On the Construction of Heterogeneous Multicast Distribution Trees Using Filtering in an Active Network

Héctor Akamine, Naoki Wakamiya, Masayuki Murata, and Hideo Miyahara

Department of Informatics and Mathematical Science
Graduate School of Engineering Science, Osaka University
1–3 Machikaneyama, Toyonaka, Osaka 560–8531, Japan
Tel: +81–6–6850–6588 Fax: +81–6–6850–6589
akamine@ics.es.osaka-u.ac.jp

Abstract. We propose a heuristic algorithm for the construction of a heterogeneous multicast distribution tree used for video transmission that satisfies different QoS requests. Our approach assumes an active network where active nodes can filter the video stream. By appropriately choosing the active nodes at which filtering is performed, the proposed algorithm constructs the multicast tree where, for example, the total required bandwidth is minimized. We assume existing multicast routing algorithms to form multicast groups, and the resulting distribution tree is a hierarchical arrangement of those groups.
We evaluate the performance of our algorithm and compare it against two other approaches: simulcast and layered encoded transmission. Results show that we can get advantages when network nodes participate in the construction of the heterogenous multicast distribution tree, such as the possibility to set up a larger number of simultaneous multicast sessions.

Keywords: heterogeneous multicast, active networking, video filtering.

1 Introduction

The problem of multicast distribution of video gets even more complex when considering a heterogeneous environment, where different clients joining the same multicast session have different quality requests due to limitations in the network bandwidth or in the end hosts' processing capabilities. We must deal with this heterogeneity processing the original video stream in a way that several different quality streams with different rate can be provided, and using a distribution method to give each client the stream with the quality closest to its requirement.

In simulcast, the server produces a different video stream for each requested quality. This is the easiest way of providing the multiple QoS requirements simultaneously, but leads to the waste of system resources. Our research group has proposed the use of flow aggregation [1], where clients with similar QoS requests are aggregated to minimize the number of streams produced at the sender. However this still remains as a simulcast transmission, and the server and the network would be overloaded when the number of clients is considerably large.

Layered encoded video has also been considered to address the problem of heterogeneity. In layered video, a video stream is decomposed into a base layer stream and some enhancement layers. The base layer is enough for decoding the video sequence but in the lowest quality, and the reception of additional layers is necessary to decode higher

quality video. Each client chooses the appropriate set of layers to achieve the preferred video quality. McCanne et al. proposed [2] a framework for the transmission of layered signals using IP multicast. The advantage of this approach is that it can be used in the current network infrastructure. Problems with layered video include the difficulty in generating the video layers, the required bandwidth overhead, and the granularity. Since only a few layers can be produced, we get only a few different qualities. We also considered the use of flow aggregation with layered coded video [3] and showed that we can accomplish further effective video multicast. However, this approach still retains the limitations of layered encoding, although we introduced a way of constructing video streams of 12 layers.

Against the heterogeneity, the research community has gotten interested in the possibility of introducing limited programmability inside the network, allowing the network nodes to decide if it is necessary to modify the contents and the routing of data packets after performing some computation. These processing-enabled nodes are called active nodes [4, 5]. Expectations for active networking are wide-ranging, starting from the possibility of accelerating the rate of introduction of new protocols to the introduction of novel services at the node level. An example is the introduction of new approaches for multicast [6-8] that exploit the advantages of intermediate node processing.

Heterogeneous multicasting of video is a natural candidate for enjoying the benefits of active networking. The server becomes free from providing video streams of different qualities if the quality regulation can be performed inside the network. At video filtering nodes, new video streams of lower qualities can be derived from the received ones at the expense of processing capability. The methods for reducing the quality of a video signal depend on the encoding format used.

Research into filtering by Yeadon et al. [9] and Pasquale et al. [10] predates active networking research, but propose a filtering propagation mechanism to vary the location where filtering occurs according to the requirements of downstream clients. AMnet [8] proposes a model and an implementation for providing heterogeneous multicast services using active networking. According to this approach, a hierarchy of multicast groups is formed, in which some active nodes that act as receivers in a multicast group become roots in other multicast groups, but it is not explained how the multicast groups are conformed and how the root senders of each multicast group are elected.

