Tree-Based Application Layer Multicast using Proactive Route Maintenance and its Implementation

Tetsuya Kusumoto, Yohei Kunichika[1], Jiro Katto, and Sakae Okubo
Graduate School of Science and Engineering, Waseda University
3-4-1 Okubo
Shinjuku-ku, Tokyo, 169-8555 Japan
81-3-5286-3393

{kusumoto, yohei, katto}@katto.comm.waseda.ac.jp, sokubo@waseda.jp

ABSTRACT

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 children start searching for a new parent. 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 new parent over the overlay trees before the current parent leaves. A proactive approach allows a node to find its new parent immediately and switch to the backup route smoothly. In our proposal, the structure of the overlay tree using a redundant degree can decide a new parent 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.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design; C.2.4 [**Computer-Communication Networks**]: Distributed Systems

General Terms

Design, Experimentation, Performance

Keywords

Application Layer Multicast, Redundant Overlay Tree, P2P Streaming, 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]-[16]. 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 to receive the data flow again after a node departure is important for multicast applications such as live media streaming, because all the children nodes are disconnected. It is therefore quite important to maintain the media quality by quickly reconstructing the overlay trees, 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 node. It usually takes several seconds to restore the overlay tree. It is therefore important to find an effective mechanism to reconstruct the overlay trees.

On the other hand, a 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) [12], 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 [13], 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 children on the tree it is willing to

P2PMMS'05, November 11, 2005, Singapore.

[1] Yohei Kunichika is currently with RICOH Corporation, Japan

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 children 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 children do not have enough residual degrees to ensure their backup routes, the parent node employs the residual degrees of grandchildren nodes and below in calculating until they can finally ensure backup routes. 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 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 children) and children nodes. It means that our proposal does not generate much overhead to ensure backup routes. 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 degrees 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 of ALM protocols and the problem description of this paper. Section 3 provides our proposal in detail. Section 4 presents the simulation and implementation results. Finally, Section 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 is used to compute and instruct the construction of the delivery tree based on the information of metrics (e.g. distances, degree bounds) provided by the overlay members. This information is exchanged between nodes. Such a measurement technique often consumes a lot of bandwidth. This type of mechanism exchanges information with some hosts constantly and is called Mesh protocol. There are also Narada [3] and Scattercast [4] known as Mesh-first protocol. The Narada protocol keeps state about all other members that are part of the group. This information is also periodically refreshed. Distribution of such state information about each member to all other members leads to relatively high control overhead. The Scattercast protocol builds a routing table using a protocol called Gossamer for neighbor discovery in environment with multicast proxies. As most mesh protocols require each member to estimate distance to all or a large number of the members, they are not suitable for large scale applications.

In contrast, Yoid [5], Overcast [6] and Peercast [7] are distributed tree based protocols for larger groups. Our proposal is also a tree based protocol. This constructs a shared data delivery tree first. Packets are transported from the source node to its children and from the children to the grandchildren and below in order. In some methods, each member discovers a few other members of the multicast group that are not its neighbors on the overlay tree and establishes and maintains additional control links to these members after tree construction. The Yoid protocol incorporates loop detection and avoidance mechanisms when members change parents in the tree. If there is a loop path in the tree, the data streaming consumes away the bandwidth. The Overcast protocol targets creating high bandwidth channels from one source to receivers. It may not consider the latency, but minimizing tree depth reduces buffering delays. The Peercast protocol considers join and leave algorithm. It uses the round trip time method in join and the grandfather method in leave. A node in tree based protocols does not make as many connections as in mesh protocols. Tree construction of our proposal is based on the Peercast algorithm. Additionally our proposal considers robustness against node leaves and failures.

OMNI [9] defines a local transformation for the overlay tree to minimize the average latency of the entire hosts with degree constraints. Local transformation occurs between nearby nodes on the overlay tree periodically, and each host uses probabilistic transformation to optimize the overlay tree as a whole. In this paper, we do not consider dynamic tree reconstruction. However we use a round trip time as a metric in the tree construction, thus our proposal constructs a low delivery latency tree to some extent. ZIGZAG [8] and NICE [14] use a hierarchical cluster-based approach to construct overlay trees. Both of them use cluster leaders to manage the clustered overlay structure. The ZIGZAG protocol avoids network bottlenecks and keeps end-to-end delay lower. The hierarchy of the NICE protocol is for scalability to large groups. Our proposal does not adopt a hierarchical structure, but can adopt it along with the recent works.

