also explains the slight increase in the delay between the leader peer and all peers in Fig. 26(a).

As the number of candidates increases, our method becomes less effective as shown in Fig. 28 where 100 peers among 1000 peers are candidates. Due to the limited fanout of a peer, not all of candidates can be located near the root. In addition, a candidate has less chance to speak and move towards the root. We also see that the delay once decreases first but gradually increases afterward. As shown in Fig. 29, a tree topology is getting unbalanced as time passes, i.e., as candidates speak. This is for the same reason explained in the previous paragraph, but the influence becomes more significant with more candidates. Therefore, we consider that the appropriate number of candidates for one tree is about tens to have a smooth video conference.



(a) All peers



(b) Candidates

Figure 28: Delay between the leader peer and peers (BA model, fanout defined by link type, 1000 peers, 100 candidates)

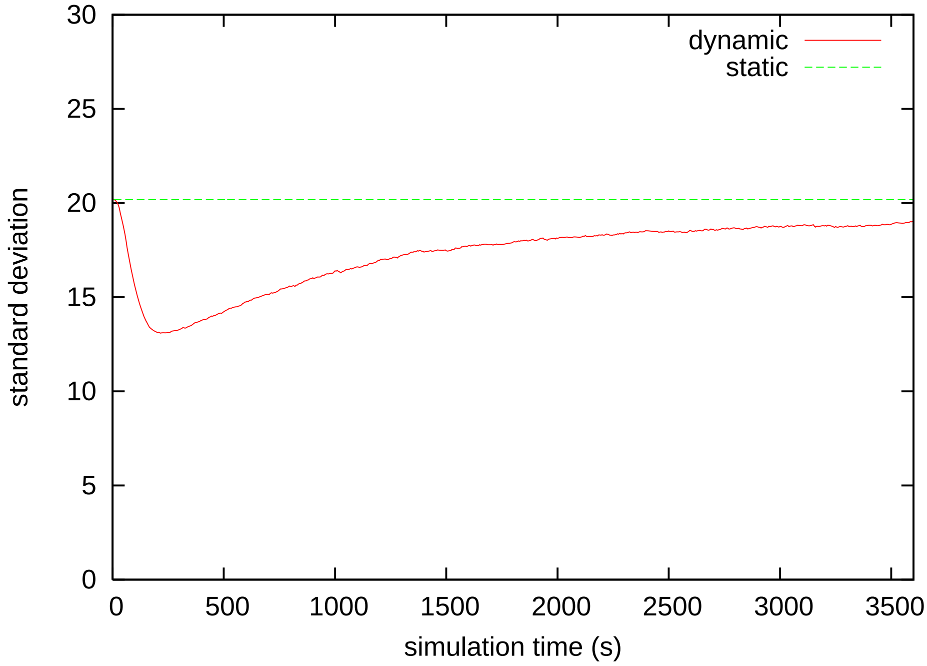


Figure 29: Standard deviation of delay between the leader peer and all peers (BA model, fanout defined by link type, 1000 peers, 100 candidates)

3.4 Results on Load

The average and maximum number of messages that passed through a link during a 60-minutes conference and the initialization phase is 1189 and 79680 for the case of 1000 peers and 10 candidates, on average. On the other hand, they are 1156 and 76690 for the case of 1000 peers and 100 candidates. The reason that the load for the case of 10 candidates is larger than that for the case of 100 candidates can be explained as follows. When the number of candidates is small, they can be located near the root. Once a candidate speaks, it moves towards the root by sometimes overtaking a parent peer with a larger fanout than the promoting candidate. As the candidate stops speaking, the overtaken peer again tries moving towards the root by overtaking the promoted candidate. Since it has many children, the promotion involves much message exchange. Thus, the load becomes high. On the other hand, when the number of candidates is large, not all candidates can effectively move towards the root and some of them stay at the bottom. Although such a candidate would overtake a parent by promotion, the fanout of the overtaken peer is small and the next promotion for fanout comparison does not involve much overhead.

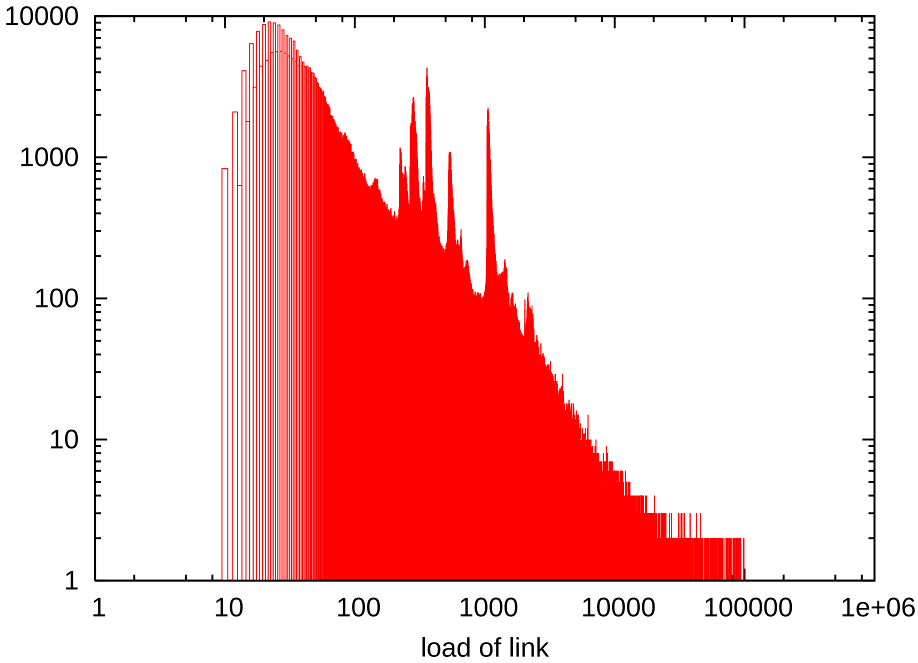


Figure 30: Frequency distribution of the load on link (BA model, fanout defined by link type, 1000 peers, 10 candidates)