In this work we are facing one of the aspects related with heterogenous multicasting using active node filtering: how to decide in which active nodes filtering must be performed to achieve an efficient multicast distribution tree. We propose an algorithm that forms a hierarchy of multicast groups, where the top level group root is the video server. The members of this group are the clients which requested the highest quality video, and one or some active nodes which filter the video stream, producing one with lower quality. These active nodes become roots of other multicast groups to satisfy the requirements of other clients. Analogously, these new multicast groups can have one or some active nodes as members that become roots of even lower level groups.

We assume that filtering functions are provided within active nodes. How the filtering mechanisms are implemented is out of the scope of this work.

This paper is organized as follows: Section 2 explains our algorithm in detail; Section 3 evaluates its performance, comparing it with other approaches for distributing video; Section 4 concludes our work.

2 Construction of the Multicast Distribution Tree

In this section we detail our approach for the construction of the multicast distribution tree. We are assuming the following:

1. The server collects all of the client's requests, and it builds an appropriate multicast distribution tree previous to the start of the video transmission. We assume the server can get all the information it requires, such as the network topology and information about active nodes. The drawback of this centralized approach is lack of scalability: the proposed algorithm suffers with big topologies or large number of clients. We can alleviate this issue to some extent using clustering to group requests from closely located clients with similar quality requests.
2. For simplicity, we assume one QoS dimension, and therefore each client request can be expressed by a numerical value, that denotes the requested quality.
3. We can re-filter an already filtered video sequence in order to obtain another one with lower quality. We are not taking into consideration the effect that delays due to filtering can cause to the perceived quality.
4. We do not replace existing multicast routing algorithms, indeed we assume they are provided by the network layer. Our goal is the formation of the multicast groups, i.e., election of the roots and members of each one of them.
   DVMRP and MOSPF [11] are widely used multicast routing algorithms in dense mode subnetworks that build spanning trees rooted at the source using shortest path techniques. Therefore, we assume that the paths the multicast routing algorithm uses are the same as the unicast paths from the source to each one of the destinations, as created by Dijkstra's algorithm.

2.1 Sketch of the Active Application to Handle Filtering

Due to the centralized approach at the server, the complexity of the required application can be reduced. We would require the following steps:

1. Service Announcement. The server uses a well-known multicast address to inform the possible clients about the session contents. The protocol used to send these messages can be similar to the SAP protocol used in the MBONE [12].
2. Subscription. Each of the clients that wants to participate in the session sends a request containing the desired quality.
3. Calculation of the distribution tree. When this step is completed, we have the multicast groups and the nodes that are going to perform filtering defined.
4. Sending the information to the active nodes designated to perform filtering. The server transmits the following information to the nodes:
   - Multicast address as receiver: the active node is going to get the data to be filtered through this address
   - Multicast address as sender: the active node will send the data using this address
   - Filtering code and parameters: the kind of filtering to apply to the stream

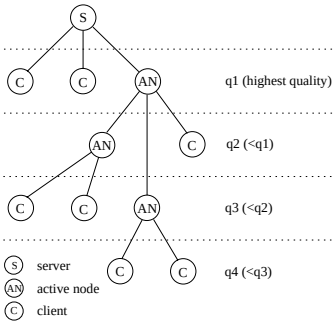

Fig. 1. Example of a logical topology

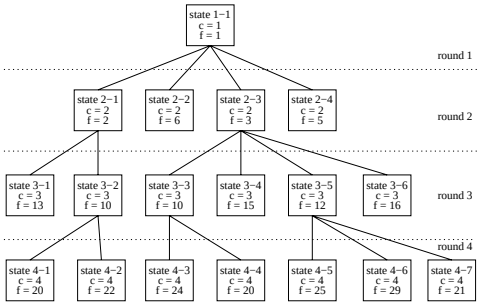

Fig. 2. Example of a state tree

5. Sending the corresponding multicast group address to each client, in order to have them subscribe.
6. Subscription of each client to the corresponding multicast group.
7. Data transmission. The server transmits the required streams to the multicast groups in which it is the root. Active nodes inside the network designated as filters redistribute the video stream as required.

