ON DESIGING END-USER MULTICAST FOR MULTIPLE VIDEO SOURCES

Yoshitaka Nakamura *Hirozumi Yamaguchi* *Akihito Hiromori*
Keiichi Yasumoto[†] *Teruo Higashino* *Kenichi Taniguchi*

Graduate School of Info. Sci. and Tech.
Osaka University
Toyonaka, Osaka 560-8531 Japan

[†]Graduate School of Information Science
Nara Institute of Science and Technology
Ikoma, Nara 630-0192, Japan

ABSTRACT

In this paper, we present a new application level multicast protocol called Emma (End-user Multicast for Multi-party Applications) suitable for communication systems where multiple video sources are exchanged in real-time among end-hosts, such as video-conferencing. The primal feature of Emma is that video sources with the higher priority given by users are prioritized among others for the provision of quality of service at the user level and that all the operations in Emma are done in a distributed manner. Our experimental results have shown that Emma can achieve reasonable performance on overlay networks with high user satisfaction.

1. INTRODUCTION

Group communication is in most cases regarded as the delivery of the same data to all/some members within a group. IP multicast is considered as an attractive solution for such a group communication, however, there is a known problem of the deployment of multicast infrastructure. On the other hand, multiple unicast connections lacks scalability as the group size growths.

As a realistic solution for the above problem, a new kind of communication paradigm which realizes multicast communication in the application layer (called ALM: application-level multicast) has had much attention. Recently, a lot of researches for ALM have been proposed [2, 3, 4, 5, 6, 7]. In particular, end system multicast (Narada) [2], ALMI [3] and Yoid [7] consider multi-party applications as one of their target systems. However, no method considers to handle multiple video sources efficiently.

In this paper, we propose a new application level multicast protocol called Emma (End-user Multicast for Multi-party Applications) suitable for communication systems where multiple video sources are exchanged in real-time among end-hosts. A video-conferencing system is a good example of Emma (see Fig. 1). A member in video-conference may want to show on his/her desktop some video windows simultaneously, for example, the conference site view, a chair's view and some key persons views. He/she may also have a priority requirement like "I strongly require the key persons' views, prior to the site view and the chair's view". Emma controls those multiple streams based on the priority requirements given by users in a fully distributed manner, for the provision of quality of service at the user level.

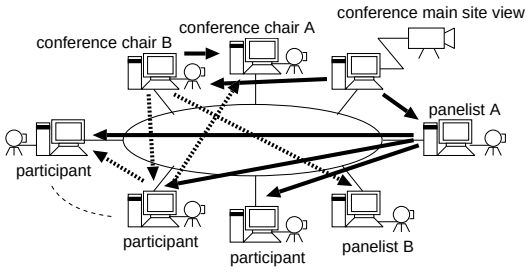


Fig. 1. Emma application example (video-conferencing). The site view is sent to panelist A and chair B's hosts and from them further sent to some other hosts.

2. EMMA PROTOCOL

Due to limitation of space, we overview the basic functionalities of Emma.

2.1. Join Management

We assume that there exists a server called a *lobby server* which keeps and manages the list of all the current nodes' profiles (IP addresses, port numbers, node IDs, node affiliations etc.) as well as the profiles of video sources sent by those nodes (resolutions, rates, codecs and content information etc.). A node who wants to join a session first asks the lobby server to obtain the list of member profiles. The member profiles are updated by periodical membership reports to the lobby server.

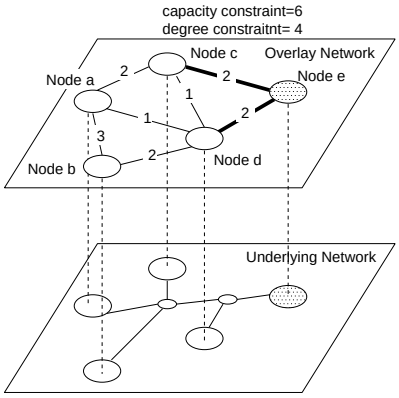


Fig. 2. Overlay network expansion by a new node. Values of overlay links represent their capacities.