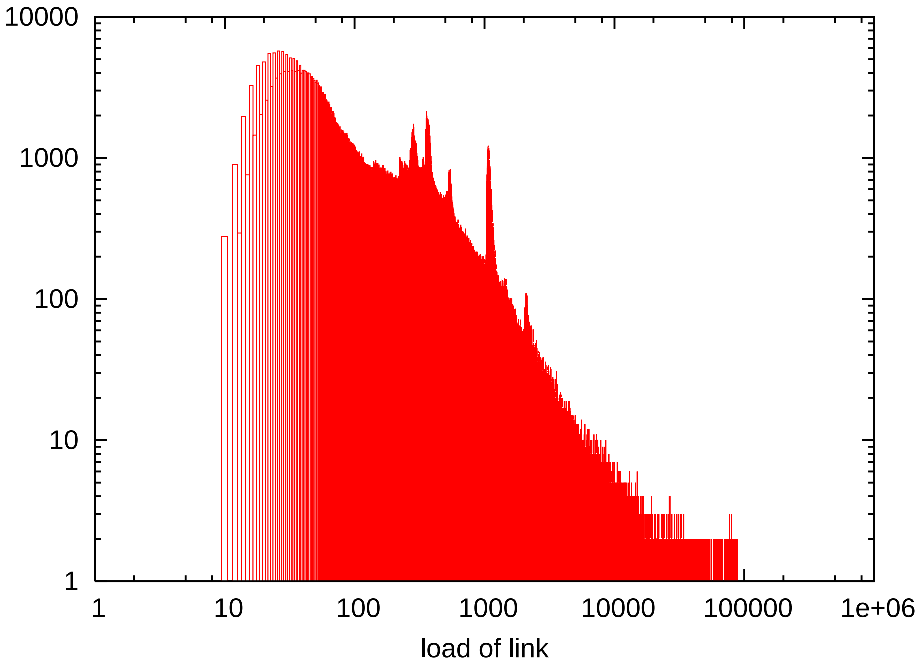


Figure 31: Frequency distribution of the load on link (BA model, fanout defined by link type, 1000 peers, 100 candidates)

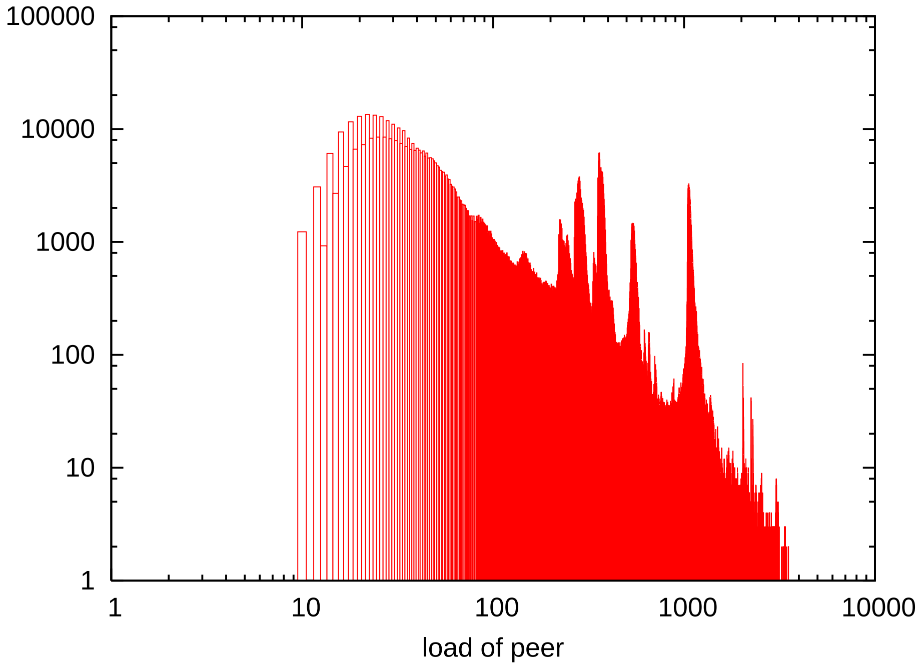


Figure 32: Frequency distribution of the load on peer (BA model, fanout defined by link type, 1000 peers, 10 candidates)

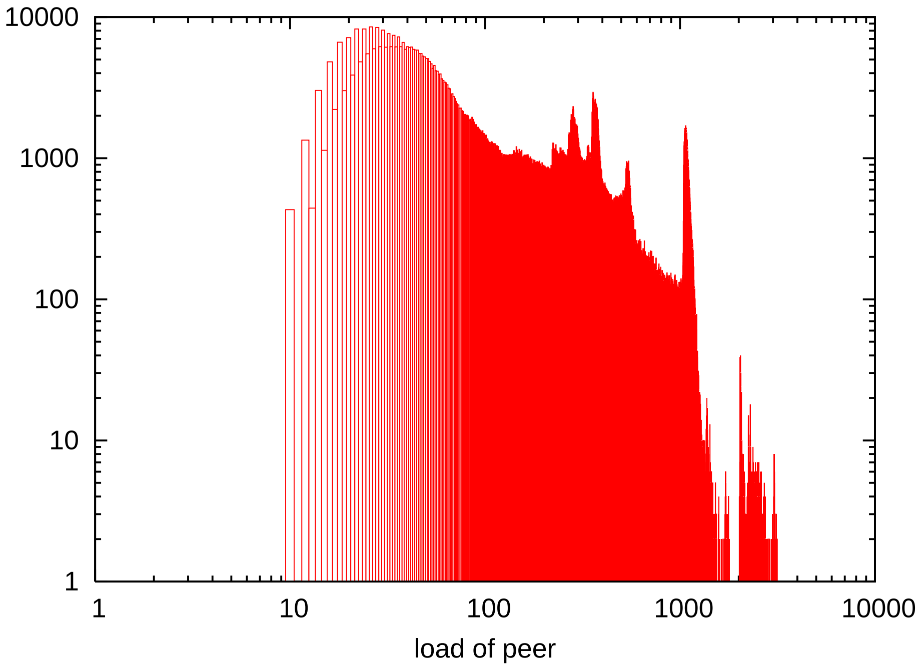


Figure 33: Frequency distribution of the load on peer (BA model, fanout defined by link type, 1000 peers, 100 candidates)

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 87 and 85 bps on average and 5840 and 5620 bps at maximum for a case of 10 and 100 candidates, respectively. This is considerably smaller than the typical rate of streaming data, i.e., from 64 Kbps to 8 Mbps. Figures 30 and 31 illustrate the frequency distribution of load on link in all simulation runs. Independently of the number of candidates, they show the same tendency. The skewed distribution is due to the different location of link.

Figures 32 and 33 illustrate the frequency distribution of load on peers, i.e., the number of messages that a peer sends and receives in all simulation runs. The figures are similar to each other and they are also similar to Figs. 30 and 31, where the load on the most of peers low and only small sets experienced relatively high load. The number of messages that a peer handles is 358 and 349 on average and 2790 and 2580 at maximum for the case of 10 and 100 candidates among 1000 peers, respectively. It consumes 26 and 26 bps on average and 204 and 189 bps at maximum. The load is small compared with the capacity of access link and most of commercially available client systems can afford this. At the worst case scenario with 10000 peers, the maximum load on link is about 60 Kbps and that on peer is about 2 Kbps assuming that the load is proportional to the number of peers. They are still small enough and thus our system can be considered scalable to the number of peers.