There is a method using both delay and bandwidth as a metric, which places more emphasis on bandwidth and less on delay [12]. Scheme [13] selects some low delay nodes first, and selects the node in those so that the bandwidth can be used most efficient.. Our proposal does not use bandwidth as a metric in tree construction but our main purpose is holding backup routes proactively with low overhead.

The problem caused by node failures in overlay multicast has been recognized in more recent work. Peercast uses a reactive approach to deal with node leaves or failures in overlay multicast. It finds appropriate places in the subtree of the grandparent or the root for the affected nodes after failure happens. The time to find an appropriate place may be long and those affected nodes may even compete with each other to connect to other nodes. Our proposal differs in that each node has its backup route before node departures, so the time to find its appropriate place after node departure can be reduced. We will describe the difference

between a reactive approach and a proactive approach in detail in Section 2.2 and 2.3.

PRM [10] uses a proactive method for overlay multicast. It constructs a tree first. Randomized forwarding of this method enables fast recovery from failure of overlay nodes. A node in this method constantly sends packets for Randomized forwarding. Once a node in our proposal gets its backup route, it does not send packets for holding backup route unless the neighbors have concern with changes of the overlay topology. Another proactive method [11] uses a backup parent. It decides the backup parent before node departures happens. When the node departure happens, affected nodes receive data from the backup parents. Our proposal achieves holding a backup route of each node with lower overhead. We will describe the difference between our proposal and other proactive approaches in Section 2.3 and 3.

Multiple Description Coding (MDC) is used in Split Stream [15]. This splits a media stream into multiple stripes, and using separate multicast trees 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 these 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 children. Affected nodes cannot receive a data temporally 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 children nodes send a heartbeat packet to each other periodically. When the children nodes fail to receive heartbeat packets from the parent node over a period of time, the children nodes figure the parent node as a failure. However, the children nodes need a timeout period to recognize the failure. They cannot receive data flow all that time. Peercast proposes several recovery processes after a node departure, as listed below.

- Root

When a node leaves the tree or fails, each of its children tries connecting to the root. The subtree rooted at each of its children is maintained. Only children of the departed node rejoin the root. The root will try to accommodate them. The root accepts them as long as its degree does not become its max. If degree of the root is exhausted, the root will redirect some or all of them to its descendant. This redirection algorithm is also used in other recovery processes.

- Root-All

When a node leaves the tree or fails, all its descendants contact the root.

- Grandfather

Like the Root, when a node leaves, the children of the departure node contact the notified grandfather. When a node fails, the children contact the root node because the children cannot receive a message about their grandfather from their parent.

- Grandfather-All

When a node leaves the tree or fails, all its descendants contact the grandfather.

In these methods, it has been shown that the grandfather approach is most efficient. We therefore choose Peercast algorithm with the grandfather process as a comparison with our proposal.

The main task in this paper is reconstructing the tree by finding a new parent for each affected child as fast as possible when node departures happen. However, especially in the node failure phase, it takes long time to find a new parent because each affected node connects to its new parent by contacting the root in the tree, and the root might be quite far from the affected node. Furthermore, when the degree of upper layer nodes of the tree is exhausted, the redirection operation has to be repeated and might reach the node located at the lowest layer. In addition to taking a long time, redirection generates extra packets. If the number of children of a departed node is large, obviously the grandfather will not be able to accept all the children, so redirections will happen. Therefore, it is inevitable that it takes a lot of time to find a new parent in the reactive approach.

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 interruption time than that of the reactive approach.

PRM proposed a proactive approach with randomized forwarding in reconstructing an overlay tree when nodes departure happens. In PRM, each overlay node chooses a constant number of other overlay nodes at random and forwards data to each of them with a low probability. Randomized forwarding seems to be effective in some situations, but this scheme may generate some overhead traffic to send packets at random constantly.

