On the Performance of an Application Layer Multicast Protocol

Xiaobing Hou, Shuju Wu
Central Connecticut State University
xhou@ccsu.edu, swu@ccsu.edu

Abstract

This paper studies the performance of an application layer multicast protocol, namely Adaptive Overlay Multicast (AOM) protocol. We introduce the concepts of fan-outs and foster limit in building application layer multicast tree, and exemplifies the performance and adaptability characteristics of AOM to network dynamics with extensive simulation results.

Introduction

Overlay multicast (also called application layer multicast) was proposed to provide multicast service at the application layer using p2p connections and remove the dependence on multicast support of the underlying networks [1-4]. Operations such as membership management and routing are implemented at the application layer, and data distribution is over a multicast tree that consists exclusively of end hosts and unicast connections. Both IP and overlay multicast use a tree to achieve distribution efficiency, which results in *performance clusters* when tree links or nodes are under fault. A performance cluster includes those members that suffer the performance degradation caused by the same fault. The larger the cluster size is, the more the group communication is jeopardized. Therefore, tree construction and fault adaptation are important.

The Internet is dynamic and unpredictable in nature. Dynamic events such as group membership changes, node failures, link failures or network congestion can cause the quality of an overlay multicast tree to degrade over time. We term any of these as a *fault*. A fault caused by dynamic group membership or node failure is easier to detect and the effect on application performance is temporal. However, faults caused by network congestion in the Internet could last a much longer time. They cause end-to-end performance degradation without a total loss of connection, and thus cannot be detected by simple node failure detection mechanisms. Experiments on the MBone [5] show that even for a small multicast group of 11 members, each member experiences a very long consecutive loss of up to a few minutes and this loss happens in almost every trace. In [6], link loss rates in a MBone group of 8 members are measured in one-hour long intervals and shown to vary between 2% to 35%. On a specific link, loss rate higher than 15% happens frequently and often lasts about 10 minutes. Also, from the results of Internet measurements [7], [8] it is not unusual to find long-lasting high loss periods between Internet nodes, although the average loss rate over a day could be low. When such faults happen in a multicast tree and are close to the multicast source, the size of the performance cluster will be large which adversely affects most of the group members.

Multicast tree building algorithms employed by different overlay multicast protocols exhibit different scalability and adaptability characteristics under network dynamics during the multicast session time. A protocol may build a well-formed initial overlay multicast tree under stable network conditions but may not be able sustain the application performance with the presence of underlying network perturbs. Multicast protocol performance has been addressed in [9] in the context of traditional reliable IP multicast. Most of the previous overlay multicast protocols focused on the construction of overlay multicast trees. Therefore, the adaptation performance to network dynamics is either passive and limited [3], [4], [10], or not scalable [1], [2]. The adaptation is passive and limited because although a member periodically looks for a new parent in the tree, it does not use end-to-end performance as a guide thus may not help end-to-end application performance.

In [11] and [12], we proposed Adaptive Overlay Multicast (AOM) that employs both end-to-end and local metrics to build the overlay multicast tree. This paper, however, introduces the concepts of fan-outs and foster limit in AOM and studies how they affect the application layer multicast tree quality. The paper is organized as follows. Section 2 introduces the related work. Section 3 describes AOM and its fault adaptation algorithm. Section 4 presents the simulation study on AOM with different fan-outs and compares the tree quality with a well-know application layer multicast tree protocol. Section 5 concludes the paper.

Related Work

The previous overlay multicast studies focus on self-organizing the group members into a delivery tree and can be classified into centralized, distributed direct-tree, and distributed mesh-first approaches. ALMI [13] takes a centralized approach where a central controller builds the overlay and disseminates the tree information to the group members. NARADA [2] and Gossamer [1] build a mesh first and run a DVMRP-like routing protocol to build the tree. NICE [10], YOID [4] build the tree directly, i.e., the tree is extended when a new joining member connects to an existing member. All these protocols use Round Trip Time (RTT) as the building metric. HostCast [14] utilizes the shortest end-to-end delay in path finding, but no effort is given to match the overlay multicast tree to the optimized IP multicast tree. None of the above protocols has investigated the loss adaptation issue.

