HostCast: A New Overlay Multicasting Protocol

Zhi Li and Prasant Mohapatra
Department of Computer Science
University of California at Davis
Davis, 95616, CA
{lizhi,prasant}@cs.ucdavis.edu

***Abstract*— Though the merits of IP-based multicast is undeniable, the deployment of IP multicast has met many difficulties. In the past several years, lots of research work have been done on overlay multicast (end-system multicast, application-layer multicast). In this paper, we propose a new overlay multicast protocol: HostCast. Besides constructing a data delivery tree, HostCast uses a simple and efficient approach to form an overlay mesh for control and maintenance. The mesh can effectively facilitate the overlay multicasting. HostCast improves the reliability of overlay multicast tree and decreases the convergence time as demonstrated by the results obtained via simulation.**

I. Introduction

Multicasting protocols can transmit one copy of data traffic to multiple receivers at the same time saving network bandwidth. Many applications involving group communication are inherently suitable for multicasting, which has been an active research topic for several years. However, because of many reasons, such as management, scalability, inter-domain routing, IP multicasting has not been widely deployed[5].

Recently, many researchers have focused their work on overlay multicast. In overlay multicast, the group members self-organize into an overlay multicast tree. The data replication, multicast routing, group management, and other functions are all supported at the application layer. As it does not require changes to the Internet infrastructure, overlay multicasting can be easily deployed. Since the idea was proposed, several overlay multicast routing protocols have been proposed. They primarily differ in multicast tree formation and maintenance[3][7][12].

As the end hosts are not stable and reliable routers, several dynamic factors are of concern in overlay multicast, such as group members' joining or leaving, unexpected halt of end host. These factors greatly affect the stability (reliability) of the multicast tree. One group member's leaving will affect all its descendent nodes, thereby decreasing the tree's stability. The current solution to this problem is: if a group member realizes that it cannot receive the packets from its parent node, it just repeats the join procedure. Using this method, it usually takes a long time for the node to find a new parent and for the overlay tree to converge again.

In overlay multicast, the members need to continuously measure the overlay path conditions to find the best overlay path from the source to itself (best root path). In most of the previously proposed protocols, the members only measure the overlay links between themself and the neighbor nodes in the mesh. So, a member can only realize the condition of the link connecting itself to its parent, not the path connecting itself to the root of the tree (*root path*). When an intermediate overlay link of a root path is broken, the node cannot realize this fact quickly.

This research was supported in part by the National Science Foundation through the grants CCR-0296070 and ANI-0296034.

In this paper, we propose a new overlay multicast routing protocol: HostCast. Our goal is to design a protocol that can accurately measure the overlay path condition and help the group member to find a new parent node quickly when its original parent node is lost. Similar to the tree-first protocols[7], HostCast also uses two steps: setting up an overlay tree and forming a mesh. However, contrary to the previous approaches, HostCast constructs the multicast data delivery tree and control mesh at the same time. Each node in the mesh only has limited number of neighbors, which is enough for the group members to gradually improve the performance of the multicast tree. This can greatly reduce the amount of control traffic. Also, the design of mesh makes the group members measure the root path condition accurately and find a new parent quickly when some group members leave. Therefore, the multicast group can quickly converge again after the departure of group members. Results obtained via simulations have demonstrated the performance benefits of HostCast.

The rest of the paper is organized as follows. In Section II, we will introduce the basic idea of the proposed protocol. The details of the protocol are described in Section III. We evaluate its performance in Section IV, followed by the related work and concluding remarks.

II. HostCast: Protocol Overview

A. Basic Idea of HostCast

HostCast requires that the multicast group has an overlay data delivery tree and a corresponding control mesh, both of which cover all the group members. Based on the idea of overlay multicast, the delivery tree and control mesh only cover the end hosts or multicast servers with no routers. Using HostCast, we can provide multicast service without global IP multicast support.