2.2 Distribution Tree Construction Algorithm

Our algorithm forms a distribution tree in a request by request basis, taking the requests in descending order of quality. In the case that there are many requests with the same quality, we first take the ones from the clients closer to the sender. We try to use the sender to stream to the clients that require the highest quality, and choose the nodes located in the best place to perform filtering. The designated active nodes become the root node of a new multicast group of a filtered video stream of lower quality. The filtered stream is then sent to clients that demanded lower quality streams. We form a hierarchy of multicast groups as is proposed in AMnet [8], as shown in Fig. 1.

Each step in the construction of the tree defines a state. The state is defined by a variable c that stands for the number of the clients that have been already considered, and the characteristics of the distribution tree needed to serve those clients, that is, which nodes have been used to filter and produce, if any, the stream with the requested quality. Fig. 2 depicts a sample state tree. Each state is denoted as $c - i$, where c stands for the number of clients, $i \leq N_c$ is the state index, and N_c is the number of states in that round. At the first round, there is only one state 1-1, where only one client with the highest demand is satisfied by being provided the video stream at the required quality directly from the server. From a state in round c, it is possible to derive several states for round $c + 1$, depending on how the stream that the new client demands has been generated.

When deriving states from a round in the state tree, we define a set of "candidate senders" to provide the requested stream to the client newly considered in the next round. Either the original server of the video sequence or any of the active nodes in the network can be the candidate sender. For a given flow request and candidate sender, one of the following situations is possible:

1. The candidate sender is already requested to relay a stream with the desired quality by a previously processed client. In this case the client subscribes to the multicast group the stream belongs to.
2. The candidate sender is already requested to relay a stream with a quality higher than the one requested. In this case, this stream must be filtered at this candidate sender. Then, a new multicast group is created with the candidate sender as the root, and the requesting client becomes a member of this multicast group.
3. The candidate sender is not relaying a flow. In this case, the candidate sender must first subscribe to a multicast group, filter the stream that receives as a member of this group, and become the root of a new multicast group. The requesting client subscribes to this new group to get the stream.

The election of the filtering nodes is based:

1. On the distance, i.e., number of hops, between the client and the candidate node. The first candidate to choose is the closest one to the client that already belongs to the distribution tree, i.e., that relays or filters a flow to satisfy requests of previous rounds. The next ones are chosen close to this one.
2. On a function f that considers other factors such as total bandwidth used, link utilization, and/or the use of node resources. This function can be thought as a measure of how good is the complete distribution tree being formed. A lower value of f means a better distribution tree.

For simplicity, we assume only one variable that comprises the node resources, and that a filtering operation reduces the value of this variable by a predefined amount. If one active node has already exhausted its resources, filtering cannot be performed, and it is not considered as a candidate sender.

As described above, our algorithm belongs to the category of exhaustive search algorithms. It means that the number of possible states in each round directly affects the efficiency of our algorithm. In the worst case, the number of candidate senders is equal to the number of active nodes in the network, say A, plus the original server. In such a case, the number of states N_c in round c becomes $(A+1)^{c-1}$. Since this is computationally expensive if the number of requests or active nodes in the network is not small, two parameters were defined to restrict the number of states N_c to analyze:

- We limit the number of candidate senders to expand in each round to a fraction b of the total candidate senders.
- We restrict the number of new states generated in a round to a maximum of m.

In each round, we select up to a maximum of m states to expand, the states chosen are the ones with the lowest values of f. Each state is expanded with $b \times (A+1)$ new states, in which each new state implies a different candidate sender elected to satisfy the request of the next client. The election of these new states is done by the distance in number of hops criterion explained above. We continue expanding the state tree until all the clients' requests are satisfied. Then, the state with the lowest f is chosen.

