Overlay Multicast Protocol for Delivering Layered Data Structure

Kohei Ogura*, Hideaki Imaizumi†, Masaki Minami‡, Osamu Nakamura‡ and Jun Murai‡

*Graduate School of Media and Governance, Keio University
5322 Endo Fujisawa Kanagawa, Japan
Email: koh39@sfc.wide.ad.jp
†Graduate School of Frontier Sciences, The University of Tokyo
7-3-1 Hongo Bunkyo-ku Tokyo, Japan
Email: imaq@k.u-tokyo.ac.jp
‡Faculty of Environmental Information, Keio University
5322 Endo Fujisawa Kanagawa, Japan

Abstract—This paper proposes an adaptive Overlay Multicast protocol for real-time group communication in a heterogeneous environment. Overlay multicast technology relies on end nodes by delegating the multicast functionality from routers in IP Multicast. Overlay multicast research has two major issues to consider: heterogeneous resource environment and instability of end nodes. In our protocol, data is divided into multiple layers using abstract layered data structure. The number of layers is used for the main metric to construct the multicast tree to satisfy the demand for end nodes resource environment. Furthermore, multi-path layer distribution method for fast recovery from multicast tree partition and congestion avoidance method are proposed to deal with the unstableness at end node.

I. INTRODUCTION

Demand for group communicating applications has rapidly increasing, due to rich environment at end user on both creating and distributing high-quality multimedia contents. Although there is a very large demand for such communication, end users are still not free to go for enjoying neither broadcast self-produced real-time comedy show nor videoconferencing with large group of people.

IP Multicast is the traditional method for group communication over the Internet that satisfies the demand. Though long time has elapsed since IP Multicast was initially proposed, IP Multicast still has both technical and policy issues such as inter-domain routing, diffusion of multicast capable routers, multicast address allocation, etc. to solve for wide area deployment [1].

Recently, Overlay Multicast, a substitute technology for IP Multicast, has become a hot topic for researchers. The basic idea of Overlay Multicast is to delegate multicast functionality such as data replication, group management, multicast routing from the IP layer to an upper layer, mostly the application layer. Fig.1 illustrates how each method bear the function of data replication, which is performed by routers in IP Multicast and by end nodes in Overlay Multicast.

Overlay Multicast sets aside the deployment issues on IP Multicast, which is a big concern. Since the idea of Overlay Multicast first appeared [2], a number of Overlay Multicast routing protocols have been proposed [3]–[7]. By delegating the multicast functionality to application layer, Overlay Multicast network relies on end nodes. This means every end node joining to the multicast group constructs and maintains the multicast tree to deliver the data. In such an environment, Overlay Multicast research has two major issues to consider: heterogeneous resource environment and instability of end nodes.

Each end node joining the multicast tree has a different resource environment such as link bandwidth and computing resource. Each node will request different quality of contents to satisfy their resource constraint. Multicast method should handle this request flexibly. At the same time, multicast method should not limit the node to join by its resource constraint.

Instability of end nodes should also be considered for constructing a stable multicast tree. A node failure will cause multicast tree partition, which stops the data transmission. From this reason, fast recovery method of multicast tree is required for reliable multicast tree.

This paper proposes an adaptive overlay multicast protocol called LOLCAST (Layered Overlay multiCAST) designed for group communication in a heterogeneous environment whose applications include real-time video-streaming. LOLCAST assumes the multicast group size as several hundreds. LOLCAST uses layered coding and layered data structure to adapt to the heterogeneity of end nodes. In addition, this paper introduces fast recovery method from multicast tree partition and congestion control method using the characteristic of layered data structure.
II. RELATED WORKS

This section represents a detail description and the drawbacks of exiting methods for adapting to the heterogeneity of end nodes in Overlay Multicast.

There are mainly two approaches proposed to solve this difficulty. Primary method is multiple version approach and the secondary method is multiple layer approach. Fig.2 illustrates comparison between data structure used in each approach. Example of data structure supports four different qualities of data.

A. Multiple version approach

In multiple version approach, multiple video data containing different qualities and bit-rates for a single content (from low quality to very high quality in Fig.2) is sent by the source node. Receiver node selects the stream which suits their resource environment, especially the network bandwidth. This approach is taken by End System Multicast [2].

