IEICE TRANS. INF. & SYST., VOL.E89–D, NO.12 DECEMBER 2006

PAPER Special Section on New Technologies and their Applications of the Internet IV

Proactive Route Maintenance for Tree-Based Application Layer Multicast and Its Implementations

Tetsuya KUSUMOTO[†a)], ***Student Member***, **Jiro KATTO**[†], ***and*** **Sakae OKUBO**[†], ***Members***

SUMMARY The purpose of this study is to maintain efficient backup routes for reconstructing overlay trees quickly. In most conventional methods, after a node leaves the trees, its child nodes start searching for the new parents. In this reactive approach, it takes a lot of time to find a new parent. In this paper, we propose a proactive approach to finding a next parent as the backup route node over the overlay tree before the current parent leaves. A proactive approach allows a node to find its new parent node immediately and switch to the backup route node smoothly. In our proposal, the structure of the overlay tree using a redundant degree can decide a backup route node without so much overhead. Simulations demonstrate our proactive approach can recover from node departures 2 times faster than reactive approaches, and can construct overlay trees with lower overheads than another proactive method. Additionally we carried out experiments over actual networks and their results support the effectiveness of our approach. We confirmed that our proposal achieved better streaming quality than conventional approaches.

key words: *application layer multicast, peer to peer streaming, overlay tree, proactive route maintenance*

1. Introduction

ALM (Application Layer Multicast) implements the multicast functionally at end-hosts. Different from IP multicasting, which unrealistically needs global deployment of routers with IP multicasting capability, ALM needs only installation of application software and requires no change in the current network infrastructure. In addition, it provides flexibility in routing such as multipath packet transfer and load balancing. The most active research area in ALM is design of routing protocols [2]–[14]. There are several measures to evaluate the effectiveness of the routing protocols as the following: (a) quality of the data delivery path, that is measured by stress, stretch and node degree parameters of overlay multicast tree, (b) robustness of the overlay, that is measured by the recovery time to reconstruct a packet delivery tree after sudden end host failures, and (c) control overhead, that represents protocol scalability for a large number of receivers.

In the ALM session, each end host is a member of the delivery tree, and it leaves freely and may fail sometimes. This is not a problem in IP multicast, because the non-leaf nodes in the delivery tree are routers and do not leave the multicast tree without notification. In ALM, one of the problems which we have to consider is to reconstruct the overlay multicast tree after a node departure. The time of disconnecting is important for multicast applications such as live media streaming, because all the child nodes can not receive packets for the time. Quick reconstructing the overlay trees is therefore quite important to maintain the media quality, but little attention has been given to this problem. Most researchers use a reactive approach, in which nodes start searching for their new parent after departure of their old parent nodes. It usually takes several seconds to restore the overlay tree. It is therefore important to find an effective mechanism to reconstruct the overlay trees.

Proactive approach takes into account the node departure before it happens. The basic idea is that each non-leaf node in the overlay multicast tree pre-computes a backup route. In Probabilistic Resilient Multicast (PRM) [11], each host chooses a constant number of other hosts at random and forwards data to each of them with a low probability. It enables each host to have a backup route. However, PRM generates extra data overhead.

Another proactive approach was proposed by Yang et al. [12], which we call Yang's approach in this paper. It calculates the *degree* each host has, and ensures backup route proactively whenever a node leaves or joins. *Degree* represents a outbound link. It is inevitable to consider the degree bound in overlay multicast, which can be easily observed in streaming applications.

Each host limits the number of child nodes on the tree it is willing to support. For example, assume the bit rate of media is B and the outbound bandwidth of an end host is bi. The total number of connections it can establish with the outside world is $[bi/B]$. We describe the total number of the connections as *maximum degree* of the end host. In Yang's approach, a parent node calculates the *residual degree* of its child nodes first. *Residual degree* is represented as unused degree. Let $d_m(x)$ be the maximum degree of, $d_u(x)$ be the used degree and $d_r(x)$ be the residual degree of node x. Obviously $d_m(x) = d_u(x) + d_r(x)$.

With the degree constraints, when its child nodes do not have enough residual degrees to ensure their backup routes, the parent node employs the residual degrees of grandchild nodes and below in calculating until they can finally ensure backup routes. As the number of nodes increases, the search area becomes large. This calculating process generates extra data overheads and is not scalable. Volume of control traffic can be significant for some overlay multicast applications.

We therefore propose a new proactive approach in order to avoid the degree limitation and generating heavy

Manuscript received March 13, 2006.
Manuscript revised June 22, 2006.

[†]The authors are with the School of Science and Engineering, Waseda University, Tokyo, 169–0072 Japan.
a) E-mail: kusumoto@katto.comm.waseda.ac.jp

DOI: 10.1093/ietisy/e89–d.12.2856

overheads. By forcing at least one reserved degrees in each host, backup routes can be always established among the parent of x (i.e. the grandparent of x's child nodes) and child nodes. Our proposal achieves fast recovery in low overhead because backup route search is handled in local area. Backup route search of our proposal does not depend on the session size. It means that our proposal does not generate much overhead to ensure backup routes. Our proposal can be applied to the applications of any size of session. Therefore the differences are the way of searching backup route and its scalability. We have carried out extensive simulations and demonstrate that our proposal can recover from node departures two times faster than reactive approaches and can achieve much lower overheads than Yang's proactive method. Although reserved degrees cause slight increase in delay due to the tree becoming higher, this disadvantage diminishes as the maximum degree increases. Furthermore, we implemented our proposal in software, and experimented with P2P live video streaming over the actual network. The results of our implementation verify the effectiveness of our approach and convince us that our proposal achieved better streaming quality.

The rest of the paper is structured as follows. The next section provides an overview and the problem description of ALM protocols. Section 3 provides our proposal. Section 4 presents the simulation and implementation results. Finally, Sect. 5 concludes the paper.