2.3 Example

Figs. 3 and 4 show an example network topology with 10 nodes. Active nodes are marked with squares and non-active ones with circles. Client requests are indicated with unfilled circles with a number that represents the requested quality. The server is

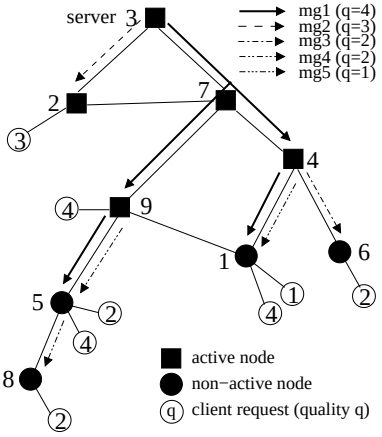

Fig. 3. Multicast groups for our algorithm

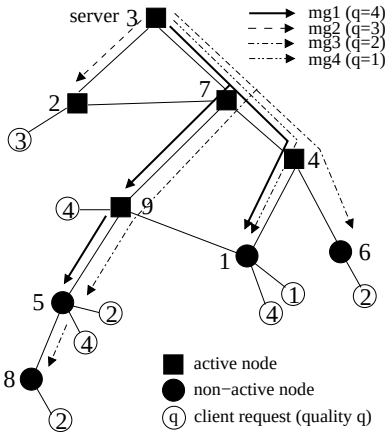

Fig. 4. Multicast groups for simulcast

Table 1. Required bandwidth for streaming video (Mb/s)

quality (quantizer scale)	single-layer video	layered video
4 (10)	14.4	22.65
3 (20)	8.8	13.64
2 (30)	6.6	8.75
1 (40)	5.4	5.19

attached to node 3. When the sender is attached to an active node, we must distinguish if the filtering is performed at the active node, or if the stream is provided by the sender.

The qualities are related with the bandwidth according to the data in Table 1, taken from a previous work from our research group [13] for the MPEG-2 video coding algorithm [14]. In layered video case, the layers must be piled up to achieve higher quality video. For example, the bandwidth required for a stream of quality 4 is given as 5.19 (layer 1) + 3.56 (layer 2) + 4.89 (layer 3) + 9.01 (layer 4) = 22.65 Mb/s. The different qualities are obtained varying the quantizer scale, and active nodes derive the video stream of lower quality by de-quantizing and re-quantizing the received stream.

Fig. 3 shows the multicast groups conformed by our algorithm. Arrows show the required streams, and arrow tips point to multicast group members. Two filtering processes are needed in node 4 and one in node 9. It must be noted that active node 4 becomes member of multicast group 1, just to provide filtered streams to clients in nodes 1 and 6.

As a comparison, Fig. 4 depicts the multicast groups needed in simulcast transmission. The links between nodes 3-7 and between nodes 7-4 have to carry many different quality streams, and for this reason they can become congested links. Using the values of Table 1, using simulcast we need to use 26.4 Mb/s in these links, compared with the 14.4 Mb/s needed by our algorithm.

2.4 Applicability to Multiple QoS Dimensions

For simplicity, we assumed above the existence of only QoS dimension. If we lift this restriction, e.g., if clients request a stream with a defined quantization scale and frame rate, the proposed algorithm is still applicable, but with the difference that the stream delivered to each multicast group is characterized with two parameters instead of one. That implies that an active node must be provided with a stream whose both quality parameters are greater than the quality parameters of the stream that it must produce. Once the QoS parameters are specified, we can deduct the required bandwidth using an approach proposed by previous work of our research group [15].

3 Evaluation

In this section, we show the effectiveness of our proposed algorithm through some numerical experiments. We generate random topologies using Waxman's algorithm [16], and choose the parameters appropriately to generate topologies with an average degree of 3.5, to try to imitate the characteristics of real networks [17]. We assumed the proportion of active nodes in the network to be 0.5. For simplicity, each filtering operation is assumed to use the same amount of resources. We also assumed that the number of filtering operations that each active node can do is a random value between 15 and 30. The location of active nodes is chosen at random. The location of the server, the clients and their corresponding requests' qualities are also generated randomly, and vary from one experiment to the other. Clients can request the video stream in one of four available video qualities, according to Table 1. We apply two other approaches for multicast tree construction to the same topologies for comparison purposes. Those are simulcast and distribution of layered coded video.