Then, we consider the fairness among peers by observing the relationship among the fanout and the load in Figs. 34 and 35. In the figures, error bars show the range of values, i.e., the maximum and mininum, and a solid line connecting error bars stands for the average. As shown, the load is almost independent of the fanout. For example, if a peer with a small fanout joins the conference at the beginning of the initialization phase, it is often appointed as a temporary parent and overtaken by many peers. Consequently, it would handle more messages than a peer with a large fanout. There is a tendency that a peer with larger fanout handles 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. In conclusion, the load depends on how often and how much a peer is involved in tree construction and reorganization and it is independent on the fanout.

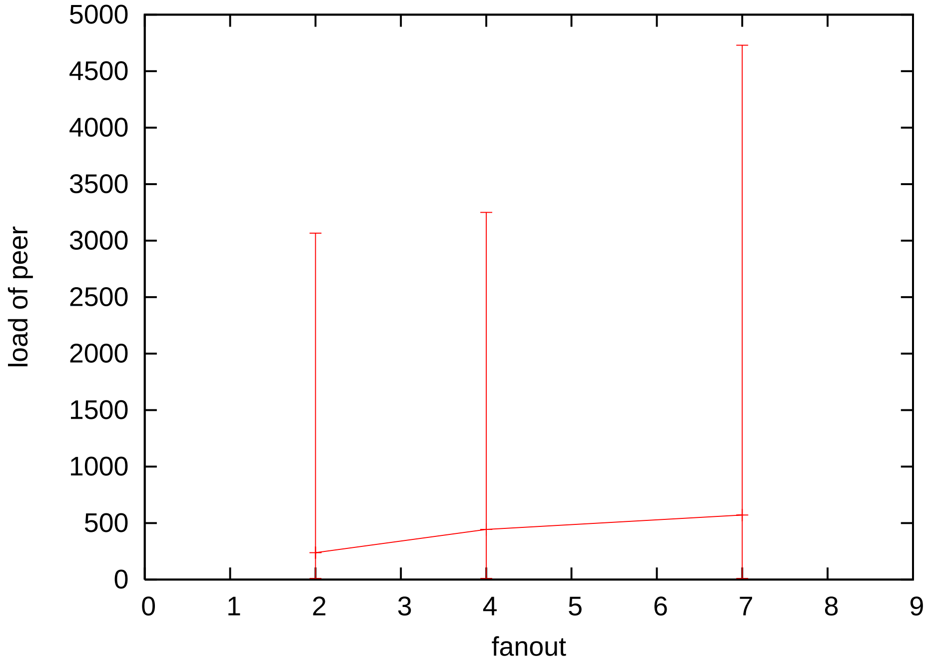


Figure 34: Relationship between the load and fanout (BA model, fanout defined by link type, 1000 peers, 10 candidates)

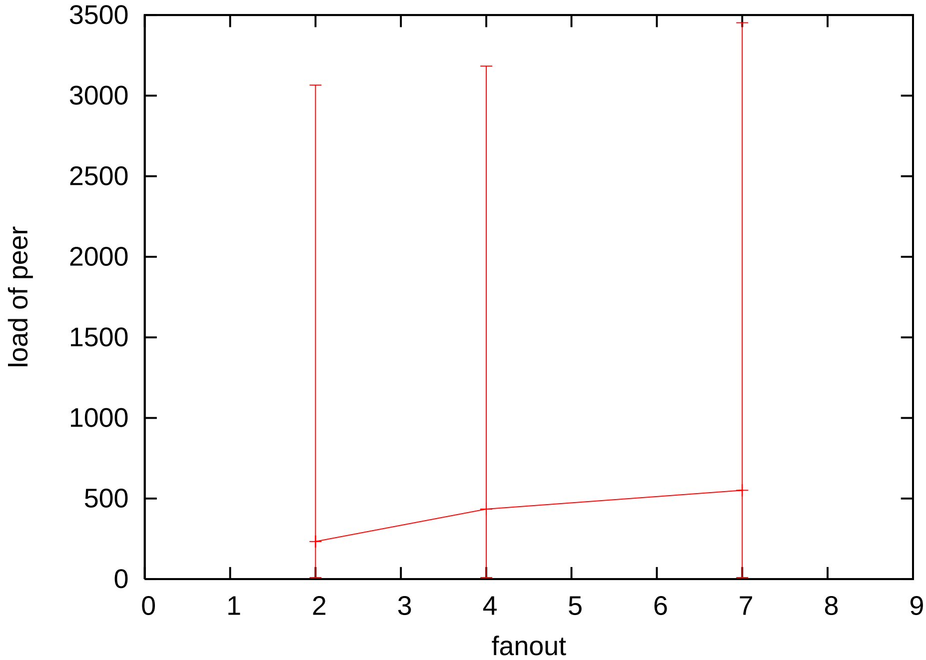


Figure 35: Relationship between the load and fanout (BA model, fanout defined by link type, 1000 peers, 100 candidates)

Table 2: Summary of freezes (BA model, fanout defined by link type, 100 peers, 10 candidates)

		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

3.5 Results on Freeze

In this subsection, we consider the case with failures under the condition with 100 peers, 10 candidates, and 30 simulation time unit. When a peer leaves a conference for some reasons, its descendants cannot receive streaming data and they cannot participate in the conference. Therefore, a failure must be recovered as fast as possible. 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 these problems as *freeze*. In the simulation experiments, 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.

Table 2 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, but the most of freezes last only for several tens of msecs. 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 of freeze is smaller than that of all peers. The reason for long freezes is that some peers wait for the timeout of 3 seconds to detect the disappearance of a temporary parent peer. Among 1000 simulation experiments, the maximum number of freeze is more than one and a freeze last for as long as 6 seconds at maximum. However, this seldom occurs.

4 Evaluation for Distribution Network with Multiple Distribution Trees

In this section, we evaluate our method on the whole system constituting of a core network and several distribution trees.

4.1 Simulation Conditions

A physical network of 1001 peers is generated by the BA model with the average degree of 2. The fanout of a peer is defined with the type of access link. At the initialization phase, the distribution network is gradually constructed. A core network consists of 10 leader peers. By mediation of the login server, the core network has the topology illustrated in Fig. 36, where arrows indicate the direction that streaming data flow.

Since each leader peer is connected with two leader peers, it has to accommodate one direct child at least, and it keeps one spare link, only peers with the fanout of three or more can become a leader peer. Please note that a peer has one more link for a parent. Therefore, in total, a peer with $f = 3$ can connect with three peers while keeping a spare. In the initialization phase, first ten peers with $f >= 3$ are appointed as leaders. When all peers have joined the distribution network and the network is completed, a conference starts.

Following results show averaged values over 1000 simulation experiments, each of which lasts for 60 minutes in simulation time unit after the first speaker begins to speak.