2. An Overview of ALM Protocols and Problem Description

2.1 Overview of ALM Protocols

Most application layer multicast protocol studies have focused on how to construct an efficient multicast tree. ALMI [2] employs a centralized solution. In a centralized scheme, a central controller computes and instructs the construction of the delivery tree based on the information of metrics provided by the end hosts, hence the load of the controller is large. In decentralized solution, end hosts exchange the metric information each other, and constructs an overlay network. There are Mesh-first protocols and Tree-first protocols in the decentralized solution. Narada [3] and Scattercast [4] are known as Mesh-first protocols. Each host in Mesh-first protocols keeps many connections to keep the session stable, hence the overhead to maintain the mesh topology becomes large. In contrast, Yoid [5], Overcast [6] and Peercast [7] are Tree-first protocols for larger groups. Tree-first protocols build the distribution tree directly. The constructed tree is rooted at a single node. For single-sender applications like content distributions from a well-known source, the tree root acts as the data source. The overhead to maintain the tree topology is less than that of the Mesh-first protocols, but the sessions are less stable because the connections of each node are fewer than those of the Mesh-first protocols. OMNI [8] defines a local transformation periodically for the overlay tree to minimize the average latency of the entire hosts with degree constraints. In this paper, we do not consider dynamic tree improvement. ZIGZAG [9] and NICE [10] try to achieve with low control overhead by building an overlay of hierarchical clusters. Our proposal does not adopt a hierarchical structure.

Node departures in overlay network have been recognized in more recent works. Methods of reconstructing the tree methods are mainly divided into reactive approach and proactive approach. Finding the next parent node is done after a node departure in reactive approach, on the other hand before a node departure in proactive approach. Peercast uses a reactive approach to deal with node leaving or failures in overlay multicast. After a node departure happens, the affected nodes start to find the new parent nodes to receive data packets. We choose Peercast as comparison protocol to our proposal, and explain details at Sect. 2.2. PRM [11] uses a proactive approach for the overlay multicast. It constructs a tree first. Randomized forwarding of this method enables fast recovery from failure of overlay nodes. Randomized forwarding generates redundant data packets constantly for node departures. Another proactive approach [12] that we call Yang's method in this paper, uses a backup parent. The backup parent is decided by calculating the residual degrees of nodes. Finding the residual degrees generates overhead. We also choose Yang's method as comparison protocol to our proposal. Our proposal achieves lower overhead for holding a backup route of each node. We will describe the difference between our proposal and Yang's proactive approaches in Sects. 2.3 and 3.

Multiple Description Coding (MDC) is used in Split Stream [13]. This splits a media stream into multiple stripes, and uses a separate multicast tree to distribute each stripe. Even if affected nodes cannot receive one stripe after s node departure happens, they can continue playing media stream by using other stripes. This paper does not incorporate MDC, but it can be easily done by applying our method to the delivery tree of each description.

2.2 Reactive Approach

Most of existing ALM methods employ a reactive approach, in which tree recovery is initiated after node departure. In this reactive approach, a node which leaves the overlay tree sends a message to inform other nodes to be affected by its leaving such as its parent and child nodes. The child nodes cannot receive data temporarily until they connect to a new parent node. When a node suddenly fails, it cannot send a message to affected nodes, and they will not notice the failure for a while. Heartbeat mechanism helps the affected node to notice the failure. The parent and child nodes send a heartbeat packet to each other periodically. When a child node fails to receive heartbeat packets from the parent node over a period of time, it figures its parent node to be in a failure. However, the child nodes need a timeout period to recognize the failure. They cannot receive data flow all that time. Peercast proposed several recovery processes after a node departure: Root, Root-All, Grandparent and

Grandparent-All. In these methods, it has been shown that the grandparent approach is most efficient, in which each of its child nodes receives information of the grandparent from the departed node and contacts the grandparent when a parent node leaves the tree. Subtree rooted at each of its child nodes is maintained. If its degree is exhausted, the grandparent will redirect them to its descendant. When a node fails, the child nodes contact the root node because the child nodes cannot recognize their grandparent. Therefore, in the reactive approach, it is inevitable that it takes a lot of time to find a new parent. We choose Peercast algorithm with the grandparent process as a comparison with our proposal.

2.3 Proactive Approach

In a proactive approach, each host has a backup route to recover from the parent departure. Once a node departure happens, affected nodes connect to their backup route node, thus affected nodes can receive data flow after lower time of disconnection than that of the reactive approach.

In Yang's proactive approach [11], each non-leaf node decides a backup parent for its child nodes. A backup route is ensured by using residual degree of nodes in the overlay tree. Each host uses (1) to figure out if its all child nodes can form a backup route.

$$\sum_{j=0}^{n-1} d(C_j) \geq n-1 \tag{1}$$

A node in multicast session has n child nodes $\{c_0, c_1, \cdots, c_{n-1}\}$. $d(C_j)$ is the residual degree of the child C_j. $\sum_{j=0}^{n-1} d(C_j)$ means the sum of the residual degrees of the child nodes. First, a parent node calculates residual degrees of the child nodes. If the total residual degree of the child nodes is not less than $n-1$, all its child nodes can form their backup routes. If not, the child nodes cannot. In this case, the parent node calculates the total residual degree including the residual degree of descendants of the child nodes. Second, the parent node selects the child that has the smallest latency from the grandparent to it. The child holds the backup route to the grandparent. The subtree of the child node which holds the backup route supplies a backup route to the other child nodes. Then, the descendants of the child and the child nodes measure the latencies to the other child nodes, and the smallest edge is selected. This operation is repeated until the all child nodes hold their backup routes.

In Fig. 1, we outline the algorithm of Yang's proactive approach to form a backup route. The parent node is node 3, and its child nodes are node 5, 6 and 7. The maximum degree of each node is 3. The sum of the residual degree of them is less than $(n-1)$, where $n=3$ in this case, hence their total residual degree is less than 2. They cannot form their backup routes among them. Then node 3 finds the descendants of its child nodes to make the total residual degree larger than or equal to 2. When the total residual degree of the child and the grandchild nodes become larger than or equal to 2, the child nodes can form their backup route. In

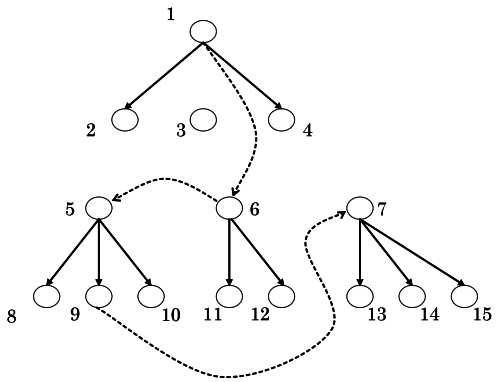


Fig. 1 Finding a backup route in Yang's approach.

Fig. 1, node 6 has the smallest latency to grandparent node and holds the backup route to node 1. Next, node 6, 11 and 12 measure their latencies to node 5 and 7. By the result, node 5 holds the backup route to node 6. Node 7 holds the backup route to node 9 by the same way. In this case, the backup routes of the child nodes are formed by using the residual degrees of the child nodes and grandchild nodes. However, searching the residual degrees does not always finish in the child and grandchild nodes. When this operation continues in lower layer, it tends to generate many packets.