HMTP [3] is a typical direct-tree protocol using RTT as the only metric. A new member moves as far as possible from the ROOT if only it could find a potential parent closer than the current one and its RTT to the current parent is larger than the potential parent's RTT to its current parent. Periodically, a member randomly selects another member in its path to the ROOT (hereafter, we call it *ROOT path*) and explores the branch under that member for a new parent. The periodic level-by-level exploration and probings among members accounts for most of the overhead in HMTP.

Recent tree building approaches include closest-first-searching (CFS) [15], adjacency matrix [16] and minimum diameter spanning tree [17]. [15] uses an approach similar to HMTP, however, a member tries to remember different branches in the tree building process. The objective is to extend the searching range of node position so that nearby nodes have a better chance to stay together in the tree. The algorithm itself does not consider the tree maintenance and adaptability issue. [16] depends on an adjacency matrix to build the multicast tree, and the matrix information is provided by the underlying P2P architecture. Therefore, the application of [16] must be tied into a P2P network. [17] takes a centralized approach where the multicast source node collects the probing results from every grid node and calculates the minimum spanning tree for multicast file distribution. This approach is expected to be adaptive to network dynamics if continuous probing is applied, however, scalability is the main shortcoming of centralized approach.

[18] studies the overlay multicast protocol in dynamic network environments. Their experiments were carried on a mesh-first protocol, NARADA, and the results show that it is important to adapt delay and bandwidth for conferencing applications. We believe that it is also necessary to study the dynamic adaptation in direct-tree protocols because: first, direct-tree protocols do not have an explicit multicast routing protocol as in NARADA and Gossamer to distribute the helpful information for the adaptation. Second, one of the objectives of direct-tree protocols is scalability. Transient study can help analyze whether a protocols is scalable by adapting to network faults efficiently and in time.

A simple and best-effort approach to improve the data delivery ratio under dynamic network conditions was recently studied in Probabilistic Resilient Multicast (PRM) [19]. The idea is that in addition to the normal data forwarding along the multicast tree, each member randomly chooses a constant number of other members and forwards the new data to each of them with a certain probability. Random forwarding incurs duplicate packets to the members that are fault-free while provides passive loss recovery at the faulty locations. PRM is not a multicast tree protocol but is a best-effort approach to improve the data delivery ratio in overlay multicast.

An Adaptive Overlay Multicast Approach

We first briefly review AOM, which is detailed in [12]. The introduction here focuses on fan-outs, foster limit and tree adaptation .

AOM Tree Protocol

For scalability, AOM tree protocol takes the direct-tree approach. The tree protocol fulfills the following tasks: tree formation, tree improvement, membership management, loop avoidance, detection and resolution.

Definition of End-to-End Delay and Local RTT

Most of the previous tree construction protocols only use Round Trip Time (RTT) to local neighbors (referred as local RTT) to connect the members. AOM, on the other hand, uses both End-to-End Delay (EED) to the ROOT and RTTs between the members to determine how to construct the tree. A a member i's EED to the ROOT is defined as the sum of its parent's EED and half of the RTT between member i and its parent:

$$EED_{i,ROOT} = EED_{i' sparent,ROOT} + 0.5 \cdot RTT_{i,i' sparent}$$

A member measures its RTT to another member by periodically sending PROBE message. The measurements are smoothed with exponential averaging. To calculate the EED, a parent puts its current EED in the PROBEREPLY message (the ROOT's EED is 0) and a child updates its own EED as follows:

$$EED_{new} = EED_{parent} + 0.5 \cdot RTT_{new}$$

Tree Formation

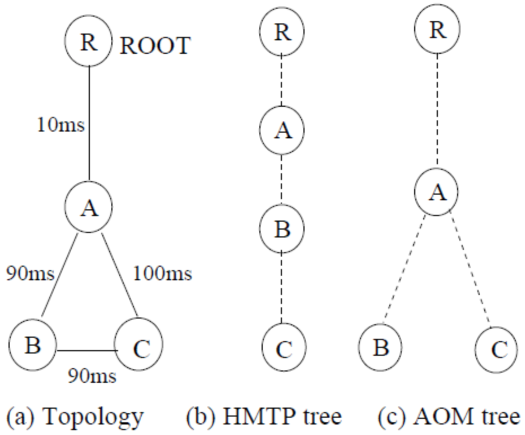

Figure 1: Example of multicast trees

A member joins the group by sending a JOIN message to the ROOT. If the ROOT can accommodate the new member it sends an ACCEPT message with the information of its current children. The new member then starts looking for the most suitable parent. In AOM tree protocol, the more suitable parent for a member is one that is closer to the member than the current parent (i.e., smaller RTT), is closer to the ROOT than the member (i.e., smaller EED), and through which the member's new EED is not penalized too much.