In Yang's proactive approach [11], each non-leaf host calculates a backup parent for its children. 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 children can form a backup route.

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

A node in multicast session has n children $\{ c_0, c_1, \cdots, c_{n-1} \}$. $d(C_j)$ 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 children nodes. First, a parent node calculates residual degrees of the children. If the total residual degree of the children is not less than n-1, all its children can form their backup routes. If not, the children cannot. In this case, the parent node calculates the total residual degree including the residual degree of descendants of the children. 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 which holds the backup route supplies a backup route to the other children. Then, the descendants of the child and the child measure the latencies to the

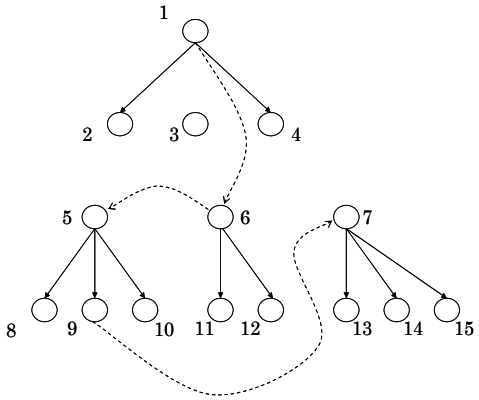

Fig.1. Finding a backup route in Yang's approach

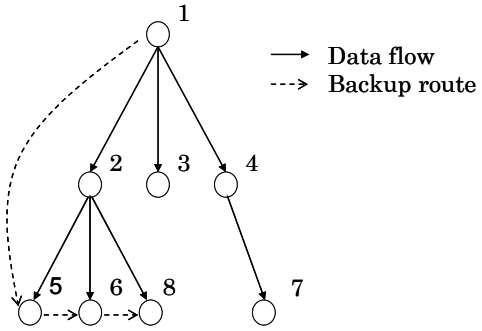

Fig.3. Finding a backup route in our proposal

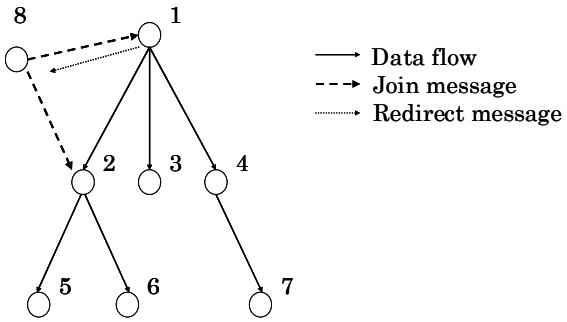

Fig.2. New node participation process in our proposal

other children, and the smallest edge is selected. This operation is repeated until the all children 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 children are node 5, 6 and 7. The maximum degree of each host is 3. The sum of the residual degree of them is less than (n-1), where n = 3 in this case, so their total residual degree is less than 2. They cannot form their backup routes among them. Then node 3 finds the descendants of its children to make the total residual degree larger than or equal to 2. When the total residual degree of the children and the grandchildren become larger than or equal to 2, the children can form their backup route. In Fig.1, node 6 has the smallest latency to grandparent 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 children are formed by using the residual degrees of the children and grandchildren. However, searching the residual degrees does not always finish in the children and grandchildren. When this operation continues in lower layer, it seems 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 host exhausting its degree. Each host constantly has residual degrees not less than 1. We apply the word a redundant overlay tree to this overlay tree. The children of each node can ensure their backup route between the grandparent and them by using their residual degree. This simplifies backup route calculation and contributes to overhead reduction. We show our proposal in detail below.

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.

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 children list. Node 2 sends the children 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 children list of node 2 includes node 8 only (i.e. no other children exist), node 2 immediately informs node 8 that node 1 is a backup parent of node 8.

