Proactive Route Maintenance and Overhead Reduction for Application Layer Multicast

Tetsuya Kusumoto, Yohei Kunichika †, Jiro Katto and Sakae Okubo

Graduated school of Science and Engineering, Waseda University
3-4-1 Okubo, Shinjuku-ku, Tokyo, 169-8555 Japan
E-mail: {kusumoto, yohei, katto}@katto.comm.waseda.ac.jp, sokubo@waseda.jp

***Abstract*— The purpose of this study is to maintain efficient backup routes for restoring overlay trees. 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 find a new parent over the overlay trees before the current parent leaves. A proactive approach can find respective new parents immediately and switch to the backup route smoothly. In our proposal, the structure of the overlay tree using a redundant degree enables to decide a new parent without so much overhead information. 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.**

***Index Terms*— Application Layer Multicast, Redundant Overlay, P2P Streaming, Proactive Route Maintenance**

I. Introduction

ALM (Application Layer Multicast) implements the multicast functionally at end-hosts. 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 restore 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 leaves freely and may fail sometimes. This does not happen 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. 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 restore 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) [13], 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 is proposed by Yang et al [14], which we call Yang's approach in this paper. It calculates the number of degrees each host has, and ensures backup route proactively whenever a node leaves or joins. It is inevitable to consider the degrees constraint in overlay multicast, which can be easily observed in streaming applications. For example, assume the bit rate of media is B and the bandwidth of the connection of an end host is $bi.$ The total number of streams it can have is $[bi / B]$, so the degree represents the total number of connections that a node can establish. This calculating process generates extra data overheads and is not scalable. Volume of control traffic can be significant for 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 degree in each host, backup routes can be always established among the grandparent and children nodes. 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 number of degrees (fanouts) increases. Furthermore, we implemented our proposal in software, and experimented with

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

Proceedings of the Joint International Conference on Autonomic and Autonomous Systems
and International Conference on Networking and Services (ICAS/ICNS 2005)
0-7695-2450-8/05 $20.00 © 2005 IEEE

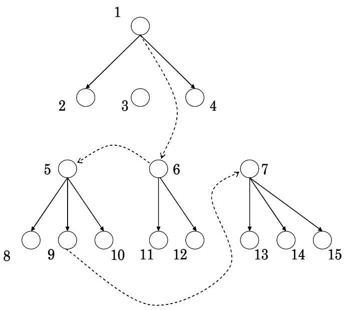

Fig.1. Finding a backup route in Yang's approach

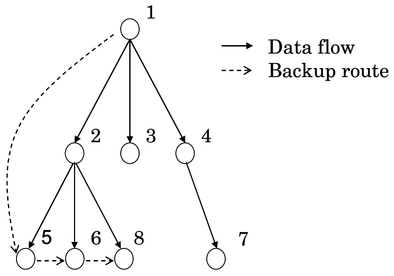

Fig.2. Finding a backup route in our proposal

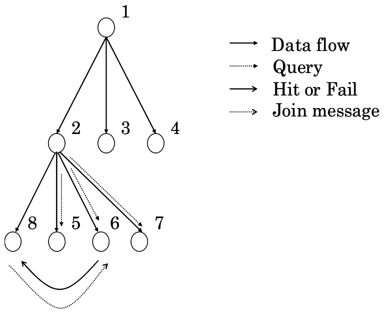

Fig.3. Reconstruction of a redundant tree

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 Ⅲ provides our proposal in detail. Section Ⅳ presents the simulation and implementation results. Section V describes related work and Section Ⅵ concludes the paper.

II. An Overview of ALM Protocols and Problem Description

A. *Overview of ALM Protocols*

Most ALM protocols have focused on how to construct an efficient multicast tree, but the problem of dealing with node failures in ALM has been recognized in more recent works. Peercast [7] uses a reactive approach to deal with node departures or failures. It finds appropriate places in the subtree of the grandparent or 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. PRM [12] uses a proactive approach for overlay multicast. It uses randomized forwarding, which enables fast recovery from failure of overlay nodes. Another proactive approach [14] uses backup parents. It decides the backup parent before node departures happens. When the node departure happens, affected nodes receive data from the backup parents.