4.2 Results on Delay

Figure 37 illustrates how the delay from ten candidates to all peers and among candidates changes. In the figure, the delay includes that of the core network. The figure shows that both of average and maximum delay are effectively reduced by reorganization. When comparing to the result for the same number of peers and candidates with single distribution tree in Fig. 26 in subsection 3.3, the delay for distributing streaming data over the distribution network becomes smaller by constructing a hierarchical network. In Fig. 26, where all 1000 peers are accommodated in a single tree, the maximum end-to-end delay from candidates to all other peers is estimated as 142 msecs (51 msecs from candidates to the leader and 91 msecs from the leader to all peers). On the other hand, the maximum delay is decreased to about 117 msecs in Fig. 37(a). In addition, the delay among

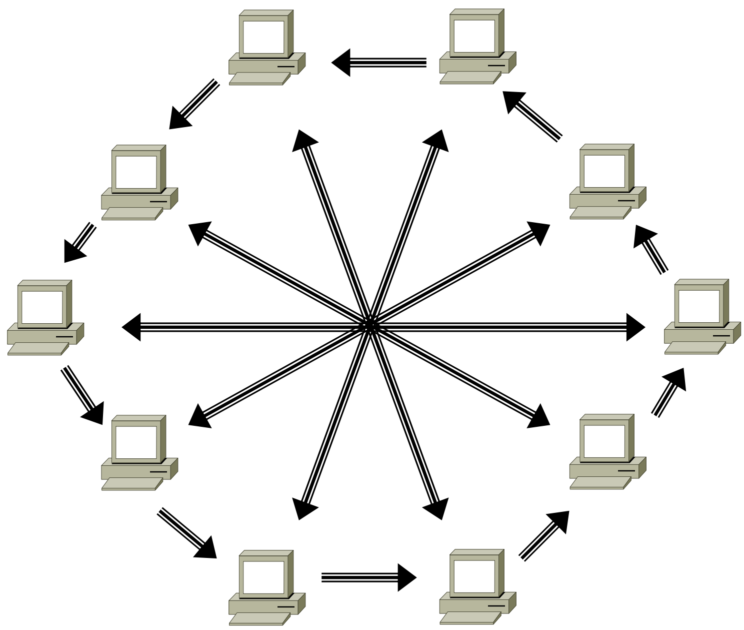
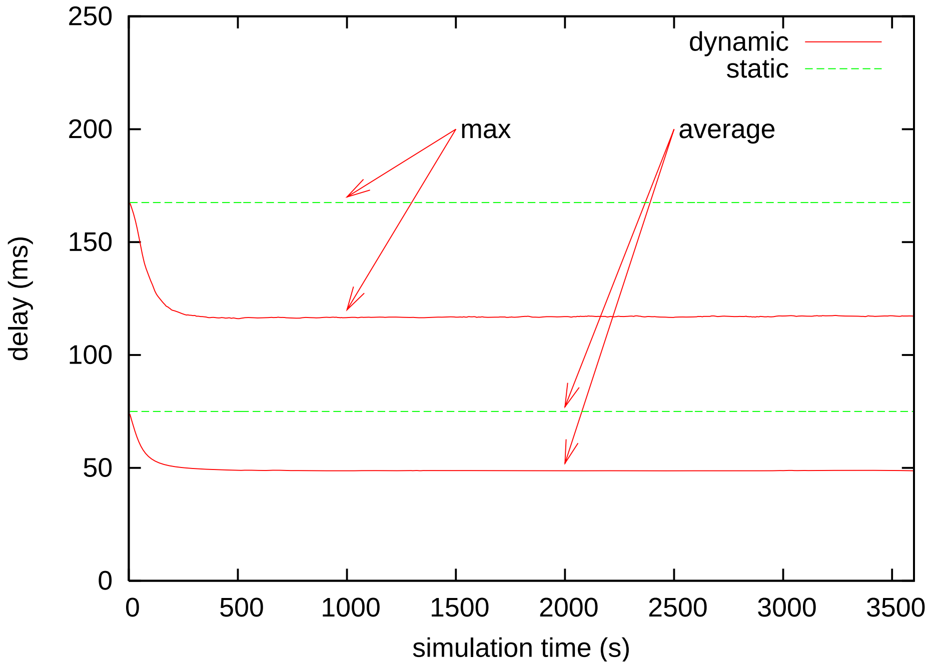


Figure 36: Core network topology

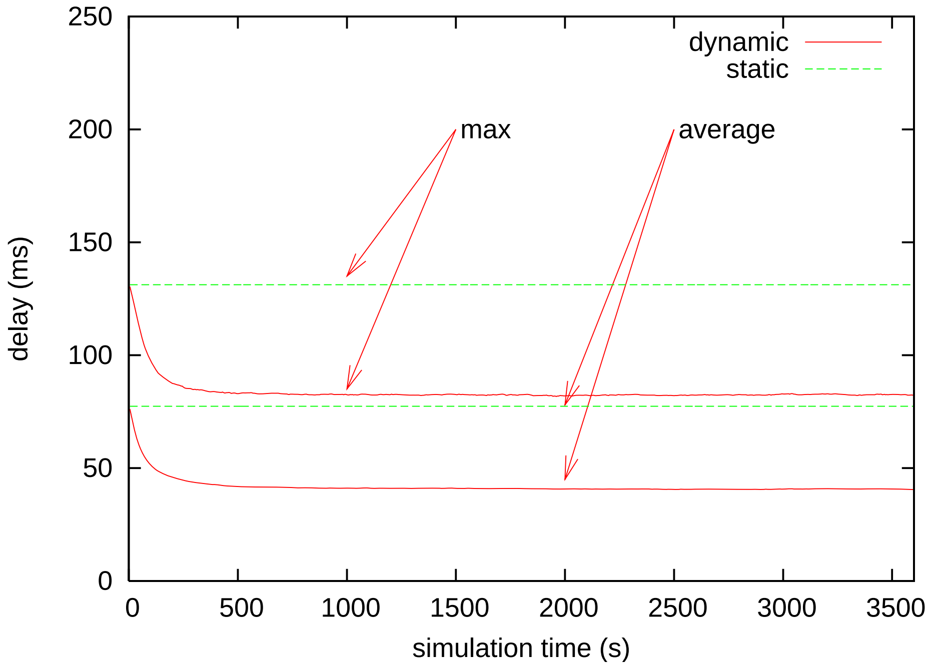
candidates is as long as about 82 msecs, i.e., less than 100 msecs [22]. It means that candidates can have a smooth and interactive conference with 1000 participants by constructing a hierarchical network.

Figure 38 illustrates the result of the case with 100 candidates. Again, the maximum delay is considerably reduced from 198 msecs (Fig. 28) to 141 msecs for candidates to peers and from 184 msecs (Fig. 28(b)) to 126 msecs for candidates. Although the maximum delay among candidates is larger than 100 msecs, we can expect a smooth conference among actively speaking candidates gathered near the root of distribution trees.