The definition of f, which is used to evaluate the effectiveness of the built tree in the algorithm can be modified according to which network parameters are most important in the construction of the distribution tree. We performed the evaluation using two simple definitions, those are for minimizing bandwidth and minimizing link utilization.

3.1 Minimizing Bandwidth

In this case the definition of f is the total used bandwidth for the video distribution tree. We call this definition for f as f_1:

$$f_1 = \sum_{i \in \mathcal{U}} B_i \tag{1}$$

where i denotes a link, $\mathcal{U}$ is the set of used links, and B_i denotes the bandwidth devoted to video distribution in link i.

To evaluate our algorithm with f_1, we generate ten 20-node and ten 50-node network topologies, varying the number of requests among 10, 20 and 50. The results are summarized in Figs. 5 and 6. In both figures, the first 10 experiments are for 10 requests, the next 10 for 20 requests and the last 10 for 50 requests.

In general, the proposed algorithm requires a lower total bandwidth than simulcast and layered video, at the cost of requiring processing at the filtering nodes. When the number of requests is small, the total bandwidth used by simulcast transmission is even smaller than the one required for layered transmission, because the overhead of the latter

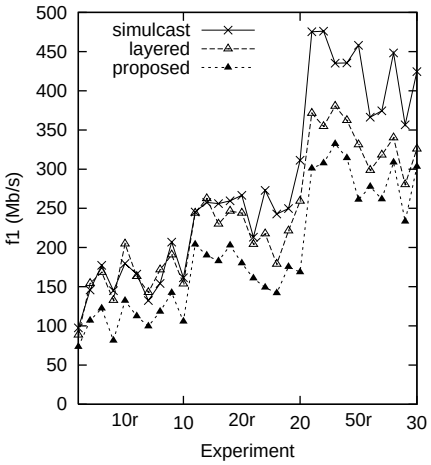

Fig. 5. Total bandwidth, 20-node network

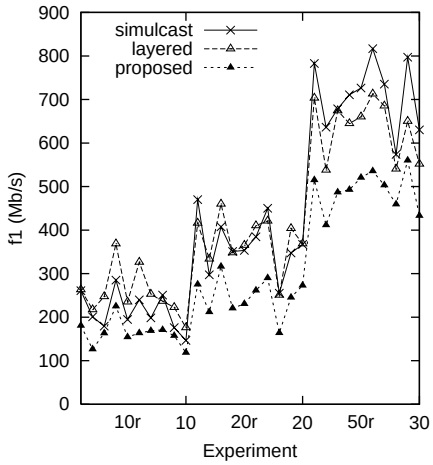

Fig. 6. Total bandwidth, 50-node network

is not justified owing to the dispersed clients. As the number of requests increases, the bandwidth required for layered encoding transmission becomes less than the required by simulcast, and becomes closer to the one required by our proposed algorithm. Since we fixed the proportion of active nodes to be 0.5 in the generated topologies, when we increase the number of requests, the number of non-active nodes that have clients attached requesting different quality streams is also increased. In those cases, several streams must be relayed toward the non-active nodes because filtering cannot be done locally, thus increasing the value of f_1.

3.2 Simultaneous Multicast Sessions

Minimization of the total bandwidth required for video multicasting is intended to avoid the extremely high load on the network and let other sessions set up their trees. In this subsection, we compare our algorithm increasing the number of sessions in the network to see how many sessions can be simultaneously set up and provided for users.

In the experiments, all the links are assumed to have a bandwidth of 100 Mb/s. We multiplex sessions, each of which is set up according to our algorithm, until the bandwidth of any link is exhausted. Here, we should note that the network we consider is best-effort and the constraint on the available link bandwidth is not taken into account in our algorithm stated in Section 2. Thus, the number we consider here is that of simultaneously acceptable sessions without causing a seriously overloaded link. The sessions are independent, and we do not use the information of the links used by the other sessions to build the current tree.