The reason for AOM to use both of EED and RTT as metrics is that EED reflects the *vertical* distance of a member to the ROOT, but RTT reflects the *local* distance between a member and its neighbor, and both of them need to be considered to make the tree match the underlying network topology better. Figure 1 shows a simple network topology in (a), the tree built by HMTP [3] that only uses RTT as the metric in (b) and the tree built by AOM in (c), assuming the joining sequence of A, B and C.

To limit the joining overhead, a member looks for a new parent from a *potential set* of parents. Assume the ROOT is at level 0. A member is at level i if its parent is at level $i-1$, and we say they are 1 overlay hop away from each other. The *potential set* of a level i member is $\{ member_{l,h} \}$ where $l = i$ and $h <= 4$. l is the level and h is the number of overlay hops from this member. The potential set is obtained by probing the *ancestors* (introduced in Section3.1.3). If a level i member finds a new parent in the potential set, its level becomes $i+1$. Parent searching continues until no new parent can be found. A nice property of such a potential set ($h <= 4$) is that when a member initially joins the group and is at level 1 or level 2, it has the opportunity to explore its position in all the tree branches, but when it moves further down the tree, the searching is limited to sub-branches.

Tree Improvement

Due to independent joining sequences and dynamic membership, it is necessary for the members to periodically re-evaluate their positions and continue to optimize the tree structure after joining the group. Since topologically close members are likely to stay close in the overlay by using both EED and RTT metrics, the tree improvement is carried locally, i.e., a member only contacts its *ancestors* for improvement to reduce overhead. The *ancestor set* of a level i member is $\{ member_{l,h} \}$, where $l \in \{ i, i-1, i-2 \}$ and $h \in \{1,2,3\}$.

Members Leaving

A single member leaving will cause the tree to become partitioned. Therefore, before a member leaves the group, it notifies its parent and children. Each child then chooses the closest ancestor (minimum RTT) or ROOT as new parent. Partitions caused by an unexpected member or link fault are detected either by the fault adaptation algorithm or by continuous loss of the PROBEREPLY messages.

Loop Detection and Avoidance

The simplest way to resolve the loop is to let each member attach its ROOT path information in the PROBE and PROBEREPLY messages. A member detects the loop by finding itself in the middle of its ROOT path and breaks the loop by re-joining the ROOT.

Performance Monitoring and Fault Detection

The previous direct-tree protocols, including HMTP, do not actively monitor end-to-end performance metrics. Therefore, they adapt to only local delay conditions as dictated by RTT increases. In AOM, a member monitors the performance of not only its current ROOT path, but also the paths through its ancestors (backup paths). Therefore, when a fault happens on the ROOT path, the member can select a backup path with better performance for its performance cluster. Currently, end-to-end delay and end-to-end loss rate are used as performance metrics for AOM.

Monitoring the EED

A member monitors the EED on its ROOT path by periodically probing its parent. The EEDs on the backup paths are measured in the same way but less frequently because no other important information is exchanged on these paths. To prevent instability problem, the periodical measurements are smoothed with exponential averaging.

Monitoring the Loss Rate

Since the ROOT path is used for data distribution, its loss rate $l_{m,ROOT}$ can be measured by the application data. Loss rate on a backup path $l_{m,a,ROOT}$ is calculated as follows:

$$l_{m,a,ROOT} = 1-(1-l_{a,ROOT})*(1-l_{m,a})$$

where $l_{a,ROOT}$ is the loss rate on the ancestor a's ROOT path and $l_{m,a}$ is the loss rate on the overlay link between the member and the ancestor a. Since there is no application data on this link, the member asks the ancestor to periodically send a *test packet*.