After obtaining the member nodes' profiles, the new node measures delays or RTTs between those nodes, selects a few nodes with the smallest delays, and then tries to establish overlay links with them. Here, each node specifies the maximum number of overlay links as *degree constraint*, depending on its machine power to manage those links. As well as a degree constraint, each node can specify the upper limit of the total number of streams on its own overlay links of the node as *capacity constraint*, depending on LAN bandwidth and machine power. The new node negotiates with the candidate nodes and if the degree constraints and capacity constraints of both nodes are satisfied, overlay links are established between them.

Fig. 2 shows an example. We assume that the degree and capacity constraints of each node are 4 and 6, respectively. In the figure, on each overlay link its capacity (the number of streams that can be delivered on the link) is shown. Now suppose that a new node e tries to join the session and that nodes c and d are the candidate nodes. Here, node e negotiates the overlay link capacity with c and they agree that a link with capacity 2 does not violate the constraints of both nodes (since node c's total capacity becomes 5, which is still less than 6). Therefore, the link is established. Similarly, node e establishes an overlay link with capacity 2 with node d. The overlay network after this join is shown in Fig. 2.

2.2. Routing Tree Construction

In Emma, on an overlay network, each node has its own spanning tree as the routing tree, and it is expanded whenever a new node joins the overlay network. Trees are constructed under the restriction of maximum delays (or hop counts) defined in the session in advance, considering overlay link capacity. Periodical message flooding is used for re-construction of the trees when the trees are broken by an existing node's leave, and is also used for constructing the joined new node's routing tree.

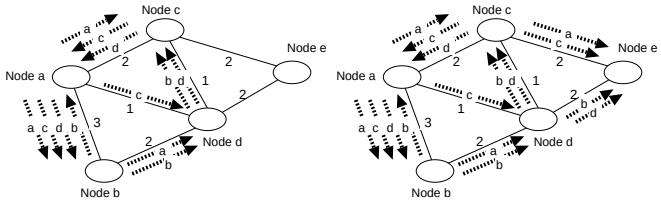


Fig. 3. Routing tree expansion by a new node on overlay network.

In Fig. 3, we show a snapshot of the tree construction when a new node joins a session. In Fig. 3(a), each of nodes a, b, c and d has its own routing tree which includes all the other nodes. Now, let us suppose that node e joins the overlay network by establishing overlay links to nodes c and d, and that the maximum hop counts specified in the session are two. Under this hop count constraint, node e can select only link c-e to be merged with the nodes a and c's routing trees by considering the hop count constraint. Similarly, node e can select only link d-e to be merged with the node b's routing tree. Here, node e can select either link c-e or d-e to be merged with node d's routing tree. However, considering the link capacity, d-e is the better choice since capacity competition does not occur on d-e. Fig. 3(b) shows the result of the above expansion. The routing tree of node e is constructed by flooding and it is not shown in the figure.

2.3. Video Content Delivery on Overlay Network

Each node is assumed to have his/her requirements to the other nodes' video sources. More concretely, each node specifies a value to indicate how much he/she is interested in receiving each video source. We call the value as a *preference value*. The preference values are used to select streams to be delivered on an overlay link when multiple streams compete for link capacity.

When a node u sends a request for node s's video source, the request message is forwarded upwards along the s's routing tree, to the intermediate node s' where the s's video source is already delivered. Node s' sends the reply message to node u along the reverse path. In each intermediate node on the reverse path, the node s's video source should be forwarded. Note that simultaneous requests by several nodes for the same video source are merged during their forwarding.

Here, if the capacity is not left on one or several overlay links on the path, Emma calculates the total sum of preference values lost by stopping the delivery of some existing streams to keep one capacity on those links. When the total sum of the preference values given in the new requests is

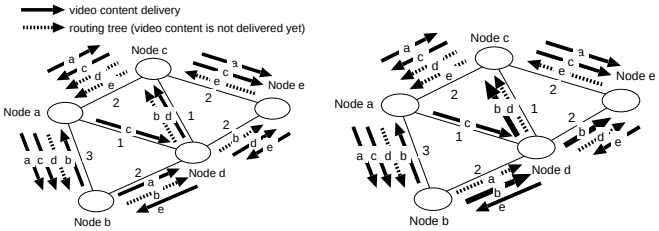


Fig. 4. Video delivery control on overlay network based on preference values.

greater than the total loss of preference values, the requests are accepted and the delivery of the corresponding streams is stopped. The above function is one of the most important characteristics of Emma, which greatly differs from existing application level multicast methods.