As mentioned above, the reactive approach takes a lot of time to recover from node departures, and the previous proactive approaches generate extra packets. We therefore propose a proactive approach which suppresses extra packets as described in next section.

3. Proactive Route Maintenance over Redundant Overlay Trees

In our proposal, each node holds its backup route with low overhead. We construct an overlay tree without each node exhausting its degree. Each node constantly has residual degrees not less than 1. We apply the word "redundant overlay tree" to this structure. The child nodes of each node can ensure their backup route between the grandparent node and them by using their residual degree. This simplifies backup route calculation and contributes to overhead reduction. We show our proposal in detail below.

3.1 Tree Construction

First, we show the process of node joining the overlay tree in Fig. 2. It is assumed that maximum degree of each node is equal to 4. We then limit the active degree of each node to 3 and reserve 1 degree for backup route maintenance. In previous work, when new node 8 requests to connect to node 1, node 1 accepts node 8 to join as its child, because its degree is not exhausted. However, in our proposal, node 1 refuses the request because the residual degree of node 1 is only 1. Node 8 sends a join request to node 2 after receiving a redirect message from node 1. As a result node 8 becomes a child of node 2.

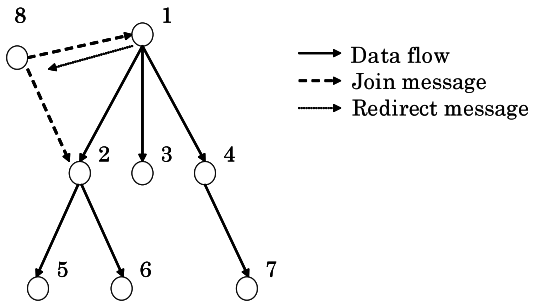


Fig. 2 New node participation process.

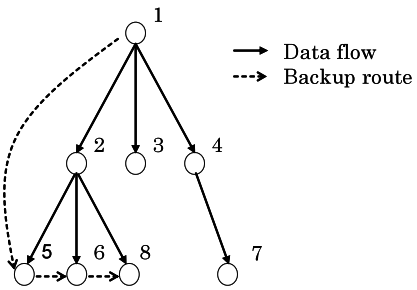


Fig. 3 Finding a backup route.

3.2 Backup Route Construction

Next, we show how to decide the backup route of each node in our proposal in Fig. 3. When node 8 joins the overlay tree and become a child of node 2, node 2 updates its child list. Node 2 sends the child list to node 1. After that node 1 measures a round trip time between node 1 and each node written on the list, and ranks the nodes in ascending order. Lastly node 1 informs them of their backup route. A node having the smallest round trip time holds a backup route to the grandparent. The second node has a backup route to the smallest RTT node, and the third node has a backup route to the second node. A node other than the smallest RTT node has the backup route to the next smaller RTT node than itself. In Fig. 3, if the ascending order of the nodes in round trip time is node 5, 6, 8, the smallest RTT node 5 has the backup route to node 1. The second node 6 has the backup route to node 5. The largest RTT node 8 has the backup route to the second node 6. In a specific case, if the child list of node 2 includes node 8 only (i.e. no other child node exists), node 2 immediately informs node 8 that node 1 is a backup route of node 8. We show the pseudo code of backup route calculation of our proposal in Fig. 4.

This backup route calculation is carried out whenever a node joins, leaves and fails. When a node leaves the overlay tree, the backup route is immediately applied and the new backup route calculation is initiated. Note that the backup route calculation is required only at the child layer of the departure node. It never goes down to the lower layers unlike in the previous approach.

In some rare cases, a node cannot use its backup route.

Backup Route Calculation
// S is set of grandparent and the child nodes
1. SortedS ← Sort S in ascending order of RTT from grandparent node
//{ SortedS[0] = grandparent node }
2. for i ← 1 to N do
3. SortedS[i].BackupRoute ← SortedS[i-1]
4. end for

Fig. 4 Pseudo code of backup route calculation.

When the current parent and backup parent node leave or fail at the same time, the node cannot connect to a new node immediately. Another case is that a node is not informed of its backup parent node. This happens when the parent node leaves the tree without noticing the node of its backup parent node before the backup route calculation is finished. In [11], handling these cases is shown. In this case, it uses the ancestor-list, which contains node information from grandparent to root nodes. Our approach also uses the same method in such cases. In the method, when a node is connected to its backup parent node and the backup parent node does not reply, it uses the ancestor list. First, it ordinarily joins the grandparent node and it follows the redirection algorithm deciding whether the grandparent node accepts the node or not. When the grandparent node does not exist because the grandparent node has left or failed at the same time as the parent has, the node tries to connect to a node in higher layers of the ancestor list. In the experiments which we present in Sect. 4, we do not use ancestor-list. If the current parent node and the backup route node leave at the same time, the affected node sends the join message to the root node of the tree.

Backup routes created in the redundant overlay tree are certainly efficient as long as each host does not exhaust its degree. However it is possible that a node exhausts its degree by accepting a node rejoining in the backup route procedure. When this happens, a tree reconstruction procedure is invoked by the node itself in order to keep the route redundancy. This procedure is carried out by asking the child nodes of a backup route node except the newly connected node whether their degree is exhausted. At the time the newly connected node finds that a certain node has residual degrees, the newly connected node moves to the node that has the most residual degree. We show the procedure in Fig. 5. Node 2 uses up its degree because node 8 joined node 2 as its backup route. Node 2 sends a query to other child nodes, which are nodes 5, 6 and 7, and they reply hit or fail messages to node 8. The hit message means it can accept joining of new hosts. The fail message means it cannot accept. Node 8 moves to the node which has sent the hit message first. In Fig. 5, node 6 sends a hit message to node 8, and node 8 joins node 6. If all messages of the child nodes are fail, the newly connected node joins the node which it has received a message first from. It receives a redirection message from the first node. We show the pseudo code of tree reconstruction of our proposal in Fig. 6.

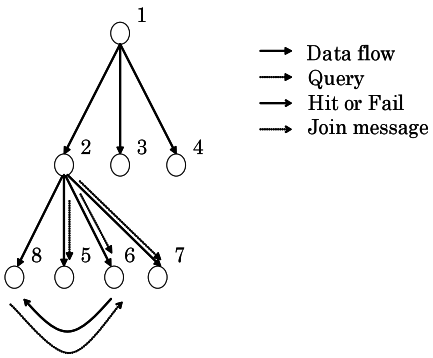


Fig. 5 Reconstruction of the redundant tree.