In Fig. 39, we further increased the number of candidates to 500. Comparing to results for the case of 100 candidates (Fig. 38), The maximum delay among candidates increases much, whereas the increase is small for the maximum delay from candidates to all peers. It is because that the number of candidates is too large to move toward the root and keep all of them near the root. We also notice the increase of the delay, which was also observed in the case of single distribution tree with many peers. It is because of the large number of candidates and the random tree assignment. Since the number of candidates is different among distribution trees, some trees suffer from the increase more than the others. If we introduce another adaptation mechanism to move candidates

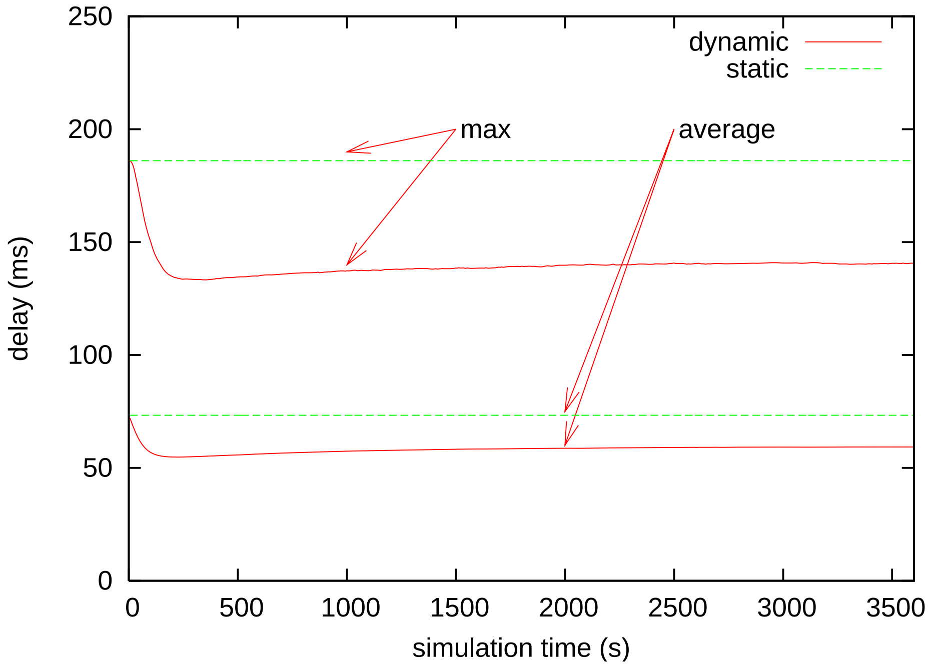


(a) All peers

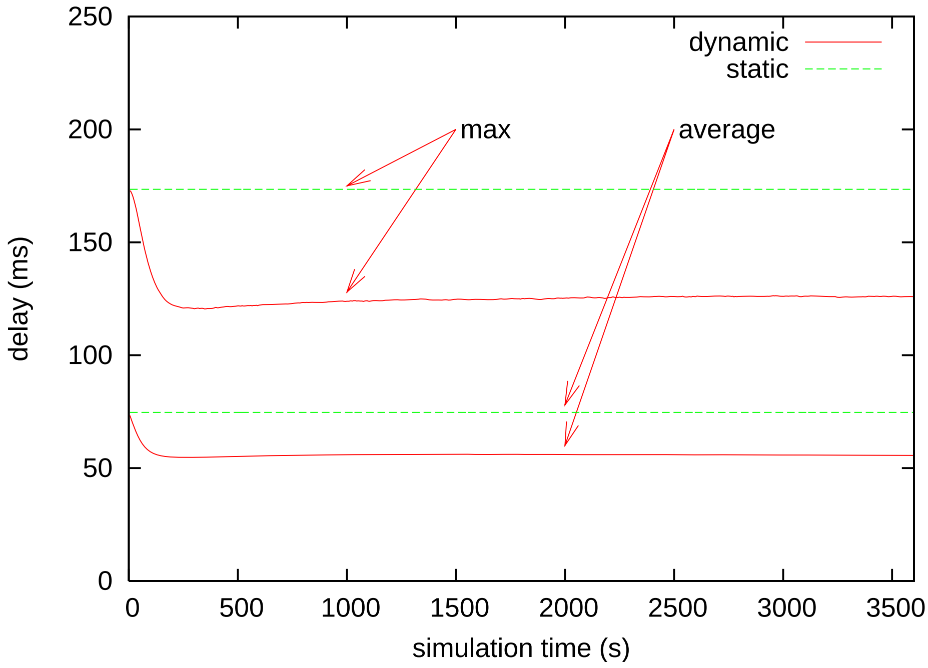


(b) Candidates

Figure 37: Delay between candidates and peers (BA model, fanout defined by link type, 1000 peers, 10 candidates, 10 trees)