Loss measurement of $l_{m,a}$ is a variation of the Average Loss Interval (ALI) method [20]. ALI is a better loss rate estimator than the Dynamic History Window (DHW) used by RON [21] and the Exponentially Weighted Moving Average (EWMA). ALI properly considers the effects of both recent and earlier loss events. Like ALI, AOM uses the weighted average loss over a few measurement intervals; but unlike in ALI where the intervals are decided by every single loss event, the intervals in AOM are of equal lengths. This is because the two methods serve different purposes: ALI works for TCP-friendly congestion control and expects the sender to respond to every loss event, but our purpose is to detect the loss rate over a period of time. For details of the algorithm, please refer to [12].

Fault Adaptation

The Approach

Without an adaptation algorithm, the only chance for protocols such as HMTP to recover from an EED fault is the periodical improvement, where RTT is used to look for a closer member. Therefore, a member can bypass the EED fault if it finds a closer member not suffering the fault. Since a closer member is not necessarily an EED fault-free member, the result is random. In addition, a faulty link may affect many members' EEDs but not local RTTs, resulting no switching efforts at all. In AOM, once the EED fault is detected, a member actively probes the ancestors for the most up-to-date EEDs and loss rates, and starts the fault adpatation algorithm as summarized in Table 1.

Table 1: Fault adaptation algorithm

```
1.  on detection of faults at member $m$:
    probe $S_a = \{ROOT, ancestors\}$ for RTTs, EEDs and loss rates.
2.  wait for reply, then update performance metrics through ancestor $a$ in $S_a$ as:
    $RTT_{new} = \beta \cdot RTT_{new} + (1-\beta) \cdot RTT_{old}$;
    $EED_{m,a,ROOT} = EED_{a,ROOT} + RTT_{new,a}/2$;
3.  add $a$ to potential parent list $pl$ if:
    $EED_{m,a,ROOT} < scale \cdot EED\_LIMIT$ &&
    $l_{m,a,ROOT} < scale \cdot LOSS\_LIMIT$
4.  find closest potential parent:
    $cur\_potential\_parent = min\_rtt(pl)$;
    if $cur\_potential\_parent == NULL$, adaptation fails, end.
    $pl = delete(pl, cur\_potential\_parent)$;
    join( $cur\_potential\_parent$ ).
5.  if not accepted by $cur\_potential\_parent$, go to step 4.
```

It is worth pointing out the difference between tree improvement and fault adaptation. Both of them involve looking for new parents. However, the tree improvement process creates a more efficient tree, while the fault adaptation process satisfies the performance requirement.

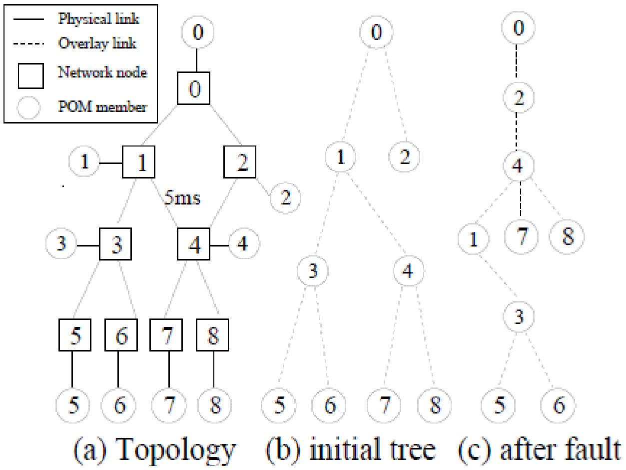

Figure 2: A 9-node network (a), inital tree (b), tree after adaptation (c)

Benefits of end-to-end performance monitoring

In this section, we show the benefits of using EED over solely using RTT to adapt to network faults. For better clarification, both cases are simulated in AOM. However, the case of using local metric will apply to other protocols like HMTP. Figure 2 shows a 9-node network topology, the overlay multicast tree before the fault happens and the new overlay multicast tree when end-to-end delay is used as fault adaptation metric in AOM. Every physical link has a delay of $10ms$ except that link *14* has the delay of $5ms$. This is to make sure that initially, member 4 selects member 1 as its parent in the tree. At simulation time 50, the delay of link *01* increases to 1.2 seconds, causing member 1, 3, 4, 5, 6, 7 and 8 suffering large end-to-end delays. The total simulation time is 180 seconds.