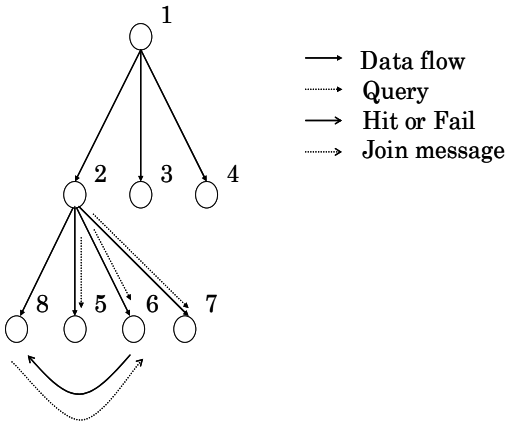

Fig.4.Reconstruction of a redundant tree

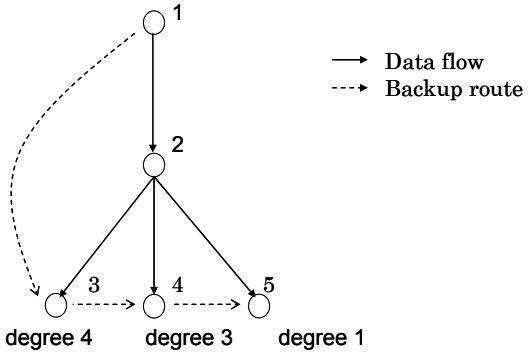

Fig.5. Treating a node of one or zero maximum degree

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 children layer of the departure node. It never goes down to calculate in the lower layers dissimilar to the previous approach.

In some rare cases, a node cannot use its backup route. 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. Our approach also uses the same method in such cases. In the method, when a node connects its backup parent node and the backup parent node does not reply, it uses the ancestor list. First, it ordinarily joins the grandparent and it follows the redirection algorithm whether the grandparent accepts the node or not. When the grandparent does not exist because the grandparent 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.

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 host exhausts its degree by accepting a node rejoining in the backup route procedure. When this happens, a tree reconstruction procedure is invoked by the host itself in order to keep the route redundancy. This procedure is carried out by asking the children 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 degree, the newly connected node moves to the node that has the most residual degree. We show the procedure in Fig.4. Node 2 uses up its degree because node 8 joined node 2 as its backup route. Node 2 sends a query to other children, which are nodes 5, 6 and 7, and they reply hit or fail messages to node 8. The hit message means it can accept join. The fail message means it cannot accept. Node 8 moves to the node which has sent the hit message first. In Fig.4, node 6 sends a hit message to node 8, and node 8 joins node 6. If all messages of the children 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.

One question in our proposal is that there are the nodes whose maximum degrees are zero or one. Existence of nodes with zero degree (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 reserved one degree in each node. A node of one degree can not have a child node. In the case that the maximum degrees of the all children of a node are one, our proposal cannot construct a subtree rooted at the children, so the tree can not be constructed effectively. In the worst case that the maximum degrees of the all children 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 exhausting their one maximum degree or zero maximum degree. 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.5. Node 2 is a parent of three children, 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 children nodes can get their backup routes.

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 grandfather policy described in Section 2.2. In simulations, we also compare our method with Yang's method, which is another proactive method