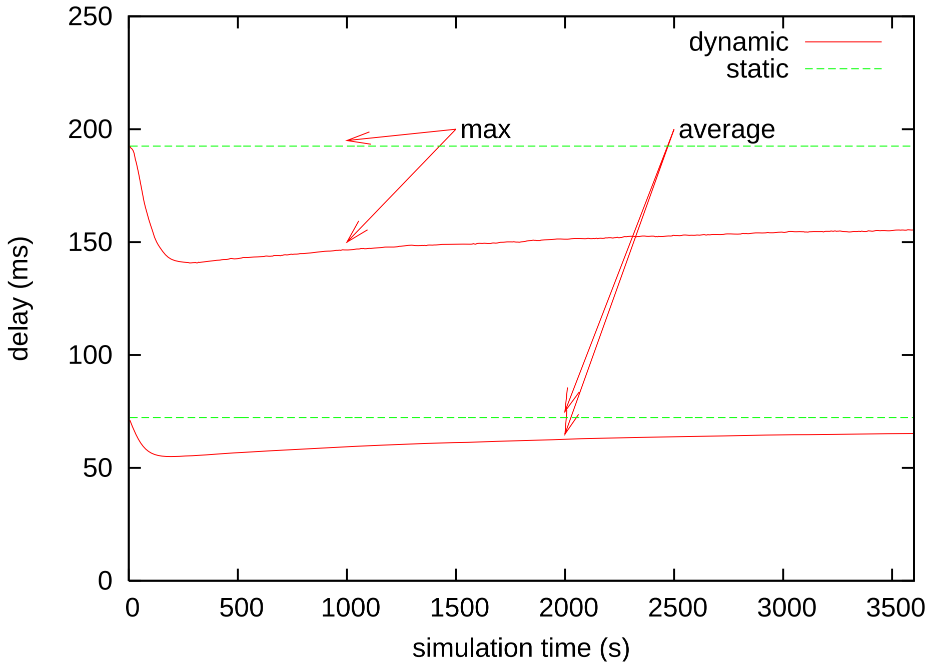


(a) All peers

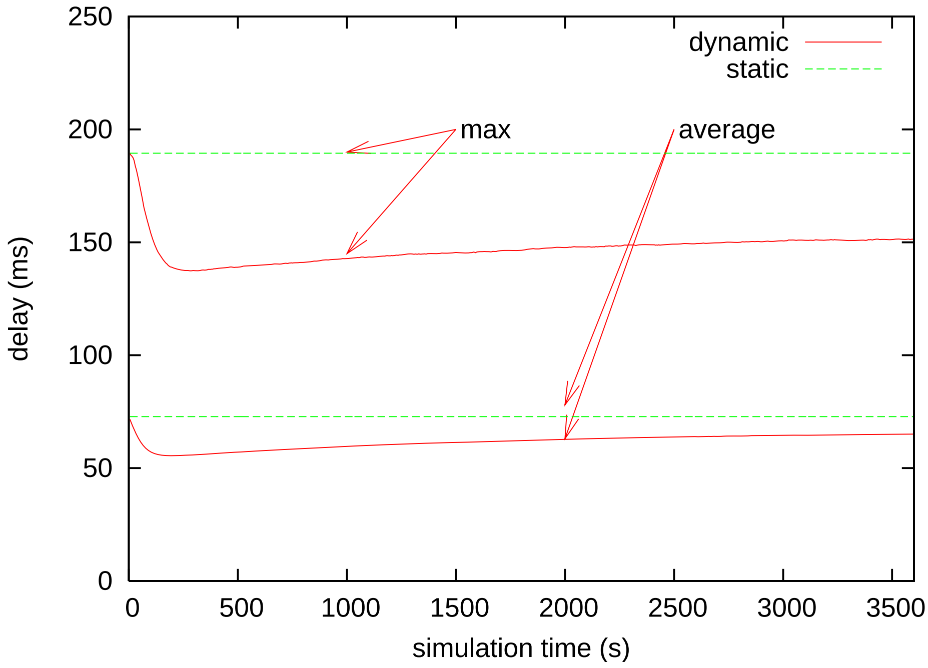


(b) Candidates

Figure 38: Delay between candidates and peers (BA model, fanout defined by link type, 1000 peers, 100 candidates, 10 trees)



(a) All peers



(b) Candidates

Figure 39: Delay between candidates and peers (BA model, fanout defined by link type, 1000 peers, 500 candidates, 10 trees)

from one tree to another and balance the number of candidates among distribution trees, we can expect the decrease of the delay. It remains as a future work.

4.3 Results on Load

Figures 40, 41, and 42 show the frequency distribution of load on link for the case with 10, 100, and 500 candidates, respectively. Figures 43 through 45 are for the load on peer. In comparison to the results for a single distribution tree (Figs. 30, 31, 32 and 33), the maximum load on peer is slightly smaller for the smaller number of candidates per tree. The reason for the cliff-shaped distribution in Fig. 45 is that the duration of a conference is rather short for many candidates.

The average number of messages that a peer handles in the initialization phase and a 60-minutes conference are 376, 355, and 342 for the case of 10, 100, and 500 candidates, respectively. The maximum are 1667, 1294, and 1259, respectively. The average load is similar with the result of a single distribution tree in subsection 3.4 (358 for 10 candidates and 349 for 100 candidates), but the maximum is much smaller (2790 for 10 candidates and 2580 for 100 candidates). It implies that our construction method of a distribution network is effective to have a smooth conference with the large number of participants.

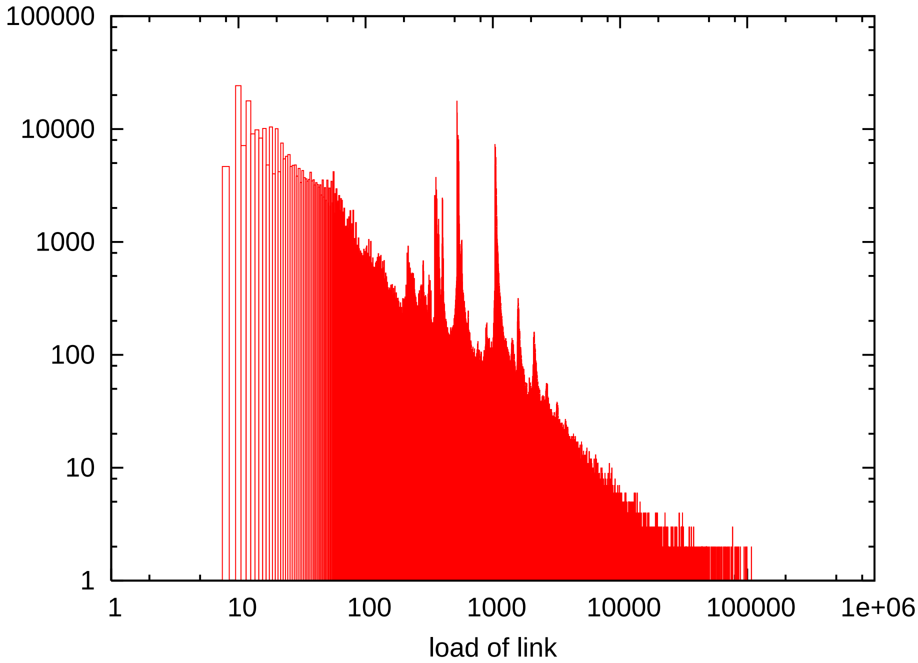


Figure 40: Frequency distribution of the load on link (BA model, fanout defined by link type, 1000 peers, 10 candidates, 10 trees)