In addition to f_1, which is to minimize the total bandwidth, we introduce other function f_2, which is related to the required average bandwidth of the used links:

$$f_2 = \frac{\sum_{i \in \mathcal{U}} B_i}{|\mathcal{U}|} \tag{2}$$

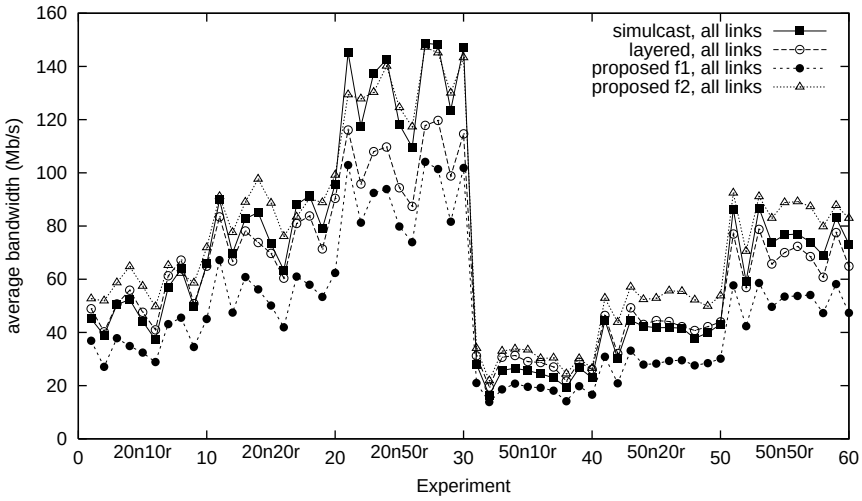

Fig. 7. Average bandwidth (Mb/s) for the first ten sessions

where i denotes a used link, $\mathcal{U}$ is the set of used links, and B_i denotes the used bandwidth in link i. We expected that with this definition, our algorithm could perform some sort of "load balancing," to avoid congesting a single link.

In Figs. 7 and 8, experiments 1-10, 11-20, 21-30, 31-40, 41-50, 51-60 refer to 20-nodes 10-requests, 20-nodes 20-requests, 20-nodes 50-requests, 50-nodes 10-requests, 50-nodes 20-requests, and 50-nodes 50-requests cases, respectively.

Fig. 7 shows the average bandwidth required to establish the first ten sessions at the same time. f_1 shows the lowest value for all the cases. Even though we chose f_2 to minimize the average bandwidth of used links in each session, when we sum all the sessions, f_2 results in the highest values. Between them lie the values for simulcast and layered video. When the number of requests is small (10 requests), the average bandwidth used by layered encoded distribution is greater, but for larger number of requests it is surpassed by the values of simulcast.

As a result of spreading the sessions in a large number of links, the variance of the bandwidth for f_2 is also reduced. The values for f_1 are even smaller due to the reason than the average bandwidth values are smaller compared with the other cases. We do not show the numerical values due to space limitations.

Fig. 8 shows the maximum number of simultaneous sessions that could be set up using each one of the three methods, i.e., the proposed algorithm, simulcast and layered distribution. The results show performance in the following order, from better to worse: the proposed algorithm using f_1, the proposed algorithm using f_2, layered transmission, and simulcast. There were some few cases in which our proposed algorithm was surpassed by the layered video approach. We expect this to occur when we have the same stream with different qualities over the same link, congesting it as it occurs in simulcast. This occurs, for example, when we have several clients connected to a non-active node that request different quality streams.