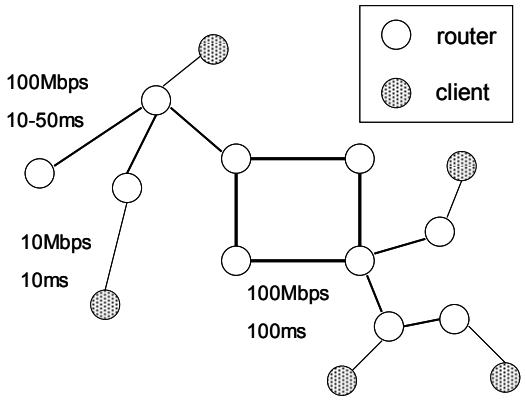

Fig.6. Simulation topology

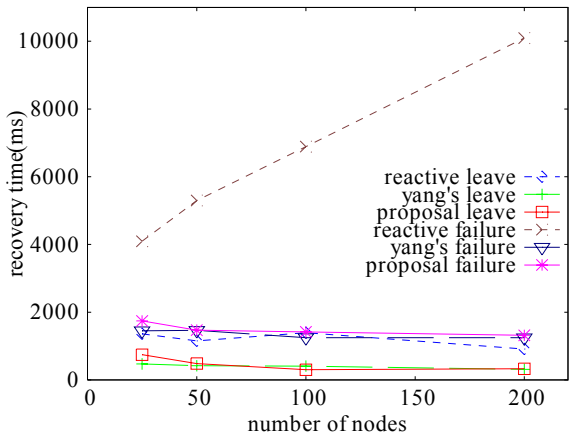

Fig.7. Average recovery time with varying number of nodes in simulation

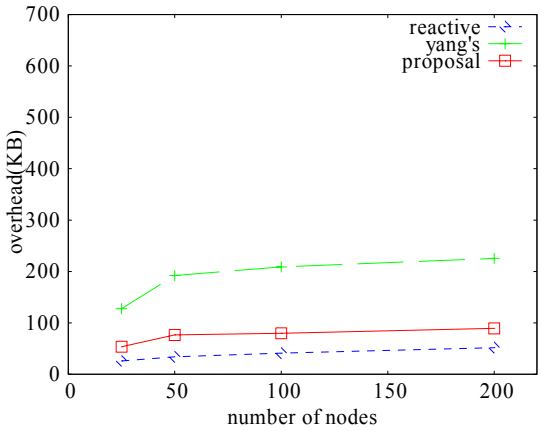

Fig.8. Overhead of the round robin method with varying number of nodes in simulation

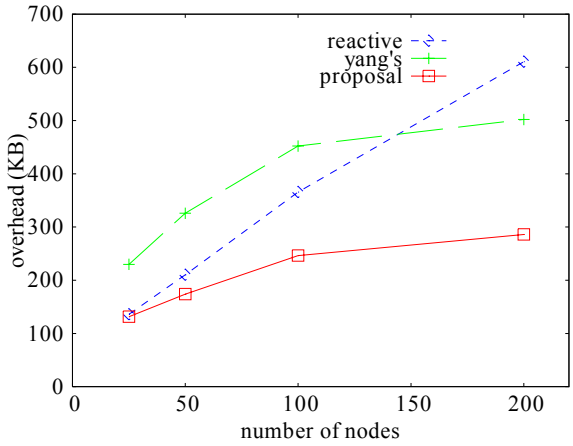

Fig.9. Overhead of the round trip time method with varying number of nodes in simulation

described in Section 2.3. We show simulation results in Section 4.1 and implementation results in Section 4.2.

4.1 Simulation Results

We carried out the simulations by ns-2[17]. We show simulation results in Figs 7, 8, 9, 10.and 11. Our simulation topology has 24 routers. Four routers of them are domain-to-domain routers as shown in Fig.6. The others are set up at random between end-hosts. The distance between two end-hosts is the sum of link delays on the shortest path between them. The delay and bandwidth between the domain routers is 100ms and 100Mbps. The delay between the routers in a domain varies from 10 to 50 ms and the bandwidth is 100Mbps. The delay and bandwidth between a router and end-hosts are 10ms and 10Mbps. The nodes randomly connect 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 10ms to 100ms. The maximum degree of each node varies from 1 to 6. For the experiment in Fig.11, we fixed the degree of each host at a particular value. The overlay tree is constructed first by all hosts, and then nodes randomly join and leave the overlay tree every 15 seconds for 300 seconds.

4.1.1 Comparison of Recovery Time

First, we use the average recovery time as a performance measure. It is the average time for an affected node to find a new parent. Fig.7 plots the average recovery time of leave and failure in simulations. Each node send 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.

In Fig.7, the average recovery time against node leaving in the reactive approach is about 1300ms in each number of nodes. The average recovery times against node leaving in proactive method (our proposal and Yang's approach) are less than about half of the reactive method, about 500ms. In case of node failures, 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 1400ms.

