NHAG: Network-aware Hierarchical Arrangement Graph for Application Layer Multicast in Heterogeneous Networks

Masahiro Kobayasi†, Hidehisa Nakayama†, Nirwan Ansari‡, and Nei Kato†
†Graduate School of Information Sciences, Tohoku University, Japan
‡Advanced Networking Laboratory, ECE Department, NJIT, USA
E-mail: kobayasi@it.ecei.tohoku.ac.jp

Abstract— **Application Layer Multicast (ALM) is highly expected to be the new technological choice contents delivery in lieu of IP multicast. Depending on each node's streaming application, ALM constructs multicast trees and delivers the stream through those trees. The problem of ALM is that when a node resides in tree leaves, the stream cannot be delivered to descendant nodes. To overcome this problem, Topology-aware Hierarchical Arrangement Graph (THAG) was proposed. By employing Multiple Description Coding (MDC), THAG first splits the stream into a number of sub-streams, and then uses Arrangement Graph (AG) to construct an independent tree for each sub-stream. However, using the same size of AG in THAG has a difficulty delivering a stream appropriately across a heterogeneous network. In this paper, we propose a method to change the size of AG dynamically in enhancing THAG performance well even in a heterogeneous network. Finally, we evaluate the proposed scheme by experiments in ns-2. By comparing with THAG, we show that our proposal scheme provides a better performance in throughput and Bandwidth Satisfaction Rate (BSR).**

I. Introduction

Recently, along with the rapid growth in network speed and bandwidth, there has been an increasing deployment of contents delivery by applying streaming technologies. Most of the streaming technologies in the current Internet are based on unicast communication. However, since streaming based on unicast communication increases the traffic load of the server and network, this research has recently been directed towards the multicast communication based streaming [1]. Most of multicast communication based on IP multicast incurs a great deal of cost. Therefore, as an alternative to IP multicast, Application Layer Multicast (ALM), in order to be tailored into the current Internet infrastructure [3] has drawn much attention.

In IP multicast, the duplication and relay of packets are done at the router. In contrast, in ALM, the multicast communication is realized by the duplication and relay of packets of the application at the end-host. If we apply ALM, we can virtually perform multicast communication in the application layer regardless whether we perform unicast communication in the IP layer or not. Besides, while IP multicast requires special devices, ALM does not. Generally, the duplication/relay of packets performed by the end-host is less reliable than the one performed by the special router.

In ALM, an overlay network is constructed at the application layer independent from the network layer by the tunneling of unicasts between end-hosts. A multicast tree is created by having the end-host which is responsible for duplication of the received media as the branch. The stream delivery by ALM is performed according to the multicast tree with the source node which owns the media as the root. So far, End System Multicast (ESM) [5], Application Layer Multicast Infrastructure (ALMI) [7], Overcast [8], and Scribe [9] have been proposed. These methods use only one multicast tree to deliver a stream. Therefore, if the stream is not delivered due to the leaving of nodes, the quality of the stream will degrade dramatically.

In order to cope with this problem, multiple-tree multicast has been proposed [4], [11], [12]. This method splits the stream into several sub-streams with Multiple Description Coding (MDC) [13] and delivers the sub-streams by using multicast trees in parallel. In MDC, we can playback the contents by receiving one of the sub-streams, and higher quality can be achieved by obtaining more sub-streams. In multiple-tree multicast, CoopNet [12], Splitstream [11], and Topology-aware Hierarchical Arrangement Graph (THAG) [4] have been proposed. Both CoopNet and SplitStream do not ensure the independence of multiple trees, implying that a node can be an interior node in several multicast trees and its leaving will prevent the descendent nodes from receiving streams. In the THAG scheme, independence of the multicast trees is ensured. This independency guarantees that the leaving of any node will affect the data delivery at most in one multicast tree.