The data delivery tree is used to deliver the multicast traffic. Each group member has an overlay routing table at the application layer. Whenever a node receives multicast traffic, it duplicates and forwards the traffic to its children members in the data delivery tree. The control mesh of HostCast is used to transmit control messages and overlay path measurement packets. The mesh can also help the group members to gradually improve the data delivery tree and avoid partitioning of the tree.

When a new member joins a multicast group, the overlay links are added to the control mesh as soon as the member finds a parent node in the data tree. Based on the mesh, the member can gradually adjust its position in the data delivery tree to improve the performance. When the data delivery tree is adjusted, the corresponding adjustment is also made for the control mesh.

The mesh links are added or deleted based on the following policies.

1) If two nodes are parent-child relationship in the data delivery tree, there is a coresponding link in the control mesh from the parent node to the child node. The parent node is the child

node's *primary parent* in the mesh. The corresponding root path from the multicast source to the child node via its primary parent is termed as *primary root path*.

2) If a node is another node's grandparent in the data delivery tree, there is an overlay link (we term it as *secondary link*) from the grandparent node to the grandchild node in the control mesh. The grandparent node is the node's *secondary parent* in the mesh. This node is its secondary parent's *secondary child*. The corresponding root path from the multicast source to a node via its secondary parent is termed as *secondary root path*.

3) If a node is another node's uncle in the data tree, there is also a secondary link from the uncle node to this node in the control mesh. The uncle nodes are also the node's secondary parents in the mesh. There are also secondary root paths from the source to the node via its uncle nodes.

Based on above policies, we can observe that the data delivery tree is a subset of the corresponding control mesh. The multicast source is also the root of the control mesh. The overlay path measurement packets are originated by the source and broadcasted to all the group members along the mesh periodically.

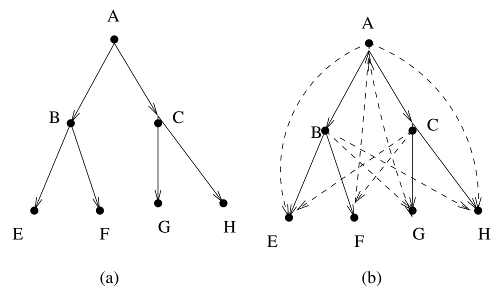

Fig. 1. A Multicast Group's Data Tree and its Control Mesh.

Figure 1 shows an example data delivery tree (a) and the corresponding control mesh (b). In Figure 1 (b), the solid lines connect the group members with their primary parents and dotted lines connect the group members with their secondary parents. For example, link A-E and C-E are E's secondary links which connect E to its secondary parents A and C. The paths A-C-E and A-E are E's secondary root paths while the path A-B-E is E's primary root path.

B. Measurement-based Selection of Overlay Paths

Because HostCast runs at the application layer, it cannot easily retrieve the underlying network topology and QoS metrics, such as end-to-end delay, available bandwidth, etc. However, to provide QoS to multicast applications, HostCast uses a measurement-based approach to derive the overlay path QoS conditions and provide QoS to multicast users.

To gradually find better root paths for group members while maintaining scalability, we use the following path estimation method. The root node (multicast source) periodically sends small fixed size (m bytes) probe packets to its children nodes in the mesh. If a node receives a probe packet from its primary parent, it will duplicate the packet and forward it to its children nodes (both primary children and secondary children). Otherwise, it will not forward packets to it children nodes. Using this approach, the group members can accurately measure the QoS conditions of their root paths (both primary and secondary). If needed, it can change its primary parent (switching parent node in the data delivery tree).

The basic theory of measurement-based approach is as follows. Each probe packet carries the time (t_1) when the multicast source sends out the packet . When one node receives the packet at time t_2, we can assume the delay D from the source to the node is $t_2 - t_1$ while the available bandwidth B is $m/(t_2 - t_1)$, assuming that the time is synchronized across all the nodes.

Suppose before one node receives the i_{th} measurement packet, the previous measurement result of a root path's bandwidth is b_{i-1} and the delay is d_{i-1}. Let the i_{th} packet (whose size is m_i, carrying time t_{i1}) following the same root path arrives the node at time t_{i2}, we use the following weighted average equations to update the measurement result.