The proactive methods enable the affected nodes to immediately connect to their backup routes. This is common to both proactive

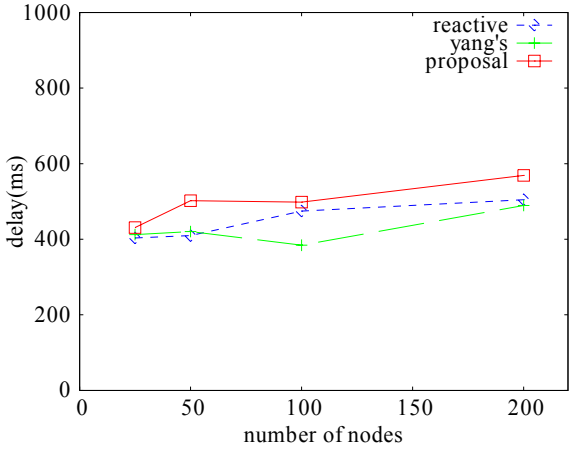

Fig.10. Average delivery delay with varying number of nodes in simulation

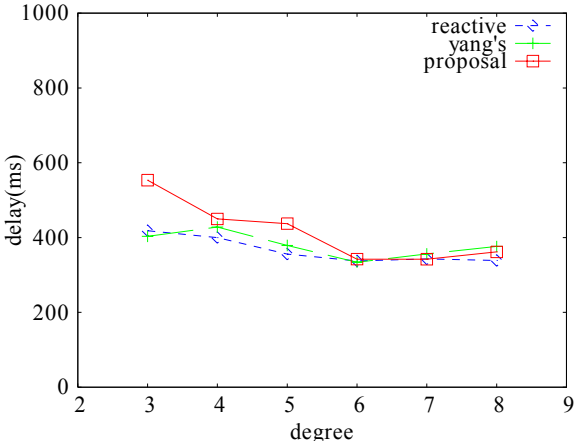

Fig.11. Average delivery delay with 200 nodes with varying number of degree

methods, so their results are nearly equal. On the contrary, in the reactive approaches, 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 in the reactive approach. As the number of nodes increases from 25 to 200, the recovery time of the reactive approach increases. This is because the height of an overlay tree becomes bigger, and many redirections happen.

4.1.2 Comparison of Control Overheads

We show the overheads of the reactive approach, Yang's approach and our proposal. The overhead is a total number of control packets to maintain the overlay tree.

For the reactive approach, the control overhead comes from the control messages exchanged for the affected nodes to find new parents. 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 children in 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 children 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.

For the proactive method, the control messages consist of two parts. 1) Similar to reactive approaches, control messages are exchanged for the children 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.

Fig.8 compares the overheads of the round robin method with redirection. The number of nodes varies from 25 to 200 in simulations. In Fig.8, 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 packets.

Fig.9 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 redirection is required. Redirection generates a volume of overheads to measure RTT.

As shown in Fig.8 and Fig.9, our proposal does not generate packets as many as Yang's approach for holding backup routes. Holding a backup route operation of our proposal needs fewer packets than Yang's approach. 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 be larger. In Fig.8, overheads of Yang's approach and our proposal are larger than the reactive approach, but in Fig.9, 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. 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 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 a backup route search and contributes to overhead reduction. However, that structure causes the height of the overlay tree to be larger and possibly