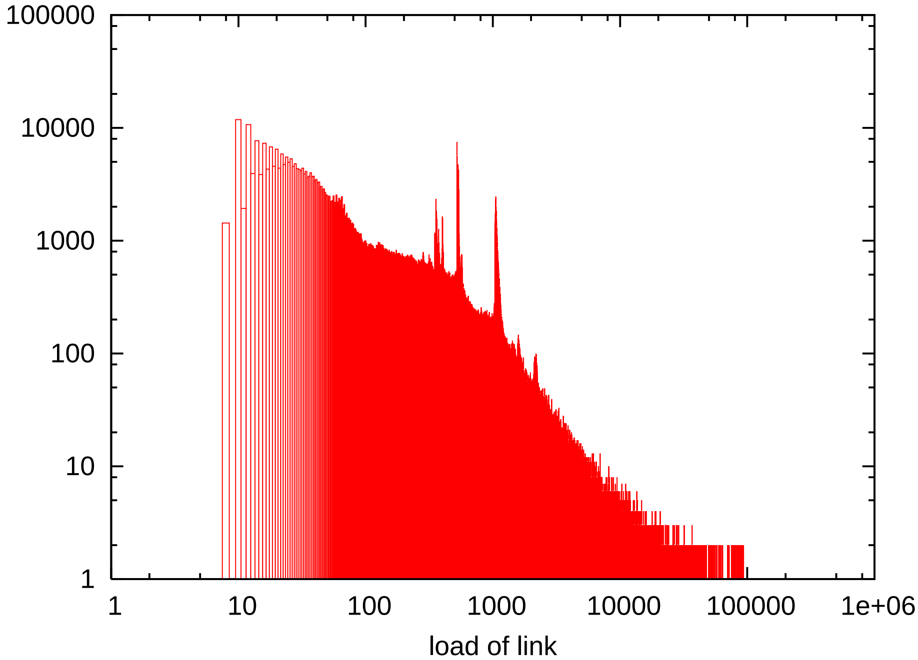


Figure 41: Frequency distribution of the load on link (BA model, fanout defined by link type, 1000 peers, 100 candidates, 10 trees)

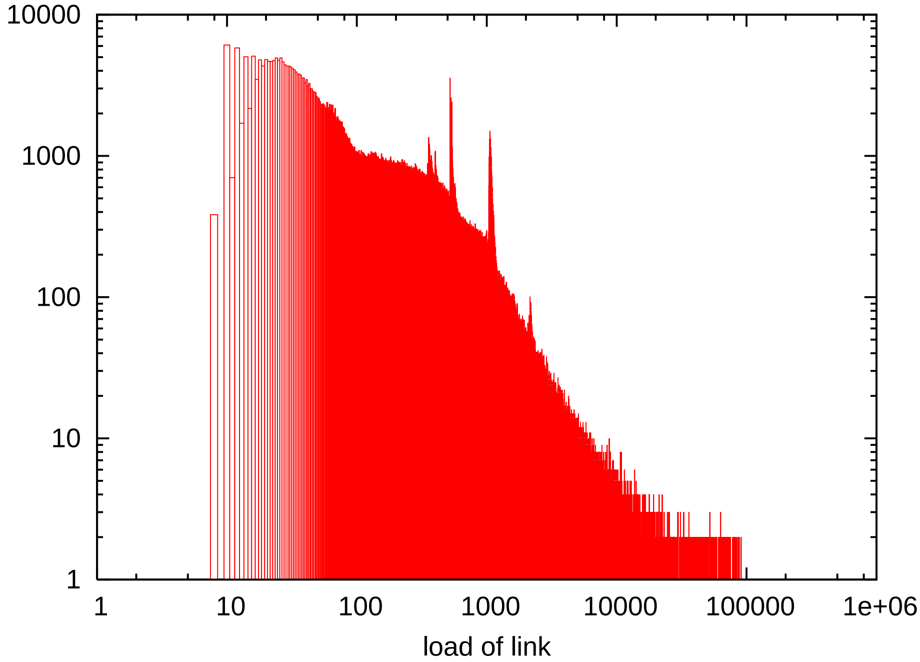


Figure 42: Frequency distribution of the load on link (BA model, fanout defined by link type, 1000 peers, 500 candidates, 10 trees)

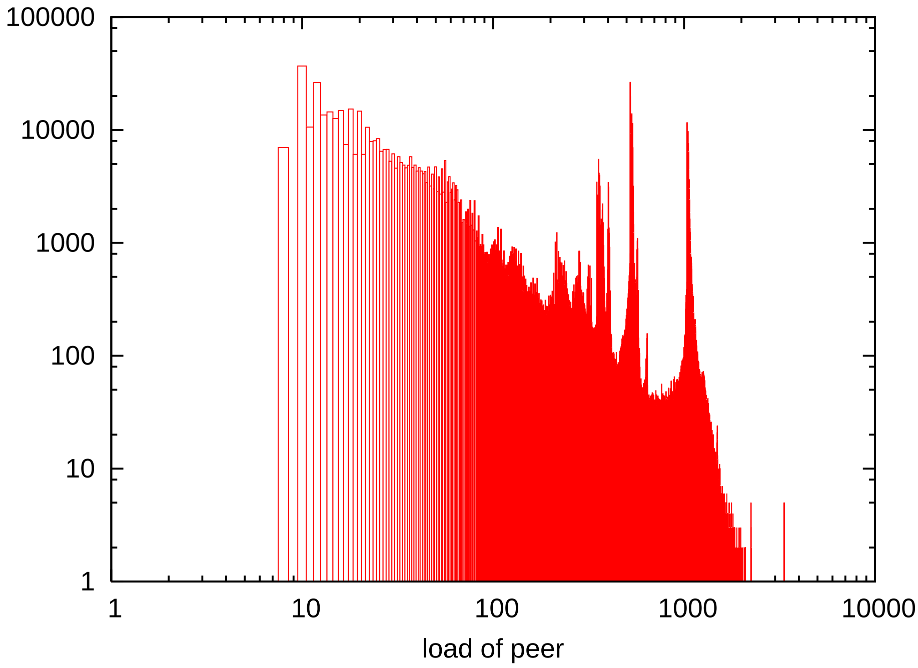


Figure 43: Frequency distribution of the load on peer (BA model, fanout defined by link type, 1000 peers, 10 candidates, 10 trees)

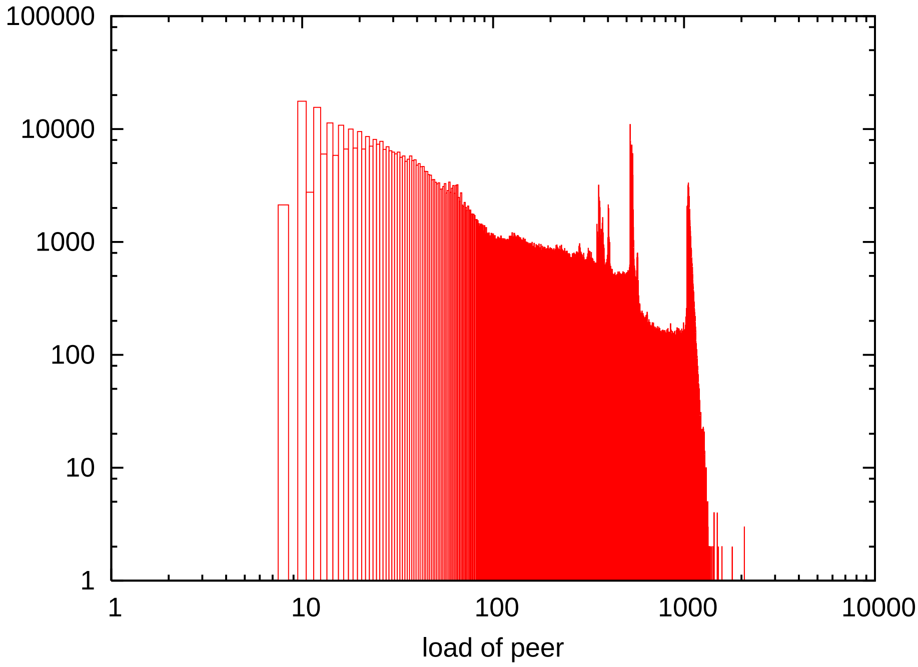


Figure 44: Frequency distribution of the load on peer (BA model, fanout defined by link type, 1000 peers, 100 candidates, 10 trees)