The root path's available bandwidth is:

$$b_i = \alpha * m_i/(t_{i2} - t_{i1}) + (1 - \alpha) * b_{i-1} \tag{1}$$

The root path's delay is:

$$d_i = \alpha * (t_{i2} - t_{i1}) + (1 - \alpha) * d_{i-1} \tag{2}$$

We use the following equations to evaluate the bandwidth variance tendency (Vb_i) and delay variance tendency (Vd_i).

$$Vb_i = \alpha * (b_i - b_{i-1}) + (1 - \alpha) * Vb_{i-1} \tag{3}$$

$$Vd_i = \alpha * (d_{i-1} - d_i) + (1 - \alpha) * Vd_{i-1} \tag{4}$$

The idea of above equations is similar to the RTT estimation in TCP. In the above equations, α is the constant weighting factor, where $0 < \alpha < 1$. Choosing a value of α close to 0 makes the weighted average immune to the change over short time. Choosing a value close to 1 makes the average value quickly respond to the temporary changes.

When choosing an overlay root path, even when b_i and d_i meet our requirement, it is desirable for Vb_i and Vd_i to both have positive values. Vb_i and Vd_i represents the variance trend of the path's QoS metrics. If the values are positive, it would indicate that the path is likely to be more stable in future.

III. HostCast: Detail Description

A. Constructing the Data Tree and Control mesh

HostCast requires the group members to form a data tree as well as a control mesh. The data tree is used to deliver the multicast traffic while the control mesh is used to improve the performance of the data tree. At each group member, it uses separate data structures to maintain the tree and the mesh.

It assumes that the group members can use some mechanism (such as a Rendezvous Point) to get the multicast source before they join the group. Then, based on HostCast, the group members can self-organize into the source-based overlay tree. One of the main functions of HostCast is to help the new member find a primary parent in the data delivery tree. A new member needs to maintain the following lists: Potential Parent List (PPL) and Non-potential Parent List (NPL). The PPL includes the current potential parents. NPL is the list of the potential parents that have denied to accept the new member as a child node. The non-potential nodes are sorted in NPL based on their distances (end-to-end delay) to the new member.

Suppose N wants to join a group G, it first adds the multicast source S to PPL and sends a join request message (JREQ) to S. When an on-tree node Y (which can be any group member, including the root node) receives a JREQ from X, it uses the Algorithm 1 to process the join request.

After sending a JREQ, the new member can use Algorithm 2 to continue searching for a primary parent node.

Algorithm 1 **Process of JREQ**

```
If its outdegree permits it to accept another primary child and X is not in its primary root path
    {Add X to this primary child list in the data tree and control mesh.
    Request its parent node to add X to its secondary children list.
        /*when Y's parent node receives this, it will add the corresponding secondary links and request X' uncle nodes to update the links.
    send JACP to X /*JACP is join acceptance message.*/
    }Else
    send JREJ to X with Y's primary children list /*JREJ is join rejection message.*/
```

Algorithm 2 **Primary Parent Node Searching**

```
If A JACP is received with source X
    {If it has no primary parent node
        Add X as its primary parent node
        Else
          send LEAVE message to X
        /*when X receives LEAVE, it will remove the corresponding overlay links in the data tree and control mesh */
    }Else
If A JREJ is received with source X and its children nodes
    {Remove X from PPL
    Add X with its children nodes to NPL
    }
if (PPL == NULL)
    {Retrieve the first node in NPL (the nearest non-potential node)
     Add its children nodes to PPL
     send JREQ to the nodes in PPL}
```

In HostCast, the membership is soft-state based, which means that a node needs to periodically send a REFRESH message to its parent nodes to maintain its membership. Otherwise, the corresponding overlay links in the data tree and control mesh will be removed. To avoid loops in the data tree, each member needs to keep the member list of its primary root path.