In Fig. 4, we show an example of this video distribution control. Fig. 4(a) shows all the routing trees (dotted lines) and delivery trees on which the video sources are actually delivered (normal lines). Obviously, the delivery trees are the subtrees of the routing trees. Now let us assume a scenario that nodes c, d and e request to receive node b's video source almost at the same time. The request messages are sent from nodes c and e to node d. Two messages are aggregated at node d and sent to node b. However, there is no capacity on links c-d, d-e and b-d. In order to keep one capacity on those links for node b's video, stopping a's video forwarding to node d at node b and stopping d's video forwarding to nodes c and e at node d is a possible solution. If the sum of the preference values given by nodes c, d and e to the b's video source (the total gain of preference values) is greater than the sum of that given by node d to the a's video source and those given by nodes c and e to the d's video source (the total loss of preference values) and if the loss is the smallest among all the other possible solutions, these requests are accepted and the video distribution is changed as shown in Fig. 4(b).

2.4. Leave and Failure Management

In Emma, when a node leaves an overlay network due to some reasons such as failure, the overlay network can be repaired automatically so that existing video streams continue to be delivered. Let us assume that a node u leaves an overlay network. Let v denote each neighbor node which has detected the node u's leaving. For each video source s delivered via node u, node v decides an alternative node which can forward s's video source and then connects himself to the node. Accordingly, the descendant nodes of v can continue to receive the s's video source and the total sum of preference values are kept as large as possible. In order to switch the disconnected link to an alternative node quickly, each node collects and keeps the information about the nodes in each delivery tree.

3. PERFORMANCE EVALUATION

We have developed a script written in an object oriented scripting language Ruby, for the event-based simulation of Emma for evaluating its performance.

We have produced networks based on a hierarchical topology model called tiers[1] consisting of LAN, MAN and WAN where we assigned about 50% of nodes in each LAN to overlay nodes (users).

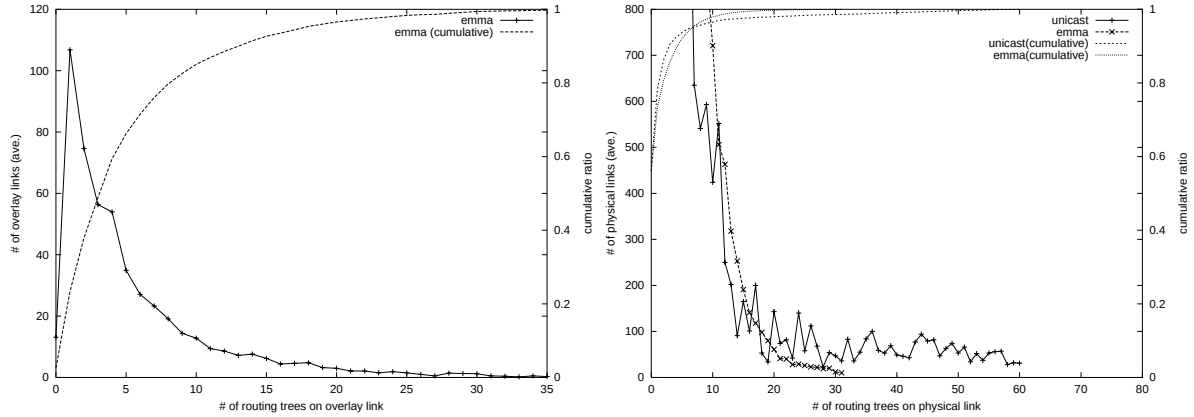
We used the following simulation scenario assuming a video conference. Users join the conference in random order, and each user requests three video sources. For each user $i(i = 1..N)$, $2N/i$ is assigned as its preference value like Zipf's law. By this, we represent bias in user preferences such that user preferences for video data of the chair person and the main conference room are always high. After each user has joined the conference session, it first requests to receive a video at random independently of its preferences. By this, we represent a situation that users change their preferences as the conference progresses (for example, the situation that the key person changes as the conference progresses).

3.1. Routing Tree Efficiency

We have carried out experiments on a network with 146 nodes (including about 66 overlay nodes on average) 10 times.