Tree Reconstruction

```
1. Connect (BackupRoute)
2. if BackupRoute.Degree ==BackupRoute.MaximumDegree
then
3.    Query (SiblingNodes)
4.    if Reply == HIT then
5.       JoinMessage (FirstHitNode)
6.    else
7.       JoinMessage(FirstReplyNode)
```

Fig. 6 Pseudo code of tree reconstruction.

3.3 Problem of Non-contributing Node

One question in our proposal is that there are some nodes whose maximum degrees are zero or one. Existence of nodes with zero degree (namely receiving only) is a common problem in ALM. Nothing could be done but they are treated as a leaf node in the overlay tree. This is similar to the case of an incentive approach adopted by recent P2P file sharing system like BitTorrent [16]. Handling of the nodes which have one maximum degree is a specific problem in our proposal, because we construct the redundant overlay tree by forcing at least one degree reserved in each node. A node of one degree can not have a child node. In the case that the maximum degrees of all the child nodes of a node are one, our proposal cannot construct a subtree rooted at the child nodes, so the tree can not be constructed effectively. In the worst case that the maximum degrees of all the child nodes of the root node are one, our proposal can not construct the tree any more. To avoid this case, we allow the nodes of one maximum degree to have a child although their degree is one. Another problem is that they cannot provide backup routes because of their maximum degree of one or zero. We then decide that each node can have only one node whose maximum degree is one or zero, and place the node at the end of the backup spanning tree so that the node need not provide a backup route. We show this case in Fig. 7. Node 2 is a parent of three child nodes, which are node 3, 4, and 5. The maximum degree of node 5 is one. We place the node 5 at the end of the spanning tree of the backup routes, and node 5 needs not provide a backup route to other nodes. Finally, all the child nodes can get their backup routes.

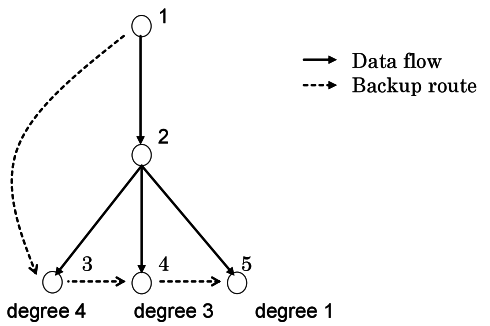


Fig. 7 Treating a node of one or zero maximum degree.

4. Performance Evaluation

We evaluate the performance of our proactive approach using simulations and software implementations. We are mainly interested in the resilience performance, how fast the overlay tree can be reconstructed and how small the control overheads can be kept by redundant backup routes. We compare our proactive method with a reactive method which uses grandparent policy described in Sect. 2.2. In simulations, we also compare our method with Yang's method, which is another proactive method described in Sect. 2.3. We show simulation results in Sect. 4.1 and implementation results in Sect. 4.2.

4.1 Simulation Results

We carried out the simulations by ns-2.26 [15]. We show simulation results in Figs. 9–17. Our simulation topology has 24 routers. Four routers of them are domain-to-domain routers as shown in Fig. 8. The others are set up at random between end-hosts. The distance between two end-hosts is represented as the sum of link delays on the shortest path between them. The delay and bandwidth between the domain routers are 100 ms and 100 Mbps. The delay between the routers in a domain varies from 10 to 50 ms and the bandwidth is 100 Mbps. The delay and bandwidth between a router and end-hosts are 10 ms and 10 Mbps. The nodes are randomly connected to one of the 20 routers except the four inter domain routers. The total number of nodes varies from 25 to 200. The link latency varies from 10 ms to 100 ms. The maximum degree of each node varies from 1 to 6. For the experiment in Fig. 16, we fixed the degree of each host at a particular value. The total simulation time is 300 sec. In the beginning, all nodes join the tree, and after that each node randomly chooses to stay, leave, fail or re-join per 15 sec intervals. We do not consider process delay for forwarding packets in the simulations.

4.1.1 Comparison of Recovery Time

First, we use the average and maximum recovery times as performance measures. Recovery time is the time for an affected node to find a new parent. Figures 9 and 10 plot the

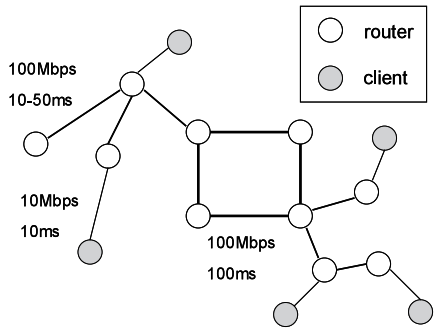


Fig. 8 Simulation topology.

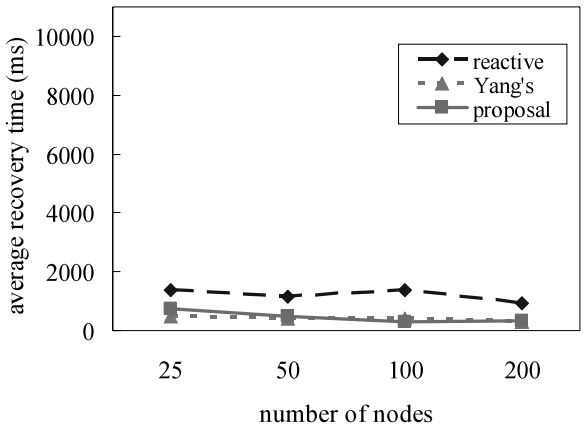


Fig. 9 Average recovery time with varying number of nodes in leave.

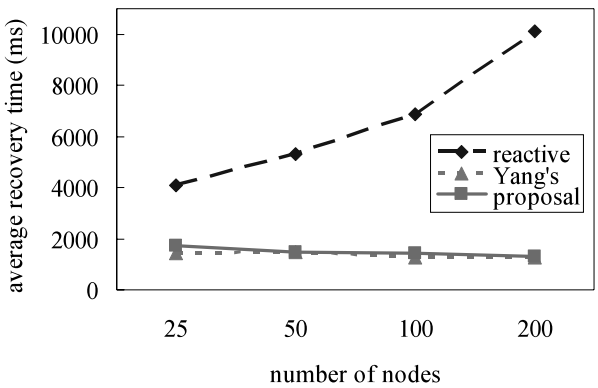


Fig. 10 Average recovery time with varying number of nodes in failure case.

average recovery time of leave and failure in simulations. Each node sends heartbeat message every one second. If a node does not receive any heartbeat messages from its connected nodes for one second, it decides that the nodes have become failure. Figure 11 shows the maximum recovery time of leave and failure cases of 200 nodes.