Based on the above algorithms, a new member can find its position in the data tree as well as in the mesh. The corresponding overlay links in the data tree and control mesh are also added at the same time. Then, based on the methods discussed in the following sections, it can gradually improve the performance of the primary root path.

B. Improve the Performance of Data Tree

With the joining of new members and the departure of old members, the data delivery tree can easily become inefficient. To improve performance, it is necessary to refine the data tree based on the dynamic network situation. The improvement is based on root paths (primary root path and secondary root path) measurement results. HostCast uses the following two methods to refine the data delivery tree.

1) Switch Primary Parent Node: Based on the overlay path measurement results, if a member realizes that a secondary root path can potentially provide better QoS than its current primary root path and the secondary parent can accept one more child, it will switch its primary parent and the secondary parent. To maintain the tree stability and avoid unnecessary switches, we can use some thresholds to determine whether the switching should be done or not.

The measurement results cannot always accurately reflect the real traffic situation of the overlay path. To maintain the stability of the data tree, we use the following method to switch the primary parent node. It first sends a JREQ to the secondary parent node and become a primary child node of this new parent. For a short period, the node has two primary parents in the data delivery tree. After a while, if it is ensured that the new primary parent can provide better service than the previous one, it sends a LEAVE message to the original primary parent to adjust the corresponding overlay links. Otherwise, it sends a LEAVE message to the new parent.

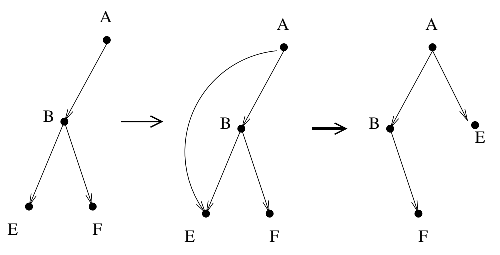

Fig. 2. Switch primary parent node to original grandparent node.

Figure 2 shows an example of data tree evolution process that node switch its primary parent node to its original grandparent.

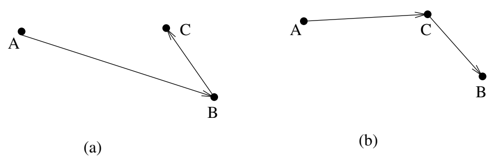

Fig. 3. Triangle Problem.

2) Substitute Primary Parent Node: One of the frequent problems in overlay tree is the "triangle problem". We use the example shown in Figure 3 to explain this problem. In 3 (a), A is the parent node of node B whose child is C. It is easy to note that if the tree is reorganized as 3 (b), it will be more efficient and save network resource. HostCast tries to avoid the "triangle problem" to improve the efficiency of data delivery tree.

If a node realizes the existence of the following situation, it will actively begin the substituting process.

1) The secondary root path via its grandparent node has less delay than its current primary parent's root path;

2) The node can accept another child node;

3) The node has larger number of descendents in the data tree compared to its parent node (except the descendents which belong the current node.)[1]

The substitution process works as follows.

1) The node sends substitute request to its grandparent to adjust the overlay links in the data tree and control mesh;

2) After receiving confirmation, it asks its primary parent to remove the corresponding overlay links;

3) At the same time, it adds the overlay links ensuring that the original parent becomes one of its primary children nodes.

C. Recovery From Data Tree Partition

If one node suddenly leaves the multicast group, its children nodes and other descendent nodes will become partitioned from the data tree. The recovery latency from data tree partition will determine the amount of multicast traffic lost for these group members.

To shorten the recovery latency, HostCast uses the following two methods at the same time to recover from a partition. If

[1]In HostCast, we assume that each nodes will propagate its descendent number to it parent nodes. Thus, all the nodes will realize their number of descendents.

one node realizes the loss of its primary parent node, it can send JREQ to its secondary parents in the control mesh hoping that one of them can become its primary parent. At the same time, it randomly picks up some nodes in its primary root path as the candidate parent nodes and sends JREQ to them.