B. Multiple layer approach

In multiple layer approach, layered video coding is the key technology. In layered coding, video data is divided into multiple layers, as illustrated in Fig.2. Data included in those layers are not overlapping each other. Layers are categorized into the base layer and enhancement layers. Base layer provides the minimum quality of original video data, and it is fundamental for decoding other layers. Enhancement layer provides additional data which improves video quality. Each layer has a dependency with layer directly below for decoding. Several layered coding methods have been proposed, including MPEG-2 scalable profile [8], MPEG-4 scalable profile [9], and H.263+ [10].

Multiple layer approach uses layered encoded data to adapt to heterogeneity. Source node sends the segmented video data with full layers, and receiver node acquires some number of layers to sustain their resource environment. This approach is taken by our previous work [7].

C. Advantages and drawbacks

As referred as above, multiple layer approach uses data structure consisting of multiple layers improving the quality of the content by increasing the number of layers. Multiple layer approach has an advantage in network bandwidth utilization compared to multiple version approach. Compared to multiple layer approach, multiple version approach needs a separate and overlapping data to support each quality level, which may overload the network bandwidth. Another advantage for multiple layer approach is that it could handle wide-range of requests for qualities very flexibly by just increasing the number of layers. One drawback for multiple layer approach is that layered coding uses complicated encoding method which requires more computing power.

III. OVERVIEW OF LOLCAST

This section introduces a new Overlay Multicast protocol LOLCAST and make an overview of the protocol. First, this section describes the data structure and definitions used in LOLCAST. Next, an example of a tree structure constructed by LOLCAST is illustrated. Last, this section describes recovery method from node failure and congestion avoidance.

A. Layered data structure

The basic idea of layered data structure is similar to multiple layer approach, in which data consists of multiple layers to support wide range of the amount of information. Layered data structure used in LOLCAST basically inherits multiple layer approach but can support not only layered coded data but also combined data from various types of data abstractly. The words ”base layer” and ”enhancement layer” will be used for describing layers in LOLCAST.

B. Definitions used in LOLCAST

In this section, we introduce the following definitions used in LOLCAST. These definitions will be used for explaining the functions of LOLCAST.

- **Layer encoded data** \( \{ l_0, l_1, l_2, ..., l_n \} \)
  This definition stands for each layered coded data. \( l_0 \) is the base layer and the rest are enhancement layers. \( l_n \) is the top layer which original data carries, sent by the source node.
• **Nodes** \{N_0, N_1, N_2... N_n\}
  This definition stands for the node joining to the multicast tree. \(N_0\) is the source node. \(n\) is the total number of nodes joining to the multicast tree.

• **Number of layer** \{L_0, L_1, L_2... L_n\}
  Number of layer is the layers which the source node maintains or a certain node requests. \(L_0\) is the maximum number of layers which the data carries sent by the source node. For example, if the source node carries layered coded data with 5 layers, \(L_0 = 5\), and \(N_0\) has layer \(l_0\) through \(l_5\).

• **Number of child nodes** \{C_0, C_1, C_2... C_n\}
  This definition stand for the number of child nodes, which receiving the stream from a certain node. Each node sets the maximum number of child nodes to support, according to its own network bandwidth represented as \(c_{\text{max}}\). In addition, the source node should set a minimum number of child nodes for every node joining to multicast tree, which will be represented as \(c_{\text{min}}\). \(c_{\text{min}}\) should be larger than one to construct a tree and node should set \(c_{\text{max}}\) equal to or greater than \(c_{\text{min}}\).

• **Depth** \{D_0, D_1, D_2... D_n\}
  Depth shows the position of the nodes in the multicast tree. The source node \(N_0\) is the top node in the multicast tree, therefore \(D_0 = 0\). If \(N_j\) has two ancestor nodes between the path to the source node, \(D_j = 2\).

C. **Tree structure of LOLCAST**

Characteristic of layered coding used in multiple layer approach makes a restriction in the method for delivering the data. The point is that there are dependencies between each layer. First enhancement layer \(l_1\) requires the base layer \(l_0\) for decoding. Second enhancement layer \(l_2\) needs both primal enhancement layer \(l_1\) and the base layer \(l_0\) for decoding, and so on. This means each layer could not be sent apart to decode the data.

![Fig. 4. Multicast tree in LOLCAST](image)

Therefore, LOLCAST uses the number of layers which each node requests as the metric to construct the multicast tree. The targeted group size of LOLCAST is at most the order of several hundreds.