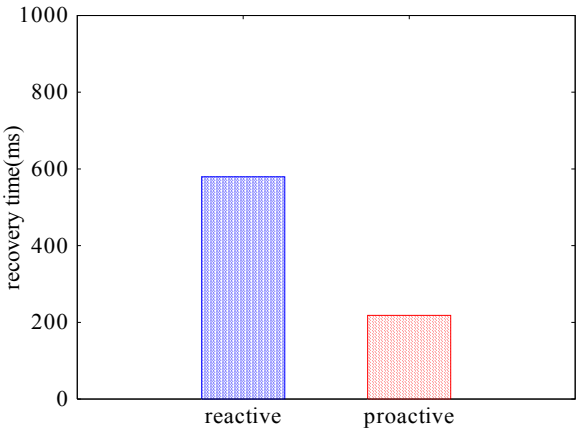

Fig.12. Average recovery time with 25 nodes in implementation

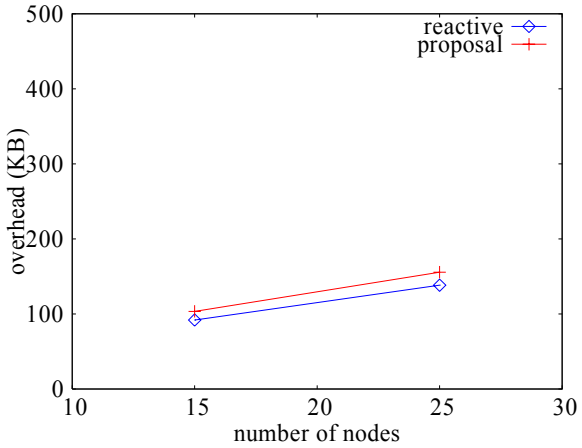

Fig.13. Overhead of the round robin method with 15 and 25 nodes in implementation

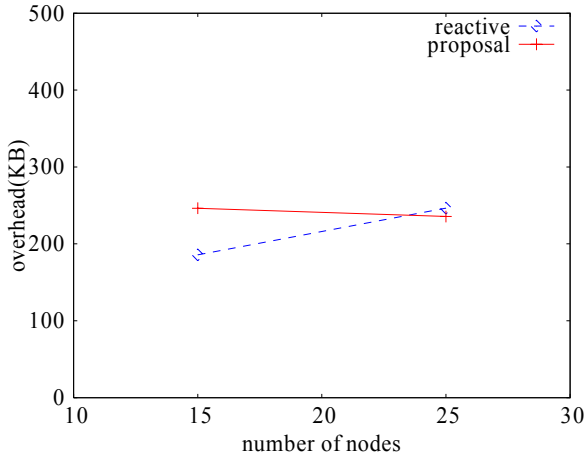

Fig.14. Overhead of the round trip time method with 15 and 25 nodes in implementation

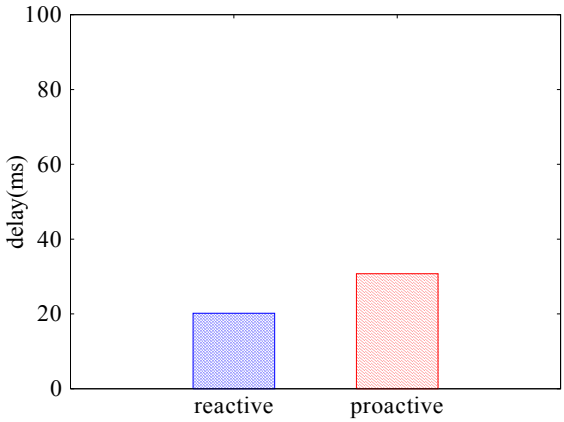

Fig.15. Average delivery delay with 25 nodes in implementation

leads to delay increase overall, because all nodes do not use their degree full. Therefore, an obvious problem of our approach is increase in data delivery delays. Fig.10 shows how the average transfer latency in the tree varies the number of nodes from 25 to 200 in simulations.

In Fig.10, we can see that the average delivery delay of our proposal is larger than other methods. This means that hop counts increase in our proposal. Next, we show an interesting result in Fig.11. Maximum degree of all nodes is fixed at the same number when the number of nodes is 200. Fig.11 shows the average delivery delay in each maximum degree. 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 transfer latency of our proposal is about 380ms 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. They lead to reduction of delay in the resilient overlay structure. 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.

4.2 Implementation Results

In addition to simulations, we implemented the prototype 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, so we have to consider the backbone with low bandwidth in future work. We will describe about this in Section 5. Firstly in the experiment, the source node waits for joining nodes. After the source node receives a join message, it starts sending the data flow of a media, 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 children. All nodes join the ALM session, and then each node joins or leaves randomly for 30 minutes. We show implementation results in Figs.12, 13, 14 and 15.

4.2.1 Comparison of Recovery Time

In Fig.12, we show the average recovery time of 25 nodes in implementation. Recovery time of our proposal is less than half of the reactive approach. 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 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

Fig.13 and Fig.14 represent the overheads when the number of nodes is 15 and 25 in the implementations. In Fig.13, we used the round robin method with 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.14 shows that overhead of our proposal is almost the same as the reactive approach at 25 nodes. We used the round trip time method with redirection. As the number of nodes increases, overhead of the reactive method increases. We can also see this trend in the simulation result of Fig.9.