To increase the chance of secondary parents' becoming primary parent, we propose an enhanced version of HostCast. In the enhanced HostCast, for each node, its secondary parents not only include its grandparent node and uncle nodes, it also includes the parent of grandparent and its other children and other grandchildren nodes. This method can shorten the latency of finding a new primary parent and speed up the data tree improvement process.

IV. Simulation Experiments

We designed and developed an event-driven packet-level simulator to evaluate the protocol. The simulations are based on the Waxman network topology[11] generated by the Georgia Tech Internetwork Topology Models (GT-ITM)[1]. The network nodes are randomly chosen in a square ($\alpha * \alpha$) grid. A link exists between the nodes u and v with the probability $P(u, v) = a * e^{-d(u,v)/(b*\alpha^2)}$, where d(u,v) is the geometric distance between u and v, a and b are constants that are less than 1. In the simulation, $a = 0.4$, $b = 0.3$, and $\alpha = 1000$. Using these parameters, we generate a random backbone topology with 1024 nodes and 2284 links. The link delay between any two neighbor nodes varies between 2 and 5 ms. In the topology, each member is directly connected to one of the backbone nodes with 1-3ms latency. The fanout of each member varies from 3 to 6, which determines the number of children nodes it can have for a multicast group. For each simulation, each member randomly selects one backbone node to attach. For each value of the following results, we executed 100 runs of the simulation and computed the average results.

The performance measures we evaluated during the simulations are :

1) Relative Tree Cost Penalty (RTCP): As we know, the overlay trees are set up at the application layer. The routing protocol does not have the underlying network topology information. The overlay multicast tree is not as efficient as IP-based multicast tree. The *RTCP* is defined as follows:

$$RTCP = \frac{Cost\ of\ multicast\ tree\ (Hops)}{Cost\ of\ Shortest\ Path\ Tree\ (Hops)} \tag{5}$$

In the simulation, we compare the RTCP value of unicast star[2] with overlay tree constructed by HostCast. We repeated the experiment with various group size. For each delivery tree, we randomly picked some backbone nodes to attach the group members. Then, we randomly selected one of the group members as the multicast source. Using the multicast routing protocols, the other members join the multicast group one by one. After the multicast trees get stabilized, we estimated the RTCP value.

Figure 4 shows the simulation results. From the results, we can observe that HostCast has lower RTCP (ranges from 1.3 to 1.6) compared to the unicast star approach. At the same time, the RTCP value increases slowly with the increase in group size.

2) Average Relative Delay Penalty (ARDP): In overlay multicast, the traffic is routed and duplicated at the end hosts. So, the paths that the multicast traffic travels are usually longer than the shortest path from the source to the destinations. This will

[2]The source unicasts the multicast traffic to the receivers one by one.

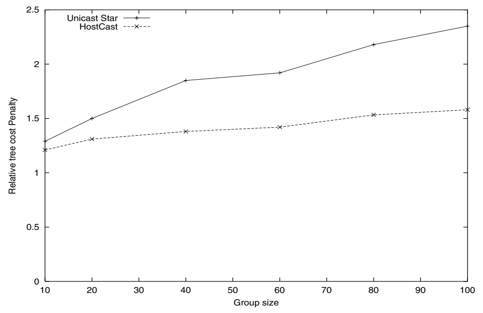

Fig. 4. Relative Tree Cost Penalty.

incur extra delay for group members. We use Relative Delay Penalty (RDP) to evaluate this aspect performance. RDP is defined as follows.

$$RDP = \frac{Primary\ root\ path\ delay\ in\ overlay\ routing\ tree}{Root\ path\ delay\ in\ shortest\ path\ tree} \tag{6}$$

Similar to the evaluation of RTCP, after the overlay tree becomes stable, we estimate the group members' relative delay penalty and average them to get the ARDP. Figure 5 shows the simulation results with varying multicast group size. The results reveal that ARDP increases with the increase in group size. For group with size of 10, ARDP value is around 1.5, while group with size 100 has the value of 2.8. This is because that each group member has limited fan-out value. When the group size increases, the multicast traffic will pass through more intermediate end hosts before arriving at the leaf nodes.