THAG is difficult to deliver a stream which can meet the various bandwidth constraints in the heterogeneous network. In THAG, the independent trees have been constructed from Arrangement Graph (AG) [14] and the upload bandwidth needed for the minimum transmission is determined by the size of AG and the streaming rate. Therefore, if the actual upload bandwidth of a node which is trying to join the multicast is less than or equal to the necessary upload bandwidth, it will not be able to send all the streams. As a result, the quality of the received stream will degrade depending on the amount of the stream which cannot be delivered. This problem is attributed to the characteristic that THAG uses all the same size of

1930-529X/07/$25.00 © 2007 IEEE
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE GLOBECOM 2007 proceedings.

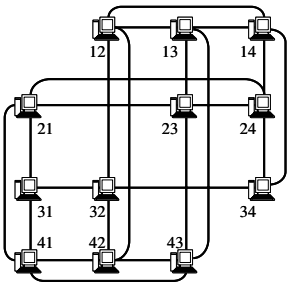

Fig. 1. AG with a size of 4

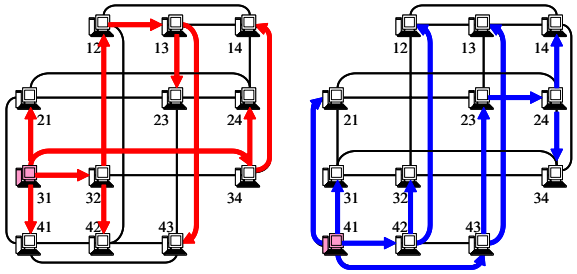

Fig. 2. Independent trees based on AG with size of 4

AG. Therefore, we propose a method to ensure the quality of the received stream by dynamically changing the size of AG according to the available bandwidth and by preventing nodes from being disabled to send the stream, when nodes join or leave the multicast. That is, our approach modifies and tailors THAG for heterogeneous networks. Our simulation results using network simulator ns-2 demonstrate that our approach provides throughput and Bandwidth Satisfaction Rate (BSR) better than the conventional THAG. The results indicate that our approach is more reliable in heterogeneous Networks.

The rest of the paper is organized as follows. Section II describes ALM which uses THAG. Our proposed ALM method is described in Section III. In Section IV, we present our simulation results and performance comparisons. Concluding remarks are given in Section V.

II. Topology-aware Hierarchical Arrangement Graph (THAG)

R. Tian *et al.* [4] proposed a method using THAG to construct ALM which uses MDC. The basic idea of this approach is to construct the independent trees and deliver the stream along the tree structures. The independent trees can be constructed by making a node which is a parent node in the specific tree to be the leaf node in all other trees. By so doing, even when a node cannot receive the sub-stream due to the leaving of a node in its upper position, the descendent node can still receive the stream from other trees.

A. Arrangement Graph (AG)

In THAG, $(S, 2)$-Arrangement Graph [14] is used to construct the independent trees. In this paper, we call S the size of AG. Generally, in AG with size S, $S(S-1)$ number of nodes can participate, and we can maintain $S-2$ number of independent trees. Fig. 1 shows an example when the size of