In Fig. 9, the average recovery time against node leaving in the reactive approach is about 1300 ms in each number of nodes. The average recovery times against node leaving in proactive method (our proposal and Yang's approach) are about 500 ms, less than about half of the reactive method.

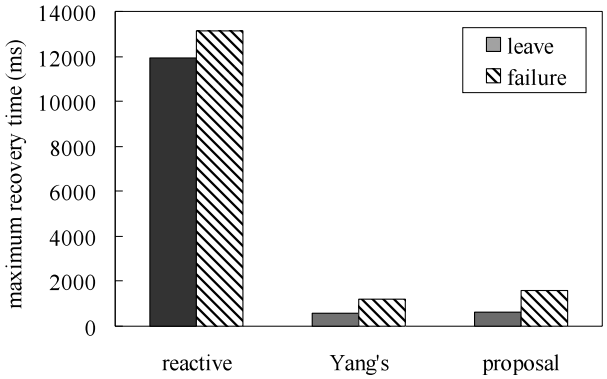


Fig. 11 Maximum recovery time with 200 nodes.

In Fig. 10, as the number of nodes increases, the average recovery time of the reactive approach becomes larger. The average recovery time of our proposal and Yang's approach are about 1400 ms.

The proactive methods enable the affected nodes to immediately connect to their backup routes. This is common to both proactive methods, hence their results are nearly equal. On the contrary, in the reactive approach, requests may be rejected by the contacted node due to degree constraint and redirection is repeated until the request is accepted. Especially in the node failure cases, affected nodes have to contact to the root nood in the reactive approach. As the number of nodes increases from 25 to 200, the recovery time of the reactive approach increases accordingly. This is because the height of an overlay tree becomes bigger, and many redirections happen.

In Fig. 11, we can see the trend notably. The maximum recovery times of the reactive approach are much larger than those of our proposal and the Yang's approach. The time of leave case of the reactive approach is as large as that of failure case. This is because, when the node close to the root node leaves the session, many affected nodes are redirected from the root node to the edge node of the tree. On the contrary, the maximum recovery times of our proposal and Yang's approach keep small against those of the reactive approach. This is because our proposal and Yang's proactive approach can connect the backup route node immediately both in leave and failure cases. The influence of failure is the time of noticing the node failure by heartbeat interval.

Our proposal and Yang's approach achieve fast recovery both in leave and failure case consistently, but the reactive approach does not show good performance in failure case and especially in maximum recovery time. Therefore, we can consider that the reactive approach is unsuitable for the real time applications.

4.1.2 Comparison of Control Overheads

We show the overheads of the reactive approach, Yang's approach and our proposal. Overhead is the total amounts of bytes of all control packets to maintain the overlay tree. Control packets are composed of Join, Redirect, Leave, and

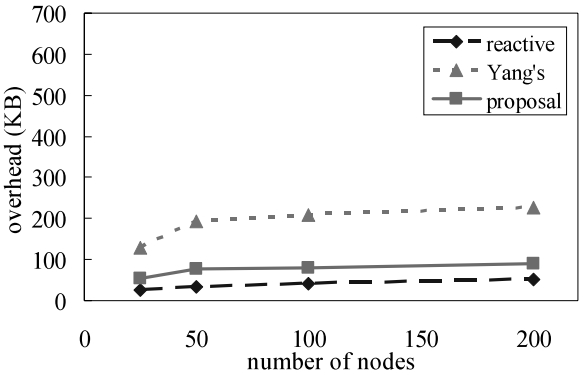


Fig. 12 Overhead of the round robin method with varying number of nodes in simulation.

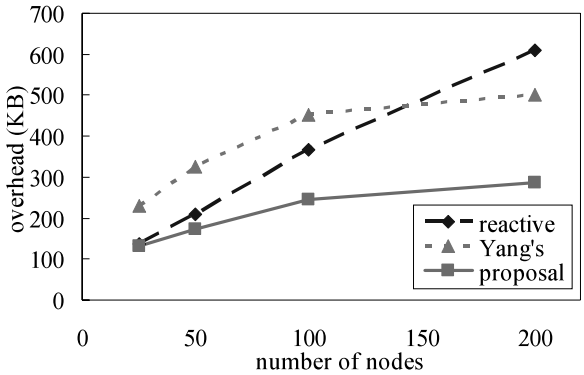


Fig. 13 Overhead of the round trip time method with varying number of nodes in simulation.

Backup messages. Join message is used in join request, Redirect message is used in redirect procedure, Leave message is used in noticing the node departure, and backup message is used to search backup routes. We assume the size of each packet is 128 bytes.

For the reactive approach, the control overhead comes from the control messages exchanged for the affected nodes to find new parents. For the proactive method, the control messages consist of two parts. 1) Similar to reactive approaches, control messages are exchanged for the child nodes of departure nodes to find their new parent, though we may need fewer steps in the proactive approach. 2) In addition, every non-leaf node exchanges information for deciding a backup route.

We experimented with two redirection methods; a round robin method and a round trip time method. In the round robin method, when a node whose degree is exhausted receives a join message from a newly joining node, the node redirects the message to each of its child nodes in pre-determined order. In the round trip time method, when a node whose degree is exhausted receives a join message from a newly joining node, the node sends its child list to the newly joining node. Then the newly joining node measures the round trip times between each node on the list and itself. After that the newly joining node sends a join message to the smallest round trip time node. The round trip time method uses more packets than the round robin method, but the overlay tree is optimized to be low latency.

Figure 12 compares the overheads of the round robin method for redirection. The number of nodes varies from 25 to 200 in simulations. In Fig. 12, we can see Yang's proactive approach generates higher overhead than others. In comparison with Yang's approach, our proactive approach suppresses the overhead. The reactive approach is the smallest in this respect, because the proactive approaches need to exchange information to decide backup routes. Furthermore, the round robin method for redirection does not generate so many control packets.

Figure 13 compares the overheads of the round trip time method for redirection. The number of nodes varies from 25 to 200 in simulations. In the reactive approach, as the number of nodes increases, the overhead increases a lot. This is because as the number of nodes increases, more redirections are required. Redirection generates a volume of overheads to measure RTT.