B. *Reactive Approach*

Most of ALM protocols 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 affected by its leaving such as its parent and children. When a host suddenly fails, it cannot send a message, and the affected nodes will not notice the failure for a while. A heartbeat mechanism helps the affected node to notice the failure by checking a connected node periodically by sending heartbeat messages to each other. If a node does not receive heartbeat messages from a connected node for a while, it assumes the connected node fails. In the failure case, however, the affected nodes need a timeout period to recognize the failure, during which it cannot receive data flow.

We use example of Peercast [7] for comparison purpose as a reactive approach. It proposed several recovery processes after a node departure, *Root, Root-All, Grandfather and Grandfather-All.* In these methods, it has been shown that the grandfather approach is most efficient, in which each of its children receives information of the grandfather from the departed node and contacts the grandparent when a node leaves the tree. Subtree rooted at each of its children is maintained. If its degree is exhausted, the grandfather will redirect them to its descendant. When a node fails, the children contact the root node because the children cannot recognize their grandfather. Therefore, in the reactive approach, it is inevitable that it takes a lot of time to find a new parent.

C. *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, so affected nodes can receive data flow with reduced interruption time.

In Yang's proactive approach [14], each non-leaf host calculates a backup parent for its children. Each host uses (1) to figure out if all its 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 degree of the children nodes. A node calculates residual degree of the children. If the total residual degree of the children can meet (1), all its children can form backup routes. If not, the node calculates the total residual degree including the residual degree of descendants of the children. In Fig.1, we outline the algorithm of Yang's proactive approach to form a backup route. We show the children of node 3 forming a backup route. Children of the node 3 are node5, 6 and 7. They have the total residual degree less than (n-1), where n = 3 in this case, so they have the total residual degree less than 2. They cannot form backup route in children layer. If it is not large enough, node 3 checks those descendants of its children to make the total residual degree larger than or equal to

2. In this case, they can form backup route by calculating the residual degree of the grandchildren, but if the grandchildren also do not have enough residual degree, it calculates the residual degree of descendant in lower layer. This operation generates many packets similar to the Peercast case.

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

III. PROACTIVE ROUTE MAINTENANCE OVER REDUNDANT OVERLAY TREES

In our proposal, 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 that residual degree. This simplifies backup route calculation and contributes to overhead reduction. We show our proposal in detail below.

We show how to calculate a backup parent in our proposal in Fig.2. When node 8 connects to node 2 as a child, node 2 updates its children list. When node 2 leaves, node 1 cannot accommodate all the children of node 2 due to its degree constraint since node 2 has three children. Therefore, node 2 sends the children list to node 1. Node 1 measures a round trip time to each grandchild, and informs a node having the smallest round trip time (the fastest node) to become its backup route. In Fig.2, if the fastest node is node5, node 5 has a back up route to node 1. The second fastest node has a backup route to the fastest node, and node 6 has a node 5 as its backup parent. The slowest node 8 has second node 6 as a backup parent.

Note that layer of the backup route calculation is required only at the children layer of the departure node. It never goes down to the lower layers dissimilar to the previous approach.

In some rare cases, when current parent departure happens, backup parent could leave or fail at the same time. Parent departure could happen before calculation for backup route calculation finishes. In [14], handling those cases is shown. It uses the ancestor-list from grandparent to root. When a node connects to its backup parent node and the backup parent node does not reply, it uses the ancestor list. First, it ordinarily joins the grandparent. If the grandparent degree is not exhausted, the grandparent accepts the node. The grandparent which does not have enough residual degrees redirects the node to its children. When the grandparent does not exist because of departure at the same time, 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 recover the route redundancy. This procedure is carried out by asking the children of backup route node except the newly connected node whether their degree is exhausted. At the time newly connected node finds that a certain node has residual degree, the node moves to the node which has residual degree. We show the procedure in Fig.3. Node 2 uses up its degree because node 8 joined node 2 as its backup route. Node 2 sends a query to other children and nodes 5, 6 and 7 send 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 sent the hit message first. In Fig.3, node 6 sends a hit message to node 8, and node 8 joins node 6. If all messages of the children are fail, newly connected node joins the node which sent a message first although degree of the node is exhausted, and receives a redirection message from the first node.