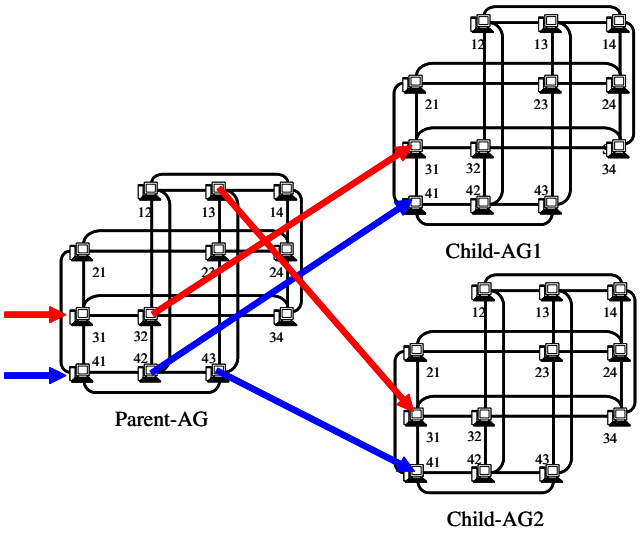

Fig. 3. Hierarchical AG

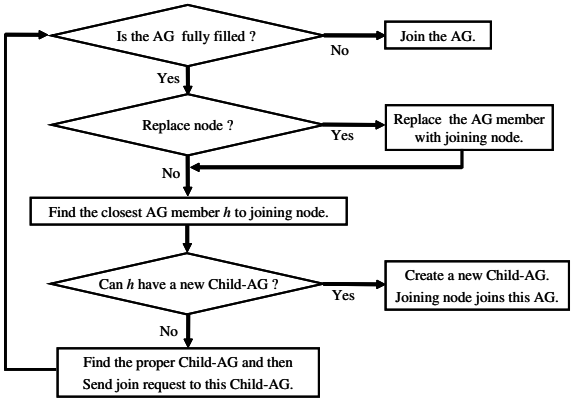

Fig. 4. Node joining procedure of THAG

AG is 4, while Fig. 2 is an example of the trees based on size 4 of AG. In the figure, the root of each tree is node $k1$ $(3 \leq k \leq S)$. As shown in Fig.2, two trees which have the root nodes 31 and 41 have been constructed. In these trees, we can see that the node which is the parent node in one tree is the leaf node in another tree. Therefore, these two trees are independent. node 21 is leaf node in all multicast trees. So, node 21 is selected as AG entrance and maintain the current states of all the AG members.

Furthermore, more nodes can join the AG in a hierarchical manner. When the number of nodes participated in AG reaches the limitation, such AG is made to be Parent-AG, which can build the new Child-AGs. Fig. 3 shows an example on how Child-AG1 and Child-AG2 are generated from nodes 32 and 42, and nodes 13 and 43, in Parent-AG, respectively. In general, nodes in each column deliver sub-streams to Child-AG. A column of nodes in the parent AG providing data to its Child-AG is referred to as the AG sources.

The stream which is delivered to Parent-AG is also delivered to Child-AG1 and Child-AG2 as well. In other words, since the delivery of sub-streams is performed based on the delivery

1930-529X/07/$25.00 © 2007 IEEE
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE GLOBECOM 2007 proceedings.

tree constructed from AG, we can easily achieve a large-scale delivery network in a hierarchical manner.

B. The Joining and Leaving of nodes

In THAG, Global Network Positioning (GNP) approach [15] is used to infer the distance between nodes in the network. In GNP, the inference of distance is based on RTT between nodes. Hereunder we describe the process of the joining and leaving of nodes in which the inferred distance is based on GNP.

At the beginning, the joining node will first send join message to the highest AG entrance. If the AG is not fully filled, the node joins that AG. Otherwise, if the AG is fully filled, for each AG member e which already joins the AG, compute the function $G(e)$ which is the ratio of the sum of distances between node e and the AG sources to the sum of distances between the joining node and AG sources as follows:

$$G(e) = \frac{\sum_i d_{\mathrm{GNP}}(e, s_i)}{\sum_i d_{\mathrm{GNP}}(n, s_i)} \quad (1)$$

Here, n denotes the joining node, s_i denotes the ith AG source of each sub-stream, and d is the distance function between nodes. Note that $G(e) > 1$ implies that the joining node is closer to the AG sources than node e. Therefore, node e with maximum $G(e)$ is replaced by the joining node n. The node e that is replaced will try to find a new Child-AG. On the other hand, for all member nodes e, no replacement is performed if $G(e) \leq 1$. In this case, n will try to find a new Child-AG. Next, the AG entrance that received the join message finds the closest AG member h to the joining node. The joining node contacts with h and retrieves the information of all its Child-AGs. If h has less child-AGs than it can serve, the joining node creates a new Child-AG and joins the AG entrance. Otherwise, the joining node contacts all the Child-AGs' entrances, and selects and joins the AG that has the smallest average distance between the joining node and the AG members. The above procedures are summarized in Fig.4. By repeating these procedures, the joining node eventually joins the closest Child-AG.