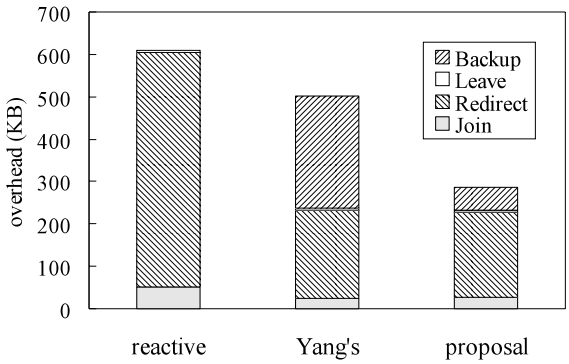


Fig. 14 Ratio of control packets.

As shown in Figs. 12 and 13, our proposal does not generate as many control packets as Yang's approach for holding backup routes. The redundant overlay tree structure of our proposal enables this. When changes of the tree structure happen by join, leave or fail, the nodes have to update the backup routes. As the session continues long time and many nodes join the session, the difference of the overheads between our proposal and Yang's approach will become larger. In Fig. 12, overheads of Yang's approach and our proposal are larger than that of the reactive approach, but in Fig. 13, the reactive approach generates more packets than the proactive approaches. In the case that nodes exchange much information in redirection and many nodes join the session, the reactive approach is not useful.

Figure 14 shows the rate of the control packets of the round trip time method with 200 nodes. We can see that the reactive approach generates many redirection messages and Yang's approach generates many backup route messages.

In most ALM protocols, each node joins the overlay tree following their own metric. This means that nodes exchange a lot of information to optimize the overlay tree in join and redirection process. ALM is also used in media streaming, where many people participate in the ALM

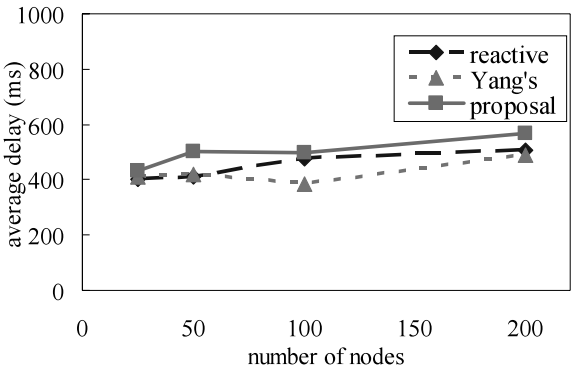


Fig. 15 Average delivery delay with varying number of nodes in simulation.

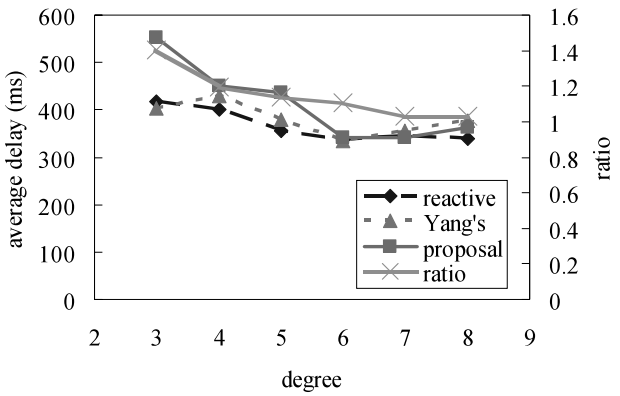


Fig. 16 Average delivery delay with 200 nodes with varying number of degree.

session. Consequently, the proactive methods are more suitable for ALM than the reactive approaches in terms of overhead. Furthermore, our proposal generates fewer packets than Yang's proactive approach for ensuring backup routes. Among the proactive approaches, our proposal can save bandwidth most.

4.1.3 Comparison of Data Delivery Delays

Proposed redundant overlay tree simplifies the backup route search and contributes to overhead reduction. However, that structure causes the height of the overlay tree to be larger and possibly leads to overall delay increase, because all nodes do not fully use their degree. Therefore, an obvious problem of our approach is increase in data delivery delays. Figure 15 shows how the average transfer latency in the tree varies as the number of nodes increases from 25 to 200 in simulations. Figure 16 shows the average delay when the maximum degree of all nodes is fixed at the same number with 200 nodes. Lastly, we show the maximum delay in Fig. 17.

In Fig. 15, we can see that the average delivery delay of our proposal is larger than other methods. The tree in our proposal trends to become larger stretch because each node does not exhaust its degree. However, the stress is smaller

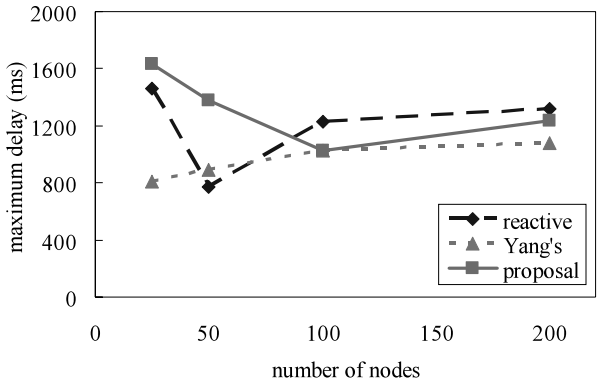


Fig. 17 Maximum delivery delay with 200 nodes with varying number of degree.

than other methods. Next, we show an interesting result in Fig. 16. Maximum degree of all nodes is fixed at the same number when the number of nodes is 200. Figure 16 shows the average delivery delay in each maximum degree and the ratio of the theoretical delay values between our redundant overlay tree structure and the ordinary overlay tree structure like the Yang's approach. In theoretical value, we assumed all the end-to-end delays are d and the tree is balanced tree. When the degree is fixed at three, delay of our proposal is largest. However, as the degree number increases, the difference between our proposal and the others becomes quite small. The average delivery delay of our proposal is about 380 ms like other methods when degree is fixed at 6, 7 and 8. We can recognize that, as the degree of node becomes larger, the difference between our proposal and the others becomes smaller. This is because larger degree contributes to reducing the overlay tree height. This leads to reduction of delay in the resilient overlay structure. Theoretical values of sum of the delay are estimated by $\sum_{i=0}^{M} N_i \times i \times d$. N_i, M and $i \times d$ represent the number of nodes which stay i-th layer (the depth of tree when root is 0 layer) of the tree, the maximum depth, and the delay of the node of i-th layer. The ratio is our redundant overlay tree by the ordinary overlay tree. The ratio is decreasing as the number of degree is increasing.