One question in our proposal is that there are the nodes which have equal or less degree than 1. 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 Bit Torrent [15]. Handling of the nodes which have one degree is a specific problem in our proposal, because we construct the redundant overlay tree by forcing reserved degree in each node. If a node of one degree connects to another node of one degree, in the worst case that all children have only one degree, our proposal cannot construct a subtree rooted at the children. To avoid this case, we firstly allow the nodes of one degree to have a child although their degree is 1. This causes another problem that they cannot provide backup routes because of exhausting their degree. We then decide that the number of nodes of one degree which each node can have is only one, and place the node of one degree at the end of the backup spanning tree, so the node of one degree need not provide backup route at the end of the backup spanning tree. In Fig.2, we place the one degree node on the node 8 place. Node 8 need not provide backup route.

IV. 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 restored and how small the control overheads can be kept by redundant backup routes. We compare our proactive scheme with a reactive scheme uses grandfather policy described in Section Ⅱ. In simulations, we also compare our scheme with Yang's method, which is another proactive scheme proposed in [14].

A. Simulation Results

Our simulation topology has 24 routers. Four routers of them are domain-to-domain routers. Nodes randomly connect to one of the 20 routers except the four inter domain routers. The number of hosts varies from 25 to 200. The link latency varies from 10ms to 100ms. The degree of each host varies from 1 to 6. For one of the results, we fix the degree of each host. The overlay tree is constructed at once in all hosts, and then nodes randomly join and leave the overlay tree every 10

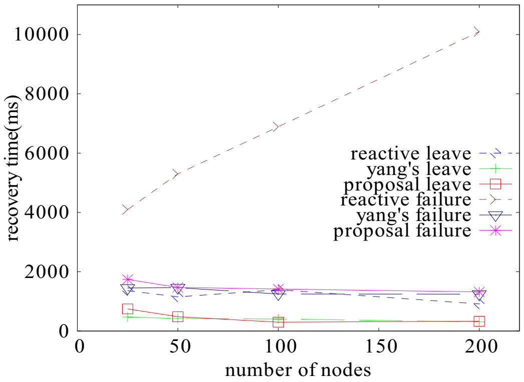

Fig.4. Average recovery time with varying number of nodes in simulation

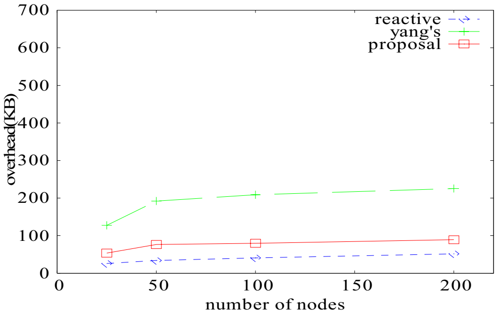

Fig.5. Overhead of the round robin method with varying number of nodes in simulation

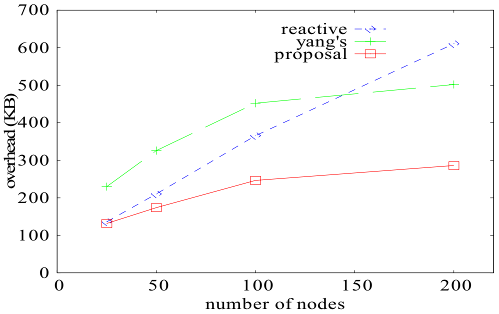

Fig.6. Overhead of the round trip time method with varying number of nodes in simulation

seconds. We show simulation results in Figs. 4, 5, 6, 7 and 8.

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. In failure case, we set the time of deciding node failure for one second with heartbeat messages. If a node does not receive any heartbeat messages from its connected nodes to one second, it decides the nodes became failure.

In Fig.4, 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 approach, 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.