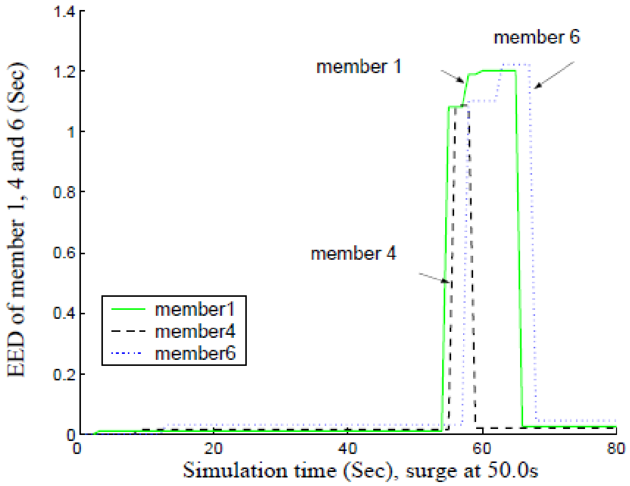

Figure 3: Adaptation of EED using end-to-end metric

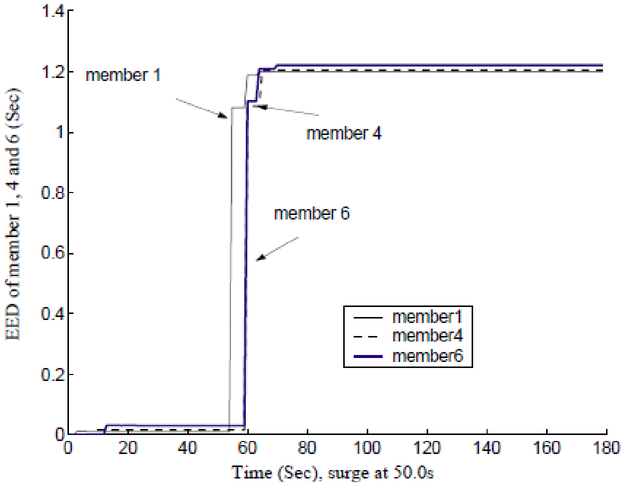

Figure 4: Adaptation of EED using local RTT

Figure 3 shows the simulation results of member 1, 4 and 6. It can be seen that member 4 (thus its children 7 and 8) changes its path before member 1 and recovers from the fault by attaching to member 2. Member 1 could not adapt to the fault by itself as the underlying routing algorithm happens to use the faulty link to probe member 2. However, member 4 invites member 1 after it switches its sub-cluster to member 2. From the figure, we also see that member 6 adapts to the fault. However, this is not because member 6 selects a new path but because its grandparent, member 1, changes the ROOT path to a better position and therefore the sub-cluster (5 and 6) recovers from the fault without extra probing and adaptation overhead. On the contrary, Figure 4 shows that if only using RTT as the performance metric, none of member 1, 4 or 6 could adapt to the delay fault although better paths exist, and the multicast tree does not change at all.

Fan-outs and Foster Limit

End-hosts over the Internet are heterogeneous; therefore, the maximum number of unicast connections that can be set up to forward the application data depends on factors such as bandwidth capacity, traffic load and host processing power, and may vary from time to time. This connection limit is called a *fan-out* limit.

A member's fan-outs should be those that are best for the tree quality. If a new connection request is simply refused when the fan-out size hits the limit, the resulting tree may have inferior quality. To solve this problem, the connections accepted by a member are classified as fan-out connections and foster connections. The fan-out connections are used to forward application data and are restricted by the fan-out limit. The foster connections are used to construct the tree and are restricted by a foster limit. If a connection request cannot be treated as fan-out connection, it is accepted as a foster connection for a period of time. Since the control packets used for tree construction are of small size and are sent much less frequently than data packets, a member can manage much more foster connections.

During the fostering period, several changes could happen. First, the new child may find a more suitable position in the current tree branch and move down. Second, due to periodical tree improvement, an existing child may move or become the child of the new member. Third, if none of the fan-outs or the foster child finds a better position, the tree stays unchanged. For the third case, then the foster child or a fan-out will be forced to move away depending on their RTTs to the parent. During the fostering period, a foster child can receive the application data from its old parent or from a randomly selected member if it is in the initial joining period. Results in Section 4.2 show how foster connections improve the tree quality.