In case a node leaves the tree, if the leaving node has child-AGs, it can contact with the entrance of a Child-AG and promotes a non-root node in Child-AG. In the Child-AG, similar maintenance can be performed afterwards. Thus, the height of THAG can be reduced as much as possible.

III. Network-aware Hierarchical Arrangement Graph (NHAG)

In THAG, we can create several independent multicast trees from AG. The sub-streams are delivered by using these multicast trees in parallel. The minimum bandwidth needed for streaming delivery is determined based on the size of AG S and the streaming rate R, which is $2(S-2)R$. For example, the required bandwidth is $3.6Mbps$ for AG which has size 8 and the streaming rate of $300kbps$. For this reason, nodes which connected with the link having bandwidth less than this amount will not be able to send all sub-streams. In the conventional THAG, the required bandwidth has not been taken into account by assuming that all AG size is fixed and sub-streams consume the same bandwidth in all links. However, in a real network, the bandwidth of links connected with users can vary from place to place. In this paper, we consider the real network and propose a method that can be adaptive to the variation of link bandwidth by changing the size of AG dynamically. We call the method Network-aware Hierarchical Arrangement Graph (NHAG).

A. Joining and Leaving of Nodes

In the joining process, the joining node calculates the requested size S_R which is the maximum AG size required to deliver all sub-streams as follows.

$$S_R = \frac{BW}{2 \times R} + 2 \quad (2)$$

Here, BW is the actual upload bandwidth for the joining node, R is the streaming rate, and $S_R \geq 3$ is the constraint. Then, the joining process searches for the joinable AG based on this S_R. The AG which receives the join request determines whether or not to let the joining node join by comparing the AG size S and requested size S_R.

The joining procedure of AG in case $S > S_R$: In this case, even when the joining node can join, the node will not be able to send all sub-streams. Therefore, in case the AG can have a new Child-AG, we create the new Child-AG with size S_R and let the joining node join this Child-AG. On the other hand, in case the AG can not have new Child-AG, if the AG already has Child-AGs, the AG sends notification to the joining node to join its Child-AG whose size is the closest to S_R. The joining node, which receives such notification message, then sends join request to that Child-AG. In case the AG does not have any Child-AG, since the number of nodes which join AG is too small, we let the joining node to join this AG temporarily. In this case, although all member nodes might not be able to deliver the stream, since the AG does not have a Child-AG and the number of member nodes in this AG is small, only a few members will not be able to send the stream. In addition, we also perform the process that minimizes the AG size and replaces preferentially the temporary node by the node replacement and renewal of the AG size which will be described later.

The joining procedure of AG in case $S \leq S_R$: In this case, we replace nodes only when the minimum value of requested size S_{min} regarding all AG members is greater than S_R. Although this procedure is similar to that of THAG shown in Fig. 4, we use the requested size as the replacement metric instead of using distance between nodes. So that, we can replace preferentially the temporary joining node which is joining the AG with a larger size even when the requested size is small.

By repeating the above process, the node can join the AG which can accommodate the requested size. This procedure is summarized in Fig. 5.

In NHAG, each AG entrance sends the maximum requested size S_{max} in the AG members to its Parent-AG periodically.

1930-529X/07/$25.00 © 2007 IEEE
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE GLOBECOM 2007 proceedings.

When a node leaves, if $S \leq S_{max}$, the AG entrance to which the leaving node belongs sends a notification message to the Child-AG to which the node with S_{max} belongs. The Child-AG which receives the notification message then promotes the node, which has the requested size S_{max}, to its Parent-AG. Thus, we can promote the node from Child-AG which has the requested size equal or more than S even when the node is leaving.

B. Renewal of AG size

Since nodes in AG are frequently replaced due to the joining and leaving of nodes, it is necessary to renew the AG size dynamically according to the network state. Therefore, we recompute the AG size when there is joining and leaving as well as the replacement of nodes.