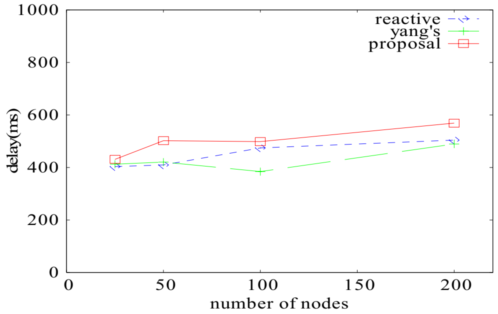

Fig.7. Average delivery delay with varying number of nodes in simulation

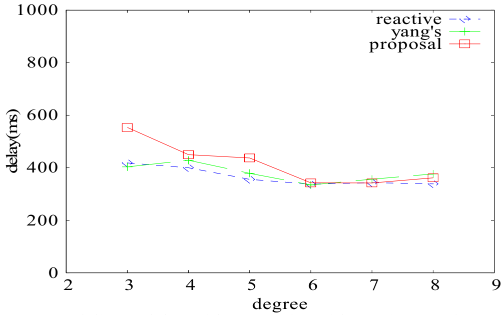

Fig.8. Average delivery delay with 200 nodes with varying number of degree

The proactive methods enable the affected nodes to immediately connect to their backup parents. This is common to both proactive 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 will be accepted. Especially in the node failure cases, affected nodes have to contact 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.

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. Control packets represent all signaling packets.

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 full receives a join message, the node redirects the message to their children in order. In the round trip time method, the redirected node receives a children list, and sends a join message to a node of the smallest RTT by measuring RTT to each child. 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.5 compares the overheads of the round robin method for

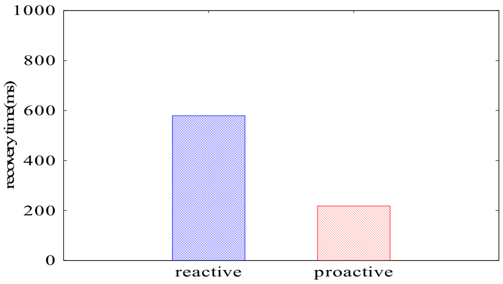

Fig.9. Average recovery time with 25 nodes in implementation

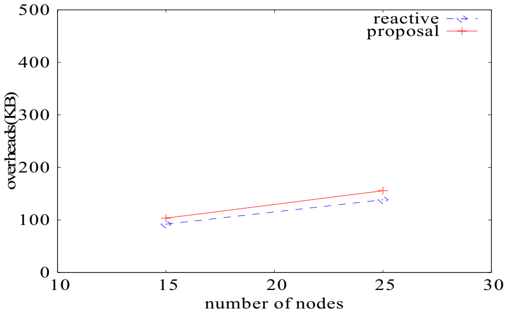

Fig.10. Overhead of the round robin method with 15 and 25 nodes in implementation

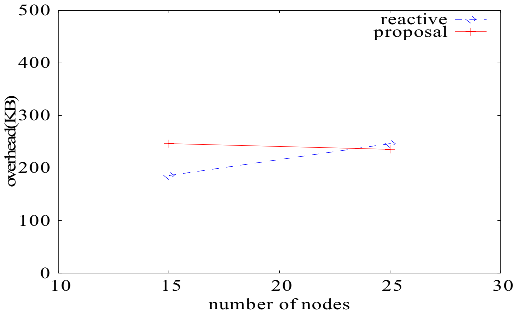

Fig.11. Overhead of the round trip time method with 15 and 25 nodes in implementation

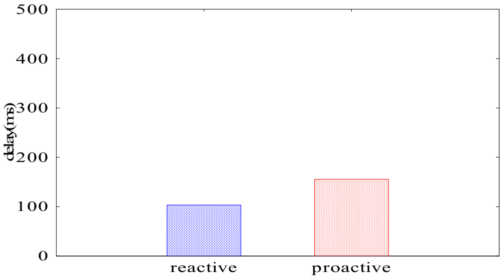

Fig.12. Average delivery delay with 25 nodes in implementation

redirection when we varied the number of nodes from 25 to 200 in simulations. In Fig.5, we can see Yang's proactive approach generates higher overhead. In comparison with Yang's approach, our proactive proposal 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.6 compares the overheads of the round trip time method for redirection, when we varied the number of nodes 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.