Comparative Performance Evaluation of AOM

This section compares AOM with HMTP [3], a typical direct-tree protocol. First, the tree quality is evaluated in randomly generated 1000-node transient-stub network topologies. The effects of foster limit on the tree quality is also studied. In this part, network conditions are static in that: 1) Link delays are pre-assigned and do not change during the simulations; 2)

The members do not leave the multicast group during the simulation period. The second part focuses on the adaptability of the two schemes. Faults like delay and loss rate surges are added to randomly selected links to observe how the schemes respond to such events.

Performance Metrics

The following metrics are used to evaluate the tree quality:

- *Group EED ratio*: the average of group members' EED ratios. A member's EED ratio is the ratio between its overlay EED and its EED in the Shortest Path Source Tree (SPST) in IP multicast. This metric measures the increase in EED in overlay multicast.
- *Average link stress*: assume $LS(i)$ is the number of duplicate packets on a link i. Average link stress is defined as: $\sum_{i,LS(i)>=1} LS(i) \;/\; \sum_{i,LS(i)>=1} 1$. It reflects the load added to a link by an overlay multicast protocol.
- *PDF for link stress*: shows the distribution of the stress over the physical links as well as the most stressed link.
- *CDF for path length*: path length is defined as the number of physical links (hops) in a member's ROOT path. A longer path is not desirable because it often implies a larger EED and certainly more processing overhead.
- *Tree cost*: the total number of physical links used by the tree. *Tree cost ratio* is defined as the ratio of overlay multicast tree cost to the corresponding SPST tree cost.
- *Control overhead*: the total control traffic used to build and maintain the tree.

Comparison of Tree Quality

Simulation Background

The simulation is implemented using Network Simulator-2 [22]. The 1000-node transient-stub network topology is randomly generated by GT-ITM [23]. For simplicity, links are assigned symmetric random delays. Since both AOM and HMTP use RTT to estimate the delay between two members, symmetric delay does not favor any of them. The simulation results are the average of 10 runs with a C.I. of 95% except for those that describe the transient behavior. Table 2 summarizes the values of the simulation parameters.

Table 2: Simulation Parameters

Parameters	AOM	HMTP
Improvement period	60 seconds	30 seconds
Foster limit	0, 50	0
Join time	uniform(0,1500) seconds	
Simulation time	2500 seconds	
Fanout limit	10	
Group density	5%, 8%, 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80%, 90%	