When the degree of node becomes large, the i-th layer of the tree can hold many nodes. Therefore, as the degree of each node increases, the ratio of number of nodes between our proposal and Yang's approach in the i-th layer comes to equal. Let k be the degree of each node, the maximum number of nodes of our proposal in i-th layer is $(k-1)^i$, and that of Yang's approach is k^i. The ratio is represented as $\frac{(k-1)^i}{k^i} = \left(1 - \frac{1}{k}\right)^i$, which approaches to unity as k increases. Therefore, we can expect that the difference of average delays between our proposal and the other methods become small when the degree increases. Increasing maximum degree is not easy, but it is possible in an application of low bit rate multimedia. What is more, we do not think that the difference of the delivery delays between our proposal and others is so critical in such a case of one-way streaming application.

In Fig. 17, when the numbers of nodes are 25 and 50, the maximum delay of our proposal is larger than those of other methods. This is because the tree construction with round trip time metric does not work effectively in small number of nodes, and the tree height of our proposal trends to become large. In small number of nodes, the nodes are scattered, so end-to-end delay trends to become large. Thus the delay from the root nodes along the tree becomes large by the tree height. When numbers of nodes are 100 and 200, the difference between our proposal and Yang's approach is small, and the maximum delay of our proposal is smaller than that of the reactive approach. This is because the tree is constructed enough by round trip time metric. In the reactive approach, when node departure happens, the affected all child nodes send join messages to their grandparent nodes and redirection procedure happen, and the redirected nodes have to connect the edge node of the tree because of degree exhausting. In that case, the maximum delay of the redirected node becomes large.

4.2 Implementation Results

In addition to the simulations, we implemented prototypes of the reactive approach and our proposal with C++ on Windows XP. Video codec is ITU-T H263+. Maximum degree of each node is fixed at 3. Total 25 nodes are deployed over three different networks and each network connects to the backbone in Japan. We can expect the backbone to have high bandwidth. We will revisit this in Sect. 5. Firstly in the experiment, the source node waits for joining nodes. The source node receives a join message, and it starts sending the data flow of a media, after encoding the media captured from the capture card in real time. When the source node exhausts its degree, it redirects the next joining node to its child nodes. All nodes join the ALM session, and then each node joins or leaves randomly for 30 minutes. We show implementation results in Figs. 18–21.

4.2.1 Comparison of Recovery Time

In Fig. 18, we show the average and maximum recovery time of 25 nodes in implementation. Recovery time of our proposal is less than half of the reactive approach value. This point is the same as in simulations. As compared to the reactive approach, we could confirm that the media playback quality of our proposal was much better than that of the reactive approach when node departures happen. In the reactive approach, playback was felt like "freeze frame" for a moment, but in our proposal, decoded pictures continued to play smoothly.

4.2.2 Comparison of Control Overheads

Figures 19 and 20 represent the overheads when the number of nodes is 15 and 25 in the implementations. Control packets are composed of same messages of the simulation. The size of each message is 50 bytes. In Fig. 19, we used the round robin method for redirection. Overhead of our proposal is more than that of the reactive approach. This is because the round robin method does not generate so much overhead in redirection and our proposal generates overhead for ensuring backup routes. On the other hand, Fig. 20 shows that overhead of our proposal is almost the same as that of the reactive approach at 25 nodes. We used the round trip time method for redirection. As the number of nodes increases, overhead of the reactive method increases.

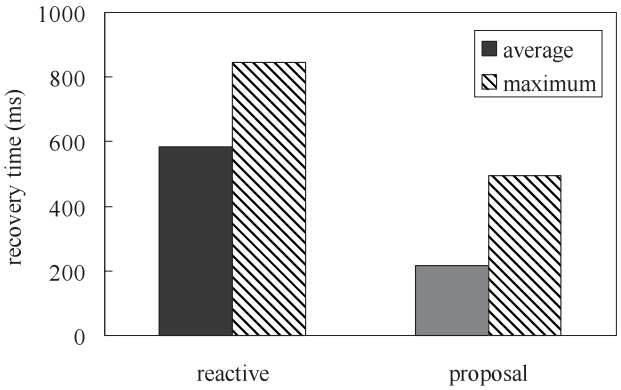


Fig. 18 Average and maximum recovery time with 25 nodes in implementation.

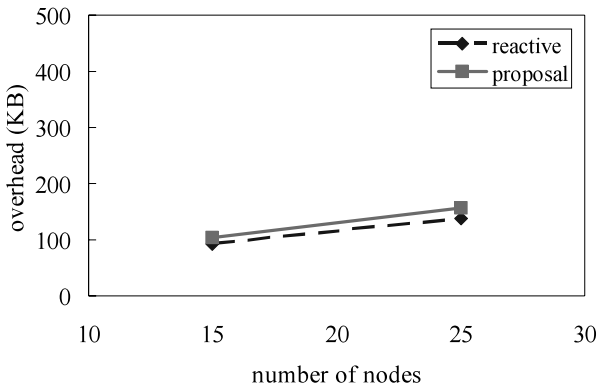


Fig. 19 Overhead of the round robin method with 15 and 25 nodes in implementation.

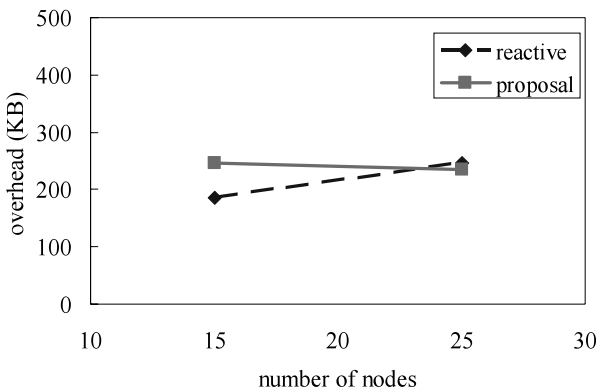


Fig. 20 Overhead of the round trip time method with 15 and 25 nodes in implementation.

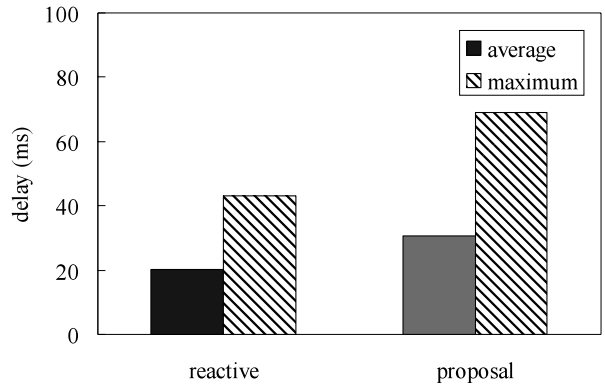


Fig. 21 Average and maximum delivery delay with 25 nodes in implementation.