First, we compute the average requested size S_{avg} of N nodes joining AG. In case the AG does not have a Child-AG, the new AG size is uploaded as follows:

$$S_{new} = \begin{cases} S_{avg} & \mathrm{N} \leq \mathrm{S_{avg}(S_{avg}-1)} \\ S & \mathrm{N} > \mathrm{S_{avg}(S_{avg}-1)} \end{cases} \quad (3)$$

where S is the current AG size. In AG with size S, $S(S-1)$ number of nodes can participate. So, in case $S_{avg} < S$ and $N > S_{avg}(S_{avg}-1)$, some AG members are not joining the current AG. Therefore, in case $N > S_{avg}(S_{avg}-1)$, S is not changed.

On the other hand, in case the AG has Child-AG, if we vary S drastically, a stream might not be sent to the Child-AGs, and hence it is critical to vary S slowly. In THAG where AG has size S, $S-2$ numbers of streams can be delivered. If we suddenly increase the AG size S, nodes within AG will have to send more sub-streams. As a result, the number of Child-AGs, which can be created, will decrease. Therefore, we propose to increase S in case $S_{avg} > S$, and the number of joining nodes in AG is upper limited by $N = S(S-1)$. Furthermore, in case the number of nodes decreases even AG has a Child-AG, since $S > S_{max}$, nodes will not be promoted from the Child-AGs. Therefore, in our approach, if the number of AG nodes has decreased to some extent ($N < (S-2)(S-2)$), we decrease S as well.

Therefore, the new size of AG is set as follows:

$$S_{new} = \begin{cases} S+1 & \mathrm{S_{avg} > S \text{ and } N = S(S-1)} \\ S-1 & \mathrm{N < (S-1)(S-2)} \\ S & \text{others} \end{cases} \quad (4)$$

IV. Performance Evaluation

A. Simulation setup

To validate NHAG, we have performed simulations of streaming over the multicast trees constructed based on ALM by using ns-2 [16]. In our simulations, the transit-stub topology created by the GT-ITM [17] tool was used as the underlying network topology ; it consists of 125 routers and 240 hosts. We set the upload bandwidth randomly distributed between 2 and $5Mbps$, and set the download bandwidth to be $10Mbps$.

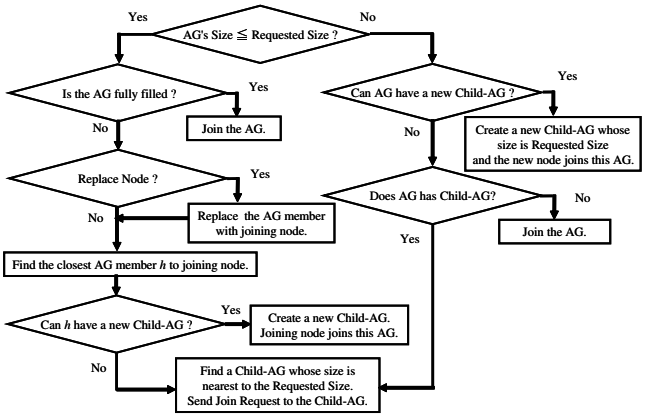

Fig. 5. Node joining procedure of NHAG

Streaming is delivered in a scenario where each node joins the ALM one by one in every 2 seconds, and after all nodes have joined, a node leaves one by one in every 2 seconds. In the simulation, the stream delivery rate is $500kbps$, and the total simulation time is 1000 seconds.

As described in Sections II and III, in THAG, the congestion level increases as the size of AG increases. On the other hand, in NHAG, we vary the AG size adaptively according to the upload bandwidth, and hence the congestion level will not increase even when the AG size increases. In order to validate our analysis, we compare the performance of NHAG with that of THAG for cases when the number of sub-streams is 2, 3, and 4. When the number of sub-streams is 2, 3, and 4, THAG uses AG with size 4, 5, and 6, respectively. We refer to THAG with AG size of S as THAGS. Here, in THAG4, the minimum of upload bandwidth needed is $2Mbps$, and hence there is no congestion. While in THAG5 and THAG6, the minimum upload bandwidth needed is $3Mbps$ and $4Mbps$, respectively, thus being subject to congestion.