4.2.3 Comparison of Data Delivery Delays

Fig.15 shows the average delivery delays in implementation when the number of nodes in session is 25. The delay of our proposal is more 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.

In summary, as shown in the simulations and implementations, we could confirm that our proposal can recover from node departures much faster than the reactive approach. Especially, we confirmed that our proposal could continue to play media streaming smoothly in implementations. With regard to overheads, we could reduce them for maintaining backup routes, and our proposal always generates less overheads than Yang's approach. In the specific case, our proposal can achieve less overheads than the reactive approach. About the transfer delay, our proposal is more than other methods, but we did not feel critical difference between our proposal and the reactive approach in playback. We realize our approach can resolve the problems of node departures and overheads while maintaining backup routes efficiently.

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, the recovery time of our proposal is much faster 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 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. Although the data delivery delay tends to be larger 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 [18]. 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.

6. 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] Y. Chu, S. G. Rao, H. Zhang, "A Case for End System Multicast," in Proceedings of ACM SIGMETRICS 2000, June. (2000)

[3] D. Pendarakis, S. Shi, D. Verma, M. Waldvogel, "ALMI: An Application Level Multicast Infrastructure," 3rd USENIX Symposium on Internet Technologies and Systems, Mar. (2001)

[4] Y. Chawathe, S. McCanne, E. Brewer, "Scattercast: An Architecture for Internet Broadcast Distribution as an Infrastructure Service," PhD Thesis, University of California, Berkeley, (2000)

[5] P. Francis, "Yoid: Extending the Internet Multicast Architectuire," http://www.icir.org/yoid/

[6] J. Jannotti, D. Gifford, K. Johonson, M. Kaashoek, J. O'Toole, "Overcast: Reliable Multicasting with an Overlay Network," 4th Symposium on Operating Systems Design & Implementation, Oct. (2000)

[7] H. Deshpande, M. Bawa, H. Garcia-Molina, "Streaming Live Media over Peers," Technical Report 2002-21, Stanford University, Mar. (2002)

[8] D. Tran, K. Hua, T. Do, "ZIGZAG: An Efficient Peer-to-Peer Scheme for Media Streaming," in proceedings of IEEE INFOCOM 2003, Apr. (2003)

[9] S. Banerjee, C. Kommareddy, K. Kar, B. Bhattacharjee, S. Khuller, "Construction of an Efficient Overlay Multicast Infrastructure for Real-time Applications," in proceedings of IEEE INFOCOM 2003, Apr. (2003)

[10] S. Banerjee, S. Lee, B. Bhattacharjee, A. Srinivasan, "Resilient multicast using overlays," in proceedings of ACM SIGMETRICS 2003, June. (2003)

[11] M. Yang, Z. Fei, "A Proactive Approach to Reconstructing Overlay Multicast Trees," in proceedings of INFOCOM 2004, March. (2004)

[12] Y.Chu, S. G. Rao, S. Ses, H.Zhang, “Enabling Conferencing Applications on the Internet using an Overlay Multicast Architecture” in proceeding of ACM SIGCOMM 2001, Aug. (2001)

[13] S. Y. Shi, J. S. Turner, M. Waldvogel, “Dimensioning Server Access Bandwidth and Multicast Routing in Overlay Networks” in proceeding of NOSSDAV 2001, June. (2001)

[14] S. Banerjee, B. Bhattacharjee, and C kommareddy, “Scalable application layer multicast,” in proceedings of ACM SIGCOMM 2002, Aug. (2002)

[15] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, “Splitstream: high-bandwidth multicast in cooperative environments” in proceeding of SOSP 2003, Oct.(2003)

[16] Bram Cohen, “Incentives Build Robustness in BitTorrent” 2003. http://bittorrent.com/bittorrentecon.pdf

[17] The Network Simulator ns-2, http://www.isi.edu/nsnam/ns

[18] The Network Emulator Nist Net, http://snad.ncsl.nist.gov/itg/nistnet/