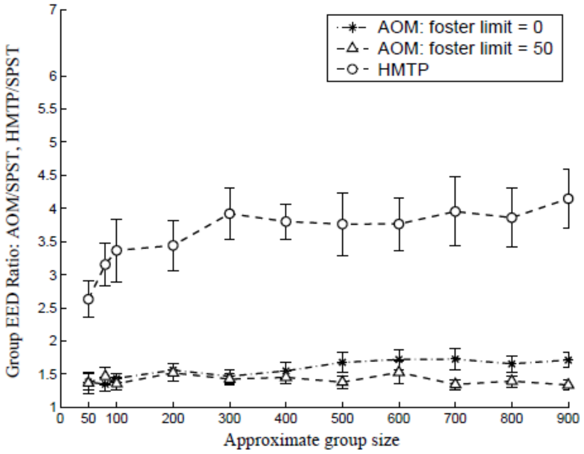

Figure 5: Tree quality: group EED ratio

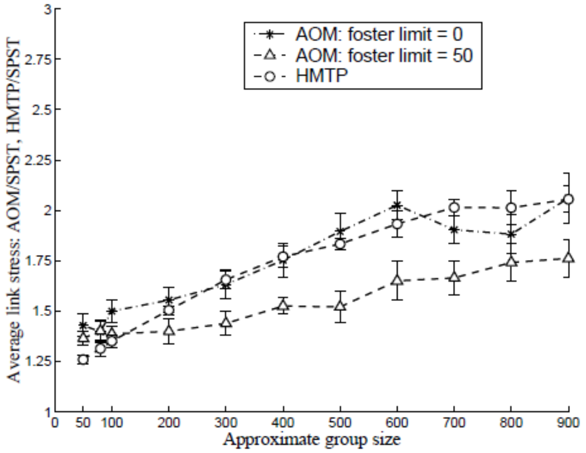

Figure 6: Tree quality: average link stress

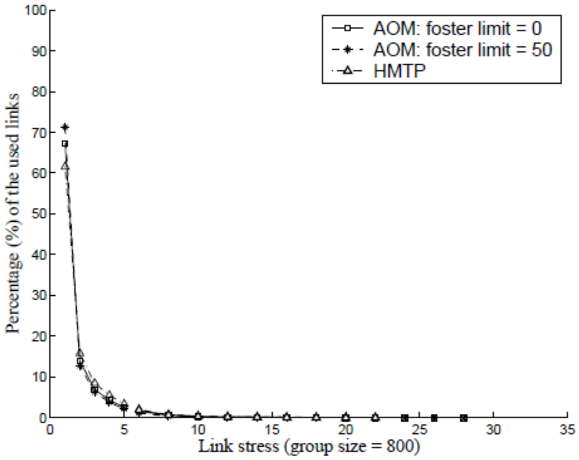

Figure 7: Tree quality: *pdf* of link stress

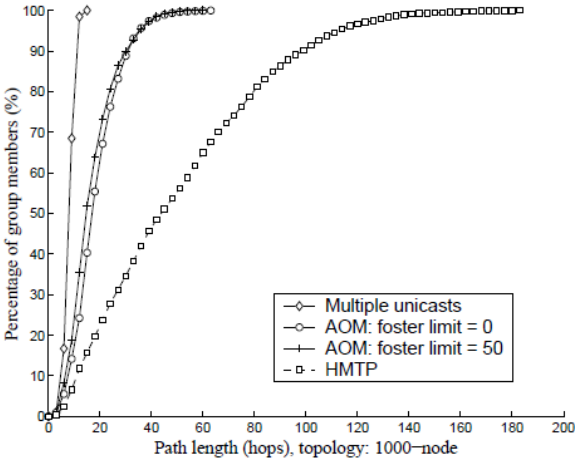

Figure 8: Tree quality: CDF of path length

Group EED Ratio

Figure 5 shows that for the group size ranging from 50 to 900, a member in AOM has a much smaller EED than in HMTP on average. This is because that the tree algorithm in AOM considers not only the RTTs but also the EEDs to the ROOT. A member in AOM has limited tolerance to increase its EED. This avoids the long paths that could occur in HMTP, as will be shown in Section 4.2.5. With the exception of when the group size is 50, the group EED ratio of AOM is at least 60% less than that of HMTP. Another observation is that from the group size of 50 to 900, while in HMTP the group EED ratio increases 60%, in AOM, it remains low and stable. This means that the AOM tree matches the underlying network topology better and the AOM scheme is more scalable.

It is also shown in Figure 5 that fostering a few members for the purpose of tree construction improves the EED ratio. At a large group size, fostering 50 children in AOM decreases the EED ratio by 15% over no fostering.

Average Link Stress

Unlike in an IP multicast tree where every link has a stress of 1, some links in an overlay multicast tree have duplicate packets. The link stress is affected by the group size and whether the tree matches to the underlying network topology. Figure 6 shows the average link stress of HMTP and AOM. In both schemes, average link stress increases with group size. This is because the more the members, the more likely that some links are repeatedly used. When the group size is small, HMTP builds lower-stress trees than AOM. However, in large groups, AOM with foster children of 50 outperforms both HMTP and AOM with no foster child by up to 17.36%.

PDF for Link Stresses

Figure 7 shows the *pdf* of the link stresses collected from 10 simulation runs. The group density of the simulations is 85%. The largest link stress in the figure is the largest link stress to appear in the 10 simulation runs. It can be seen that almost 99% of the links in both schemes have a stress number of less than 7; however, there are a few heavily stressed links in each scheme. In the group of about 800 members, the worst link stress to appear in 10 runs is 28 in AOM for both foster limits. HMTP has a smaller worst link stress of 22.