In Emma, each node is a potential sender of his/her video source. Thus, routing trees of those multiple video sources are constructed in a bandwidth-conscious way, in order to avoid overlap of routing trees on an overlay link. We have measured overlap levels of these routing trees (the number of routing trees on each overlay link). The result is shown as a distribution, in Fig. 5(a). In the figure, each curve depicts the number of links (Y-axis) at each number of routing trees (X-axis). We can see that about 80 % of overlay links have at most 10 routing trees, although 66 routing trees exist on overlay networks. We think that it is reasonable enough.

Next, in overlay networks, overlay links in a routing tree may include the same physical link. This means that the same data is transfered through a physical link more than once. Therefore, we have measured the number of duplications of a routing tree on a physical link. As comparison, we have measured the number of unicast routes from a node on a physical link. This number is bounded as the number of the other users. The result is shown as a distribution, in Fig. 5(b). We can clearly see the efficiency of Emma compared with unicast.



(a) Distribution of the overlay links against the number of routing trees on the overlay links.

(b) Distribution of the physical links against the number of duplications of a routing tree.

Fig. 5. Experimental results (146 nodes including 66 users).

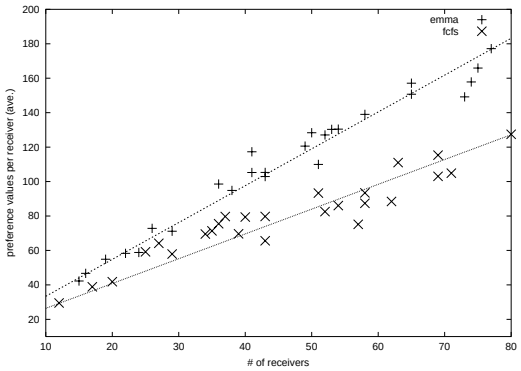


Fig. 6. Variation of the average preference value per user

3.2. Users' Satisfaction

With respect to users' satisfaction, we have compared Emma with a first come first serve (FCFS) method which accepts user requests in their submitted order as long as the overlay link capacity is left (the requests are rejected when the capacity is not left). We have measured the variation of the average preference value per user when changing the number of nodes. The result is shown in Fig. 6. Emma achieves 1.5 times as good value as FCFS since it provides a dynamic control mechanism based on preference values.

4. CONCLUSION

In this paper, we have proposed an application level multicast protocol called Emma, designed for multi-party video communication systems where each user continuously transmits realtime media while receiving some of the others with priority requirements. From the experimental results, we have confirmed that Emma could achieve the higher satisfaction of users (1.5 times as large as a normal method) keeping reasonable routing tree efficiency.

We have designed and implemented Java middleware based on Emma where host-based inter-stream QoS is supported.

5. REFERENCES

[1] K.L. Calvert, M.B. Doar and E.W. Zegura, "Modeling Internet Topology," *IEEE Communications Magazine*, pp. 160–163, 1997.

[2] Y.-H. Chu, S. G. Rao, S. Seshan and H. Zhang, "Enabling Conferencing Applications on the Internet using an Overlay Multicast Architecture," *Proc. of ACM SIGCOMM*, 2001.

[3] D. Pendarakis, S. Shi, D. Verma and M. Waldvogel, "ALMI: An Application Level Multicast Infrastructure," *Proc. of 3rd Usenix Symp. on Internet Technologies & Systems*, 2001.

[4] J. Jannotti, D.K. Gifford, K.L. Johnson, M.F. Kaashoek and J.W. O'Toole, "Overcast: Reliable Multicasting with an Overlay Network," *Proc. of the 4th Usenix Symp. on Operating Systems Design and Implementation (OSDI)*, 2000.

[5] S. Ratnasamy, M. Handley, R. Karp and S. Shenker, "Application-level Multicast using Content-Addressable Networks," *Proc. of 3rd Int. Workshop on Networked Group Communication*, 2001.

[6] Y. Chawathe, S. McCanne and E. A. Brewer, "RMX: Reliable Multicast for Heterogeneous Networks," *Proc. of IEEE INFOCOM 2000*, 2000.

[7] P. Francis, "Yoid: Extending the Internet Multicast Architecture," *Unrefereed Report*, 2002. `http://www.isi.edu/div7/yoid/`