By Fig.5 and Fig.6, we can think 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. In most ALM protocols, each node joins the overlay tree following their metric, so exchanges a lot of information to optimize the overlay tree in join and redirection process. ALM is used in media streaming, so 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.

3) *Comparison of Data Delivery Delays*

Proposed 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 leads to delay increase overall, because all nodes do not use their full degree. Therefore, an obvious problem of our approach is increase in data delivery delays. Fig.7 shows how the average transfer latency in the tree varies from 25 to 200 in simulations.

In Fig.7, we can see that latency of our proposal is larger than other methods. This is because the overlay tree of our proposal tends to be higher due to not using the full degree. This means that hop counts increase in our proposal. Next, we show an interesting result in Fig.8. Degree of all nodes is fixed at the same number when the number of nodes is 200. Fig.8 shows the average transfer latency in each 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 the 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.

B. *Implementation Results*

In addition to simulations, we implemented the reactive approach and our proposal in real network. We developed those methods on PCs. Video codec is ITU-T H263+. Total 25 nodes are deployed over three different networks. Each network connects to backbone in Japan. Firstly, all nodes join the ALM session, and each node joins or leaves randomly for 30 minutes. We show implementation results in Figs.12, 13, 14 and 15.

1) *Comparison of Recovery Time*

In Fig.9, 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 feels like “freeze frame” for a moment, but in our proposal, decoded pictures continued to play smoothly.

2) Comparison of Control Overheads

Fig.10 and Fig.11 represent the overheads when the numbers of nodes are 15 and 25 in implementations. In Fig.10, 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.11 shows that overhead of our proposal is almost the same as the reactive approach at 25 nodes. We used the round trip 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.6.

3) Comparison of Data Delivery Delays

Fig.12 shows the average transfer latency in implementation when the number of nodes in session is 25. The latency 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.

V. Related work

ALM has been studied extensively in recent years. Most ALM protocol studies have focused on how to construct an efficient multicast tree. Basically, they can be classified into centralized and distributed approaches.

ALMI [2], Narada [3] and Scattercast [4] are Mesh-first protocols and employ centralized solution. These protocols require each member to estimate distance to all or a large number of the members.

In contrast, Yoid [5], Overcast [6] and Peercast [7] are distributed Tree-first protocols for larger groups. This constructs a shared data delivery tree first. 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.

Bayeux [8] and CAN-based multicast [9] utilize a P2P routing known as a distributed hash table (DHT) algorithm. OMNI [10] defines a local transformation for the overlay tree to minimize the average latency of the entire hosts with degree constraints. ZIGZAG [11] and NICE [12] uses a hierarchical cluster-based approach to construct overlay trees. This procedure avoids network bottlenecks and keeps end-to-end delay lower.

VI. 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, 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 even achieve less overheads 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. We confirmed our approach can resolve the problems of node departures and overheads while maintaining backup routes efficiently.

Acknowledgments

This research was supported in part by the NICT R&D project "Broadcast System Using Communication Network"

Reference

[1] S. Deering, “Host Extension for IP Multicasting,” RFC 1112, Aug.1989.

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

[3] Y. Chu, S. G. Rao, H. Zhang, “A Case for End System Multicast,” in *Proceedings of ACM SIGMETRICS 2000*, June. (2000)

[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] S. Zhuang, B. Zhao, A. Joseph, R. Katz, S. Shenker, “Bayeux: An Architecture for Scalable and Fault-Tolerant Wide-Area Data Dissemination,” *ACM NOSSDAV 2001*, June. (2001)

[9] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker, “Application-level Multicast using Content-Addressable Networks,” *In Proceedings of NGC* (2001)

[10] 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)

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

[12] S. Banerjee, B. Bhattacharjee, and C kommareddy, “Scalable application layer multicast,” in *proceedings of ACM sigcomm2002.*

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

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

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

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