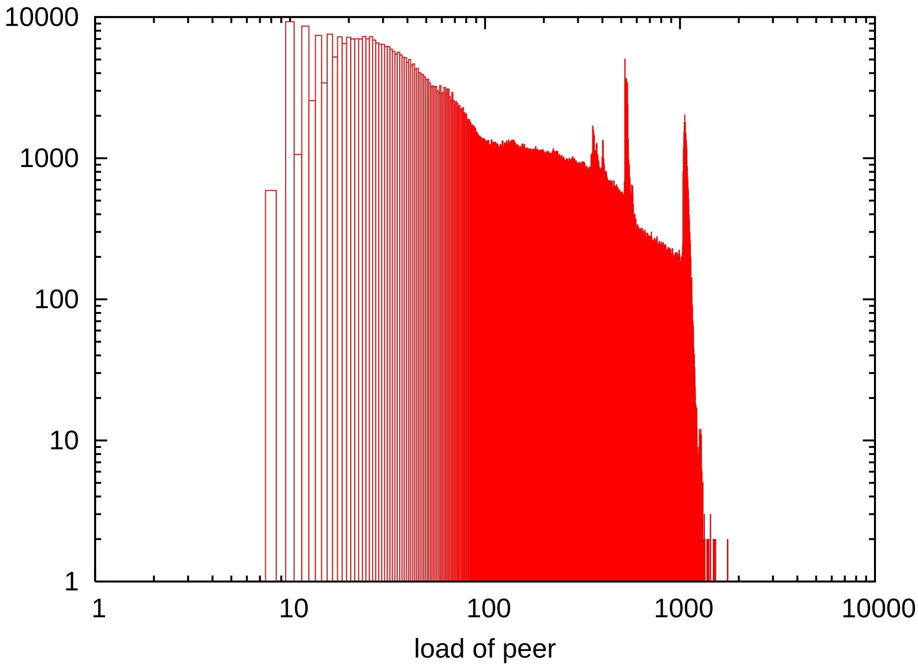


Figure 45: Frequency distribution of the load on peer (BA model, fanout defined by link type, 1000 peers, 500 candidates, 10 trees)

5 Conclusion

In this thesis, we proposed a construction method of an overlay network for scalable P2P conferencing systems. We showed that our method, constituting of the tree construction, tree reorganization, and failure recovery mechanisms, can offer smooth video conferencing with low delay and seldom and short freezes for 1000 participants. In addition, we showed that the load of control messages is very low. By connecting distribution trees by the core network, our system can accommodate hundreds or thousands of participants with the acceptable delay.

In the thesis, however, simulation experiments for the whole system did not take into account dynamic behavior of peers, including join, leave, and halt. Furthermore, we believe that the applicability and practicality of the proposal must be verified through experiments on a prototype. These remain as topics of future research works.

Acknowledgments

I would like to express my gratitude to my supervisor, Prof. Masayuki Murata of Osaka University, for his extensive and continuous support and invaluable comments on my studies. I would like to express my appreciation to Associate Prof. Naoki Wakamiya of Osaka University who has always given me appropriate guidance and valuable advice. I am deeply grateful to Profs. Koso Murakami, Makoto Imase, Teruo Higashino, and Hirotaka Nakano, Associate Prof. Go Hasegawa, Specially Appointed Associate Prof. Kenji Leibnitz, Research Assistants Shin'ichi Arakawa and Masahiro Sasabe of Osaka University, who gave me helpful comments and feedback. Finally, I heartily thank my seniors and colleagues in the Department of Information Networking, Graduate School of Information Science and Technology of Osaka University for their help.

References

[1] “SmoothCom.” available at http://www.zetta.co.jp/ecom/smoothcom/.

[2] “WarpVision.” available at http://www.ocn.ne.jp/business/infra/warpvision/.

[3] “Skype.” available at http://skype.com/.

[4] “iChat.” available at http://www.apple.com/macosx/features/ichat/.

[5] “PalTalk.” available at http://www.paltalk.com/.

[6] “NetMeeting.” available at http://www.microsoft.com/windows/netmeeting/.

[7] “Polycom.” available at http://www.polycom.com/.

[8] “LifeSize.” available at http://www.lifesize.com/.

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

[10] 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 (ICC 2006)*, pp. 158–161, June 2006.

[11] 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.

[12] X. Jin, K.-L. Cheng, and S.-H. Chan, “Sim: Scalable island multicast for peer-to-peer media streaming,” in *Proceedings of IEEE International Conference on Multimedia Expo (ICME 2006)*, pp. 913–916, July 2006.

[13] R. Zhang and Y. C. Hu, “Borg: a hybrid protocol for scalable application-level multicast in peer-to-peer networks,” in *Proceedings of the 13th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV 2003)*, pp. 172–179, June 2003.

[14] A. Nicolosi and S. Annapureddy, “P2PCAST: A peer-to-peer multicast scheme for streaming data,” in *Proceedings of the the First IRIS Student Workshop on Peer-to-Peer Systems (ISW 2003)*, Aug. 2003.

[15] S. Suetsugu, N. Wakamiya, and M. Murata, “A hybrid video streaming scheme on hierarchical P2P networks,” in *Proceedings of The IASTED European Conference on Internet and Multimedia Systems and Applications (EuroIMSA 2005)*, pp. 240–245, Feb. 2005.

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

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

[18] A. Medina, A. Lakhina, I. Matta, and J. Byers, “BRITE: An approach to universal topology generation,” in *Proceedings of the Ninth International Symposium in Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS 2001)*, pp. 346–353, IEEE Computer Society, Aug. 2001.

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

[20] E. W. Dijkstra, “A note on two problems in connexion with graphs,” *Numerische Mathematik*, vol. 1, pp. 269–271, Dec. 1959.

[21] T. Kawahara, “Recognition and understanding of voice communication among humans (in japanese),” tech. rep., Kyoto University, 2005. available at `http://www.ar.media.kyoto-u.ac.jp/lab/project/`.

[22] ITU-T, “Recommendation G.114 - one-way transmission time,” *Switzerland*, Feb. 2003.