We adopt the total throughput and Bandwidth Satisfaction Rate (BSR) as the performance metrics. BSR is the ratio between the requested streaming rate and the received streaming rate, as defined in Eq. (5).

$$BSR = \frac{The\ Received\ Streaming\ Rate}{The\ Requested\ Stream\ Rate} \quad (5)$$

The requested streaming rate is determined based on the size of AG, in which the rate in THAG4 is $1Mbps$, in THAG5 $1.5Mbps$, and in THAG6 $2Mbps$. In NHAG, we determine the requested streaming rate dynamically based on each node's upload bandwidth.

B. Simulation results

Fig. 6 shows the comparison between the average of total throughput and the minimum total throughput. The total throughput is the total of all s1ub-streams' throughputs. In Fig. 6, we can see that NHAG provides higher average total throughput than that of THAG. Furthermore, by comparing the minimum total throughput, in THAG, when the number of sub-streams is 2, all nodes can receive almost all the two sub-streams from the source. Furthermore, when the number

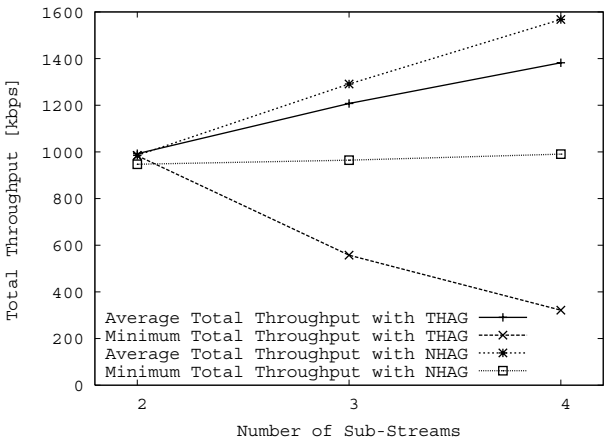

Fig. 6. Total Throughput

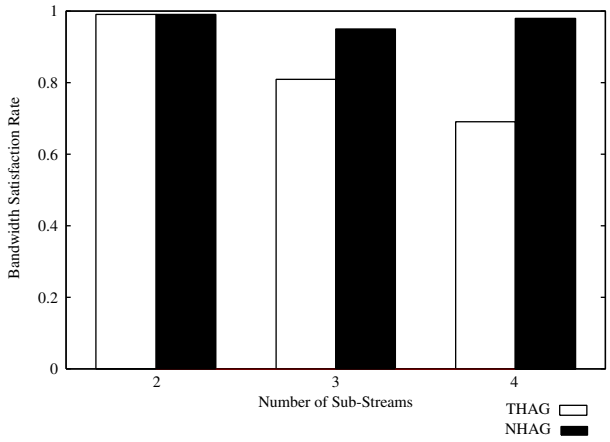

Fig. 7. Average Bandwidth Satisfaction Rate

of sub-streams is 3 and 4, we found that some nodes cannot receive all sub-streams. In NHAG, all nodes can receive almost all sub-streams based on the actual upload bandwidth.

Next, we compare BSR achievable by each method. We calculate the average value of BSRs in each node, or shown in Fig. 7. In THAG, when the number of sub-streams is 2, the BSR is almost 1 because this is an ideal case. When the number of sub-streams is 3 and 4, the average BSR is smaller than 1. On the other hand, NHAG can achieve the average BSR of almost 1, close to the ideal case.

These simulation results show that in THAG the received streaming rate is less than the requested streaming rate as the number of sub-streams increases. NHAG selects the AG appropriately based on the upload bandwidth. In fact, NHAG can achieve almost the same throughput comparable to the ideal case, and deliver the stream adaptively based on the available bandwidth.