There are three types of nodes in LOLCAST: the source node, relay nodes and leaf nodes. In LOLCAST, the source node maintains the entire multicast tree structure and serves the original data transmitted over the multicast tree. Relay and leaf nodes request the source node for the proper parent node to join, and receive the data they request. Leaf nodes only receives the data and does not have any child nodes.

An example of a multicast tree constructed by LOLCAST is illustrated in Fig.4. The minimum number of child nodes for this tree is 2 (\(c_{\text{min}} = 2\)) and the maximum number of layers is 10 (\(L_0 = 10\)). For simplicity, \(c_{\text{max}}\) for every node is set to 2. For example, node \(N_4\) is receiving data consisting of six layers \((l_0...l_5)\) from its parent node \(N_1\). In this case, node \(N_4\) could serve any child node requesting no more than six layers, which is node \(N_7\) and \(N_8\) in Fig.4. Node \(N_6\) and \(N_8\) are leaf nodes requesting one or two layers. Leaf node \(N_6\) and \(N_8\) can be expected as a node with very small resource environment, such as wireless devices.

D. **Recovery method from node failure**

Overlay Multicast relies on an unstable infrastructure, compared to IP Multicast. For this reason, researchers have large attention in the method for handling unstableness of end node. Especially, video streaming requires fast recovery to avoid loss of information, which is our target. This section introduces functions to handle this issue. First is multi-path layer distribution method and second is congestion avoidance method.

E. **Multi-path layer distribution**

This section describes multi-path layer distribution method for fast recovery in case of a node failure. There are several methods for fast recovery from multicast tree partition. Host-Cast [5] uses redundant path from the source node in control topology to shorten the time after detecting the node departure and to restart sending data. However this approach requires time to converge the multicast tree before recovery. LOLCAST uses data topology to construct a more redundant data delivery path compared to HostCast. By directly sending redundant data from multiple parent nodes, recovery time shorteneds compared with other methods.

Fig.5 illustrates how node \(N_3\) recovers when its parent node \(N_1\) fails. In the normal state, path \(N_0-N_1-N_3\) is used to deliver data from source node \(N_0\) to node \(N_3\). At the same time node \(N_3\) is redundantly receiving the base layer from node \(N_0\) and \(N_2\). In addition, all nodes has the option to request multiple
layers for redundant data, alternative for using base layer. When node \( N_1 \) fails, path \( N_0 - N_1 - N_3 \) becomes unavailable. As soon as node \( N_3 \) detects its parent node \( N_1 \) has failed, node \( N_3 \) switch to the redundant data receiving from \( N_0 \) or \( N_2 \) to reduce the loss of information. While receiving the redundant data from \( N_0 \) or \( N_2 \), \( N_3 \) recovers into multicast tree by finding a new parent node \( N_0 \) and starts receiving data with the requesting quality.

**F. Congestion avoidance**

![Fig. 6. Congestion control](image)

Fig. 6 illustrates an example of this function when congestion occurs in the path between \( N_1 \) and \( N_4 \). In this multicast tree, each node is requesting some layers in a range (ex. \( L_1 = 8 \) to \( 10 \)). In the normal state, each node receives the data with highest requested quality. In this case \( N_4 \) receives 8 layers from \( N_1 \). When \( N_4 \) detects that the path between \( N_1 \) and \( N_4 \) has congestion, \( N_4 \) drops the receiving layers one by one from the top to avoid the congestion. For detecting the congestion between nodes, existing methods can be used [11].

Not only for avoiding congestion, by setting a range for requesting data, it could handle various requests for quality flexibly. By narrowing the requesting quality range, it can guarantee the quality, but the possibility of switching nodes in the multicast tree increases. By widening the requesting quality range, the chances of switching in the multicast tree decreases offering a more stable service, but the receiving quality may change often.

**IV. DESIGN OF LOLCAST**

This section illustrates the protocol design of LOLCAST in detail. Messaging of Join Procedure and Leave Procedure are explained.

**A. Join Procedure**

![Fig. 7. Join Procedure 1](image)

Fig. 7.8.9 illustrates a case when \( N_4 \) joins to the multicast tree. A solid line stands for a data path, and a dotted line stands for a message path. This tree has two parameters, \( L_0 = 10 \) and \( c_{\text{min}} = 2 \). For simplicity, \( c_{\text{max}} \) for every node is set to 2. Join Procedure is done by messaging between nodes.