We can also see this trend in the simulation result of Fig. 13.

4.2.3 Comparison of Data Delivery Delays

Figure 21 shows the average and maximum delivery delays in implementation when the number of nodes in session is 25. The delay is more in our proposal than the reactive approach. However, in media playback, we do not feel any difference between our proposal and the reactive approach. We think this difference is not so critical if we consider the delay caused by video coding and decoding.

5. Concluding Remarks

We presented a novel method of proactive route maintenance for ALM with the redundant overlay tree. It enables fast recovery from node departures and reduction of control overheads. In comparison with the reactive approach and Yang's proactive approach, our proposal is much faster in recovery than the reactive approach, and as fast as Yang's proactive approach. In implementations, the recovery time of our proposal is faster than reactive approach. We also realized that media playback quality of our proposal was much better than that of the reactive approach when node departures happen. Control overhead of our proposal is less than Yang's approach and, in the specific case it is less than the reactive approach. In our proposal, each node constructs the backup route in the local area. The overhead of searching backup route does not depend on the session size. Our proposal achieves scalability in holding backup route. Although the data delivery delay tends to be larger in our proposal than other methods, the difference from other methods becomes smaller as the degree increases.

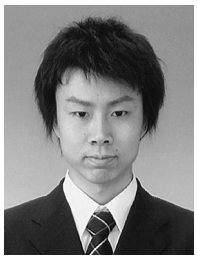
In future work, the implementation should be experimented in different environment which has bottleneck or large delay links. We will use NIST Net, which is a linux based network emulator [16]. NIST Net can emulate the critical end-to-end performance characteristics imposed by various wide area network situations, so we can set end-to-end delays and bandwidths. We will obtain more extensive results.

Acknowledgments

This research was supported in part by the NICT R&D project "Broadcast System Using Communication Network" and Grants-in-Aid for Scientific Research of the Ministry of Education of Japan on "Stream Caching for New Generation Content Delivery Networks and its Ubiquitous Extension."

References

[1] S. Deering, "Host extension for IP multicasting," RFC 1112, Aug. 1989.
[2] D. Pendarakis, S. Shi, D. Verma, and M. Waldvogel, "ALMI: An application level multicast infrastructure," Proc. USENIX USITS 2001, 99.49–60, March 2001.
[3] Y. Chu, S.G. Rao, and H. Zhang, "A case for end system multicast," Proc. ACM SIGMETRICS 2000, pp.1–12, June 2000.
[4] Y. Chawathe, S. McCanne, and E. Brewer, "Scattercast: An adaptable broadcast distribution framework," ACM Multimedia Systems Journal, vol.9, no.1, pp.104–118, July 2003.
[5] P. Francis, "Yoid: Extending the Internet multicast architectuire," http://www.icir.org/yoid/, April 2000.
[6] J. Jannotti, D. Gifford, K. Johonson, M. Kaashoek, and J. O'Toole, "Overcast: Reliable multicasting with an overlay network," Proc. 4th Symposium on Operating Systems Design & Implementation, pp.197–212, Oct. 2000.
[7] H. Deshpande, M. Bawa, and H. Garcia-Molina, "Streaming live media over peers," Technical Report 2002-21, Stanford University, March 2002.
[8] S. Banerjee, C. Kommareddy, K. Kar, B. Bhattacharjee, and S. Khuller, "Construction of an efficient overlay multicast infrastructure for real-time applications," Proc. IEEE INFOCOM 2003, vol.2, pp.1521–1531, April 2003.
[9] D. Tran, K. Hua, and T. Do, "ZIGZAG: An efficient peer-to-peer scheme for media streaming," Proc. IEEE INFOCOM 2003, pp.1283–1292, April 2003.
[10] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast," Proc. ACM SIGCOMM 2002, pp.205–217, Aug. 2002.
[11] S. Banerjee, S. Lee, B. Bhattacharjee, and A. Srinivasan, "Resilient multicast using overlays," Proc. ACM SIGMETRICS 2003, pp.102–113, June 2003.
[12] M. Yang and Z. Fei, "A proactive approach to reconstructing overlay multicast trees," Proc. IEEE INFOCOM 2004, pp.2743–2753, March 2004.
[13] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "Splitstream: High-bandwidth multicast in cooperative environments," Proc. ACM SOSP 2003, pp.298–313, Oct. 2003.
[14] B. Cohen, "Incentives build robustness in BitTorrent," 2003. http://bittorrent.com/bittorrentecon.pdf
[15] The Network Simulator ns-2, http://www.isi.edu/nsnam/ns
[16] The Network Emulator Nist Net, http://snad.ncsl.nist.gov/itg/nistnet/



Tetsuya Kusumoto received the B.E. degree in Electronic Information and Communication Engineering from Waseda University, Tokyo, Japan in 2005. His current research interests include the Application Layer Multicast. He is currently working for the M.E. degree in the Graduate School of Science and Engineering, Waseda University.

Jiro Katto received the B.S., M.E. and Ph.D degrees in electrical engineering from University of Tokyo in 1987, 1989 and 1992, respectively. He worked for NEC Corporation from 1992 to 1999. He was also a visiting scholar at Princeton University, NJ, USA, from 1996 to 1997. He then joined Waseda University in 1999. Since 2004, he has been a professor of the Department of Computer Science, Science and Engineering, Waseda University and has also been a director of the Electronic and Information Technology Development Department, New Energy and Industrial Technology Development Organization. His research interest is in the field of multimedia communication systems and multimedia signal processing. He received the Best Student Paper Award at SPIE's conference of Visual Communication and Image Processing in 1991, and received the Young Investigator Award of IEICE in 1995. He is a member of the IEEE, ACM, IPSJ and ITE.

Sakae Okubo received the B.E and D.E degrees at Hiroshima University in 1964 and Waseda University in 1996, respectively. He has been engaged in research, development and international standardization of video coding and audiovisual communication systems at NTT, ASCII Corp., TAO and Waseda University where he is currently visiting professor at Global Information and Telecommunication Institute. He is also a Member of ITE, IIEEJ and Senior Member of IEEE.