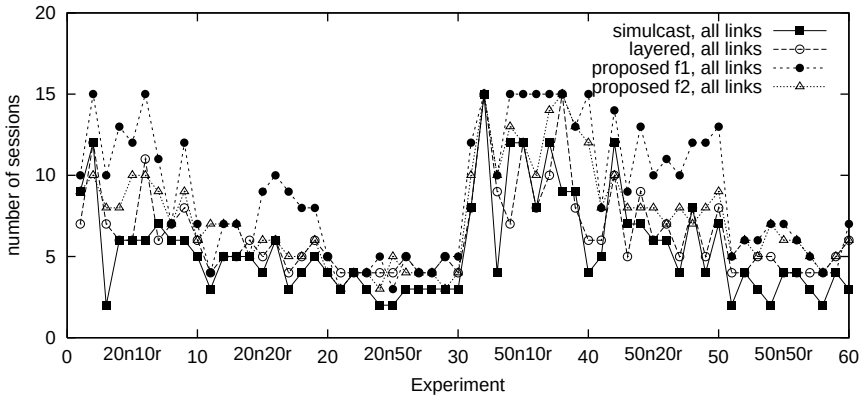

Fig. 8. Maximum number of simultaneous sessions (out of 15)

Even when the location of senders are concentrated in a region of the network, the advantage of f_2 is relatively small although results are not shown in this paper due to space limitation.

3.3 Required Computation Time

In this paper, we have not analyzed the effect of varying the values of b and m. Their election involves a trade-off between required processing time and optimality of the obtained solution. Just to have an idea, we averaged the time required by our algorithm to generate some multicast trees. For networks with 20 nodes, we required an average of 6, 14 and 56 seconds for trees with 10, 20 and 50 requests, respectively, using $m = 20$ and $b = 1$. For networks with 50 nodes, we required 101, 238, and 451 seconds for 10, 20 and 50 requests, respectively, using $m = 20$ and $b = 0.3$ for the first two cases and $b = 0.2$ for the last case. We evaluated our algorithm written in Java on an 800 MHz Pentium III machine.

3.4 Other Definitions for f

We considered above two simple definitions for f: the first one, f_1, is simply the sum of the bandwidth used in each link of the distribution tree; the second one, f_2 is an expression for the average bandwidth of used links. With f_2 we expected to increase the number of possible simultaneous sessions, reducing the bandwidth used per link, at the expense of increasing the number of used links. The problem with f_2 is that it increases greedily the number of used links in the tree, sometimes misplacing the filtering location. As we mentioned before in this section, some links can get congested when they carry the same video with different qualities. We could augment the definition of f_1 with a function that tries to avoid the existence of links that carry different quality flows simultaneously, adding a penalty value if those links exist.

We can also consider to limit the number of filtering operations performed in the distribution tree or at a single node, if we need to ensure limited use of node processing

resources. Although we assume a best-effort network, if link capacity is constrained, it is also possible to modify f to consider this restriction.

4 Summary

We presented an algorithm for electing the filtering nodes in an active network to construct a heterogeneous multicast distribution tree, which aims to minimize a function f that can be set to consider some network parameters, to achieve efficient use of the network resources.

We evaluated our algorithm choosing two simple definitions for f: the total bandwidth used, i.e., the sum of the bandwidth used in each link, and the average bandwidth of used links. We compared our algorithm with other two methods of distributing video that not consider the use of active nodes: simulcast and layered encoded distribution, and found that using our algorithm we can set up a greater number of simultaneous sessions, meaning a more effective use of the available bandwidth of the network, but at the expense of requiring processing capability at the network nodes.

In our evaluation, we choose f to take very simple forms. We have not tested further, but we think that if we elect other more elaborated appropriate definitions for f, it's possible to achieve better distribution trees.

We also presented a simple outline of how an active application can make use of our approach. It can result in a simple implementation, since the tree is constructed in a centralized approach. However, due to this reason, we can't expect that the algorithm scales to large internetworks.

We did not consider changing the distribution tree dinamically due to changes in the network conditions, necessary in the case of best effort networks that is what we considered. It is left as a future research topic.