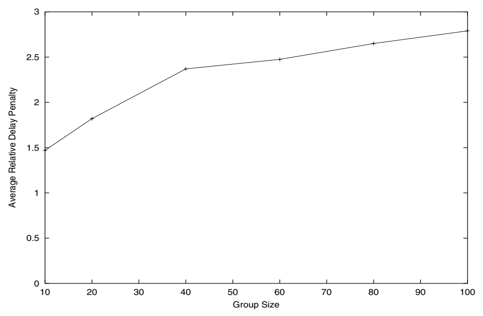

Fig. 5. Average Relative Delay Penalty.

3) Average Recovery Latency (ARL): With the group members' departure, their children nodes will take some time to find new parents and stabilize. This latency determines the amount of traffic the members will lose. ARL is used to evaluate this aspect performance. One of the merits of HostCast is that each member has several secondary parents. When it realizes that its primary parent node dies or leaves, it can quickly pick one of secondary parents to substitute its primary parent node, if possible. Other tree-based overlay protocols do not have the concept of secondary parents, so they usually take longer time to find a new parent node.

In the simulation, we first use HostCast to set up an overlay multicast tree with 100 members. After the tree stabilize, we randomly pick some group members to leave the group at the same time. Then, the children nodes of this leaving nodes begin to search for a new parent. We vary the number of departure nodes and evaluate the ARL. Figure 6 shows the simulation results. We can observe that the secondary paths can greatly decrease the recovery latency. When less number of group members depart at the same time, HostCast has more advantage because the secondary parents have more chance to accept new members. However, when more members leave at the same time, the secondary parents may also belong to the leaving members set. For the same reason, as the number of leaving members increases, the advantage of the enhanced HostCast over the basic HostCast also decreases.

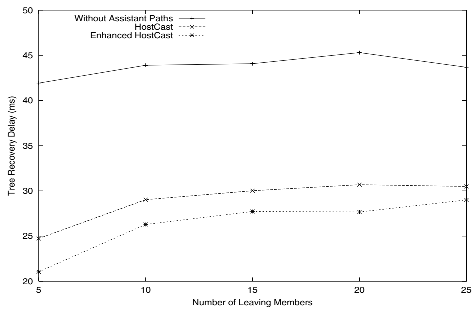

Fig. 6. Average Tree Recovery Delay.

4) Convergence Time Under Dynamic Environment: With the group membership's change, the overlay tree will go through an unstable period. The convergence time is used to evaluate the response time during the dynamic situation.

To demonstrate this effect, we insert some dynamic events during simulation and observe how long it takes for the overlay multicast group to stabilize. We use a variety of changes in the overlay multicast tree to evaluate the stability. The changes in the overlay tree includes switching overlay links, adding links and deleting overlay links. The series of dynamic events include: at time 0, 50 node join the group; after 1200 ms, 10 other nodes join the multicast group again; again 1200 ms later, 10 of the 60 group members leave the multicast group at the same time. Figure 7 shows the cumulative number of changes with time and with different probe packet intervals. It can be observed that with shorter interval time, the convergence time is shorter which means that the overlay tree can stabilize quickly again.

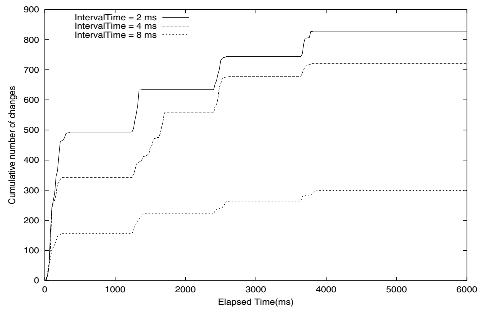

Fig. 7. Changes in the Overlay Topology.

V. Related Works

Several overlay multicast routing protocols have been proposed. They can be categorized as tree-first approach, mesh-first approach or centralized approach.