V. Conclusion

In this paper, we focus on the ALM, which splits a stream into several sub-streams with MDC and delivers each stream along multicast trees constructed independently from AG, and propose a method based on the available bandwidth. When a stream is delivered on ALM, the proposed method adapts the size of AG according to the available bandwidth. Our simulation results by using network simulator ns-2 have demonstrated the effectiveness of our proposal in terms of the throughput and BSR. Our future works will consider the integration of ALM with other evaluation criteria such as QoS.

Acknowledgement

This work was partially supported through The Strategic International Cooperative Program between JST and NSF.

References

[1] A. M. Hamad and A. E. Kamal, "A Survey of Multicasting Protocols for Broadcast-and-select Single-hop Networks," *IEEE Network*, Vol. 16, No. 4, Jul./Aug. 2002, pp. 36-48.

[2] MBONE, http://www.savetz.com/mbone/.

[3] C. Abad, W. Yurcik, and R. Campbell, "A survey and comparison of end-system overlay multicast solutions suitable for network centric warfare," in *Proceedings of the SPIE*, Vol. 5441, Jul. 2004, pp. 215-226

[4] R. Tian, Q. Zhang, Z. Xiang, Y. Xiong, X. Li, and W. Zhu, "Robust and Efficient Path Diversity in Application-Layer Multicast for Video Streaming," *IEEE Transactions on Circuits and Systems for Video Technology*, Volume 15, No. 8, Aug. 2005, pp. 961-972.

[5] Y. Chu, S. G. Rao, S. Seshan, and H. Zhang, "A Case for End System Multicast," in *Proceedings of the 2000 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, Jun. 2000, pp. 1-12.

[6] P. Francis, "Yoid: Extending the Internet Multicast Architecture," http://www.icir.org/yoid/, April. 2000.

[7] D. Pendarakis, S. Shi, D. Verma, and M. Waldvogel, "ALMI: An Application Level Multicast Infrastructure," in *Proceedings of the 3rd USENIX Symposium on Internet Technologies and Systems*, Mar. 2001, pp. 49-61.

[8] J. Jannotti, D. K. Gifford, K. L. Johnson, M. F. Kaashoek, and J. W. O'Toole Jr. "Overcast: Reliable Multicasting with an Overlay Network," in *Proceedings of the Fourth Symposium on Oerating System Design and Implementation*, Oct. 2000, pp. 197-212.

[9] M. Castro, P. Druschel, A-M. Kermarrec and A. Rowstron, "SCRIBE: A large-scale and decentralized application-level multicast infrastructure," *IEEE Journal on Selected Areas in Communications*, Vol. 20, No. 8, Oct. 2002, pp. 1489-1499.

[10] A. Rowstron, and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems," in *Proceedings of IFIP/ACM International Conference on Distributed Systems Platforms*, Nov. 2001. pp. 329-350.

[11] M. Castro, P. Druschel, A-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "SplitStream: High-bandwidth content distribution in cooperative environments," in *Proceedings of the 19th ACM symposium on Operating systems principles*, Oct. 2003, pp. 298-313.

[12] V. N. Padmanabhan, H. J. Wang, P. A. Chou, and K. Sripanidkulchai, "Distributing streaming media content using cooperative networking," In *Proc. Network and Operating System Support for Digital Audio and Video*, May. 2002, pp. 177-186.

[13] V. K. Goyal, "Multiple description coding: Compression meets the network," *IEEE Signal Processing Magazine*, Vol. 18, No. 5, Sep. 2001, pp. 74-93.

[14] K. Day and A. Tripathi, "Characterization of node disjoint paths in arrangement graphs," Technical Report TR 91-43, Computer Science Department, University of Minnesota, 1991.

[15] T. S. Eugene Ng and Hui Zhang, "Predicting Internet network distance with coordinates-based approaches," In *Proceedings of IEEE INFOCOM*, Jun. 2002. pp. 170-179.

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

[17] E. Zegura, K. Calvert, and S. Bhattacharjee, "How to model an internetwork," in *Proceedings of IEEE INFOCOMM*, Mar. 1996, pp. 594-602.