Acknowledgments

This work was partly supported by Research for the Future Program of Japan Society for the Promotion of Science under the Project "Integrated Network Architecture for Advanced Multimedia Application Systems," a Grant-in-Aid for Scientific Research (A) 11305030 and a Grant-in-Aid for Encouragement of Young Scientists 12750322 from The Ministry of Education, Science, Sports and Culture of Japan, Special Coordination Funds for Promoting Science and Technology of the Science and Technology Agency of the Japanese Government, and Telecommunication Advancement Organization of Japan under the Project "Global Experimental Networks for Information Society."

References

[1] Kentarou Fukuda, Naoki Wakamiya, Masayuki Murata, and Hideo Miyahara, "On Flow Aggregation for Multicast Video Transport," in *Proc. Sixth IFIP International Workshop on Quality of Service*, May 1998, pp. 13–22.

[2] Steven McCanne, Van Jacobson, and Martin Vetterli, "Receiver-driven Layered Multicast," in *Proc. ACM Sigcomm '96*, August 1996, pp. 117–130.

[3] Naoki Wakamiya, Kentarou Fukuda, Masayuki Murata, and Hideo Miyahara, "On Multicast Transfer of Hierarchically Coded Video with QoS Guarantees," in *Proc. Internet Workshop '99*, February 1999, pp. 174–180.

[4] Jonathan Smith, Kenneth Calvert, Sandra Murphy, Hilarie Orman, and Larry Peterson, "Activating Networks: A Progress Report," *IEEE Computer*, pp. 32–41, April 1999.

[5] David Tennenhouse, Jonathan Smith, David Sincoskie, David Wetherall, and Gary Minden, "A Survey of Active Network Research," *IEEE Communications Magazine*, pp. 80–86, January 1997.

[6] María Calderón, Marifeli Sedano, Arturo Azcorra, and Cristian Alonso, "Active Network Support for Multicast Applications," *IEEE Network*, pp. 46–52, May/June 1998.

[7] Li-Wei Lehman, Stephen Garland, and David Tennenhouse, "Active Reliable Multicast," in *Proc. IEEE Infocom '98*, 1998, pp. 581–589.

[8] Bernard Metzler, Till Harbaum, Ralph Wittmann, and Martina Zitterbart, "AMnet: Heterogeneous Multicast Services based on Active Networking ," in *Proc. IEEE OpenArch '99*, March 1999, pp. 98–105.

[9] Nicholas Yeadon, Francisco García, David Hutchinson, and Doug Shepherd, "Filters: QoS Support Mechanisms for Multipeer Communications," *IEEE Journal on Selected Areas in Communications*, vol. 14, no. 7, pp. 1245–1262, September 1996.

[10] Joseph Pasquale, George Polyzos, Eric Anderson, and Vachaspathi Kompella, "Filter Propagation in Dissemination Trees: Trading Off Bandwidth and Processing in Continuous Media Networks," in *Proc. NOSSDAV '93*, 1993.

[11] Stardust Technologies, "Introduction to IP Multicast Routing," available at `http://www.ipmulticast.com`.

[12] Mark Handley, "SAP: Session Announcement Protocol," Internet Draft, Work in Progress, November 1996.

[13] Kentarou Fukuda, Naoki Wakamiya, Masayuki Murata, and Hideo Miyahara, "Real-Time Video Distribution with Hybrid Hierarchical Video Coding in Heterogeneous Network and Client Environments," in *Proc. MMNS '98*, November 1998.

[14] ISO/IEC DIS 13818-2, "MPEG-2 Video," ISO Standard, 1994.

[15] Kentarou Fukuda, Naoki Wakamiya, Masayuki Murata, and Hideo Miyahara, "QoS Mapping Between User's Preference and Bandwidth Control for Video Transport," in *Proc. Fifth IFIP International Workshop on Quality of Service*, May 1997, pp. 291–302.

[16] Bernard Waxman, "Routing of Multipoint Connections," *IEEE Journal on Selected Areas in Communications*, vol. 6, no. 9, pp. 1617–1622, December 1988.

[17] Ellen Zegura, Kenneth Calvert, and Samrat Bhattacharjee, "How to Model an Internetwork," in *Proc. IEEE Infocom '96*, 1996.