Narada [3][4] uses mesh-first approach, in which all the group members first form a mesh. Based on this mesh and using some routing protocol, it constructs an overlay multicast tree covering all the group members. As each of group member maintains the list of all (or partial) the group members and keeps sending probe packets to measure the inter-member distance, this solution is not scalable to large multicast groups.

In contrast, the tree-first approach idea is more scalable. The members directly set up an overlay multicast tree which connect all the members. Then, based on some mechanisms (such as using mesh), they make the overlay tree more robust to the dynamic group membership. Though several tree-first based approaches have been proposed[7][12][8], none of the papers have discussed how to form an efficient mesh when facing the dynamic changes in the network. In Yoid[7], the group members randomly select several other members as mesh neighbors to avoid partition. While in Host multicast[12], it does not construct any mesh. The multicast tree improvement is done by continuously repeating the join process.

In the centralized approach, a fixed node is used to control the membership information of a whole multicast group, helping the group members to form an efficient overlay multicast tree [6][9]. It is easy to see that the fixed node is the bottleneck and its not scalable to large size group.

Some other efforts have proposed hierarchical approaches[2][10], in which the multi-level techniques are used for overlay tree construction. In [13][14], the authors have retrieved underlying network topology information to help the construction of overlay multicast tree.

VI. Conclusions

In this paper, we propose a new overlay multicast routing protocol: HostCast. To simplify the overlay mesh construction, HostCast forms the overlay trees and their corresponding meshes at the same time. The construction of the mesh makes sure that each group member has several secondary parents in addition to the primary parent node. This design makes the group members accurately measure the QoS metric of the overlay root paths while maintaining scalability (reducing the number of probe messages). At the same time, it speeds up the new parent searching process when group members' parent nodes leave. The simulations results have shown that HostCast can achieve good performance in terms of relative delay penalty, tree cost penalty, and the convergence time.

References

[1] GT-ITM: Modeling topology of large internetworks, http://www.cc.gatech.edu/projects/gtitm/

[2] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, Scalable application layer multicast, in Proceedings of ACM Sigcomm 2002, 2002.

[3] Y. Chu, S. Rao, and H. Zhang, A case for end system multicast. In *Proceedings of ACM Sigmetrics*, June 2000.

[4] Y. Chu, S. G. Rao, S. Seshan and H. Zhang, Enabling Conferencing Applications on the Internet Using an Overlay Multicast Architecture, *ACM SIGCOMM 2001.*

[5] C. Diot, B. N. Levine, B. Lyles, H. Kassem and D. Balensiefen, Deployment issues for the ip multicast service and architecture. *IEEE Network*, vol.14, num 1, 2000.

[6] D.Pendarakis, S.Shi, D.Verma, and M.Waldvogel, Almi: An application level multicast infrastructure. In *3rd Usenix Symposium on Internet Technologies and Systems*, March 2001.

[7] P. Francis. Yoid: Extending the internet multicast architecture, 1999. White paper http://www.icir.org/yoid/.

[8] L.Mathy, R.Canonico, and D.Hutchison, An overlay tree building protocol, In *NGC 2001*, 2001.

[9] V. Roca and A. El-Sayed, A Host-Based Multicast (HBM) Solution for Group Communications, 1st IEEE International Conference on Networking (ICN'01), Colmar, France, July 2001

[10] S. Y. Shi and J.S.Turner. Routing in overlay multicast networks, In *Infocom'02*, June 2002.

[11] B. M. Waxman. Routing of Multipoint Connections, *IEEE Journal on Selected Areas in Communications*, Dec. 1988.

[12] B. Zhang, S. Jamin, and L. Zhang, Host multicast: A framework for delivering multicast to end users. In *Infocom'02*, June 2002.

[13] S. Ratnasamy, M. Handley, R. Karp, S. Shenker, Topologically-Aware Overlay Construction and Server Selection, Proceeding of Infocom 2002

[14] M. Kwon and S. Fahmy, Topology-Aware Overlay Networks for Group Communication, NOSSDAV'02,