For the first step, we assume that new node \( N_4 \) can acquire the address of the source node \( N_0 \) and the max number of layers \( L_0 \) which \( N_0 \) can offer. \( N_4 \) sends Rendezvous Request to \( N_0 \) including its own node information. Source node generates a unique node identifier and sends back to \( N_4 \) using Rendezvous Accept message illustrated in Fig.7. Next, \( N_4 \) sends Parent List Request to \( N_0 \) including the requesting number of layers (\( L_4 = 5 \)).

Parent list is a list of nodes which are capable to be parent nodes, generated by the source node. The source node generates parent node list from several parameters using the tree structure, which is sorted by depth. There are mainly two conditions for a node to satisfy to be added to the parent list.

(a) Node has an open slot for children \( (c_{\text{min}} \leq C_i < c_{\text{max}}) \).
(b) Node has enough layers to satisfy the request of the new node \( (L_i \geq L_{\text{req}}) \).
(c) Node is not in the leave node list (to which nodes trying to leave from the tree are added). \( N_0 \) sends back the generated parent list including \( N_2 \) and \( N_3 \) to \( N_4 \) by Parent List Data message.

![Fig. 8. Join Procedure 2](image)

Fig. 8. Join Procedure 2

\( N_4 \) node picks \( N_2 \) from parent list and sends Join Request including node information of \( N_4 \) as illustrated in Fig.8. \( N_2 \) rechecks the parameters if it can sustain the request, and becomes the parent node for \( N_4 \). \( N_2 \) adds the node identifier, address and other information of \( N_4 \) as its child node. Next, \( N_2 \) sends Join Accept message to \( N_4 \). \( N_4 \) adds node information of \( N_2 \) to its data structure as the parent node. Finally \( N_4 \) sends Data Start Request to \( N_2 \), and data transmission starts.

![Fig. 9. Join Procedure 3](image)

Fig. 9. Join Procedure 3

After \( N_4 \) joined to the multicast tree, \( N_4 \) sends Notify Join Complete message to \( N_0 \) for updating the tree structure which it maintains, as illustrated in Fig.9.
B. Leave Procedure

Fig. 10 and 11 illustrate a case when \( N_4 \) leaves from the multicast tree. Parameters are same as in Join Procedure. For simplicity, \( c_{max} \) for every node is set to two. In Leave Procedure, it is desired for the leaving node not to make an effect to the data streaming of other nodes.

First, leaving node \( N_4 \) sends Notify Leave Progress message to the source node \( N_0 \) as illustrated in Fig. 10. This message notifies the source node that its state must be locked, so that the node will not be included in the parent list, for example. \( N_0 \) adds \( N_4 \) to the leave node list to lock the node from modification. Next, \( N_4 \) sends Leave Request to the child node \( N_5 \) to notify that \( N_4 \) is leaving.

When \( N_5 \) receives Leave Request, it runs Join Procedure to find another parent node. \( N_5 \) rejoins to the new parent \( N_1 \), and starts receiving the data from it. In this period, \( N_5 \) is receiving two data streams redundantly from \( N_4 \) and \( N_1 \). After switching the data stream from \( N_4 \) to \( N_1 \), \( N_5 \) sends Data Stop Request message to \( N_4 \), and \( N_4 \) stops sending data to \( N_5 \). Next, \( N_5 \) sends Leave Complete message to \( N_4 \) to notify that \( N_5 \) has found a new parent node and receiving data properly. Finally \( N_5 \) sends Notify Parent Changed message including the node information of both \( N_5 \) and \( N_4 \) to request the source node \( N_0 \) to update the multicast tree. \( N_0 \) updates the multicast tree to the current state.

V. CONCLUSION

In this paper, we have introduced a new Overlay Multicast protocol LOLCAST for sending layered data structure. Multiple version and multiple layer approaches are described, and their advantages and drawbacks are discussed. LOLCAST uses layered data structure for adapting to the heterogeneity of end nodes. Multi-layer distribution and congestion avoidance for node failures have been described. LOLCAST has solved the major issues in Overlay Multicast research: heterogeneity and instability of end nodes.

As future work, we are scheduling to evaluate the effectiveness and scalability of our protocol by comparing with other protocols using recent studies for evaluation on Overlay Multicast. In addition, we are scheduling to implement a sample application for real-time video streaming.

REFERENCES