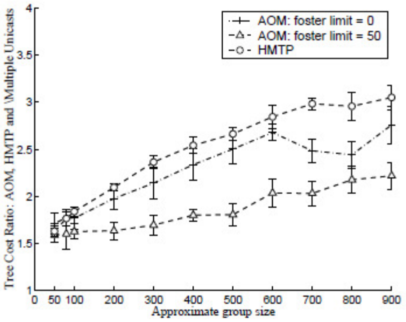

Figure 9: Tree quality: cost ratio

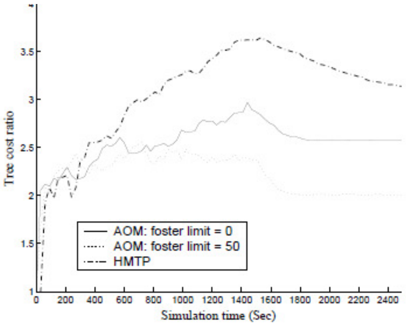

Figure 10: Convergence of tree cost ratio

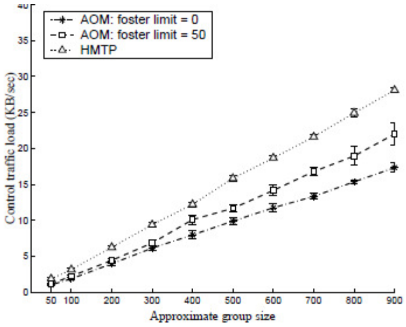

Figure 11: Overhead traffic load added by the schemes

CDF for Path Length

An ideal overlay multicast tree should provide short ROOT paths and low link stress. However, it is difficult to achieve both objectives at the same time. One extreme is the multiple unicasts tree in which the paths are short but the link stresses are high. Another extreme is that the path is extremely long, but the link stress is low. Often, longer paths result in larger EEDs.

Figure 8 shows the CDF of the ROOT path length in AOM, HMTP and Multiple Unicasts. Path length is the number of physical links involved in a member's ROOT path. In the figure, the path lengths of all the members in a group are collected from 10 simulation runs with group density of 85%. As can be seen, multiple unicasts tree has the shortest path length which is also the lower bound of the overlay multicast tree. The path length of AOM is moderate because the longest path length is 60 hops when the foster limit is 0. At least 30% of the HMTP members have a ROOT path longer than the longest path in AOM.

Tree Cost Ratio and Convergence

Tree cost reflects the total resource consumed by the overlay multicast group, such as bandwidth and processing power. Figure 9 compares the average tree cost of AOM and HMTP. For each group size, the result is the average of 10 trees and is normalized by the cost of the corresponding SPST. As can be seen that in large groups, fostering children in building multicast trees in AOM saves 20%-32% more network resources than HMTP.

Figure 10 shows the change of the tree cost in a typical run of each scheme. About 800 members join the group in the first 1500 seconds. At the initial phase, the tree cost increases rapidly. After all the members join the group, the tree cost begins to decline as the improvement algorithm continues to work. It can be seen that AOM with the foster limit of 50 has the smallest tree cost and fastest convergence.

Control Overhead

Both AOM and HMTP need control packets to build and maintain the tree. In HMTP, control traffic is used to refresh information between parents and children, measure the RTTs and query for the information used by the tree improvement algorithm; control traffic in AOM is used to exchange information between a member and its ancestors. In the simulations, 40-Byte control packets are used. Figure 11 shows that the control traffic load added by the two schemes increases with the group size. At group size of 900, the control traffic load reaches 30KB/sec in HMTP but is 20% lower in AOM with a foster limit of 50 and 40% lower without fostering. Such control traffic load is not large in the sense that it is distributed across the entire network rather than on a single link. AOM with the foster limit of 50 incurs more control traffic than with no fostering.

Conclusions

End-hosts over the Internet are heterogeneous; therefore, the maximum number of unicast connections that can be set up to forward the application data (called fan-outs) depends on factors such as bandwidth, traffic load and host processing power, and may vary from time to time. Our study shows that appropriate fan-outs limit and the foster limit (temporary connects used for multicast tree construction rather than application data distribution) can improve the overall tree quality and the application performance.

Multicast applications have different performance requirements. For example, media streaming applications are sensitive to delay, loss and available bandwidth; content distribution, such as server replication and large software distribution, can be loss intolerant; a delay jitter requirement must be satisfied in voice applications. Therefore, future work for AOM should consider more end-to-end performance metrics such as available bandwidth and the jitter requirement.

References

[1] Y. Chawathe and S. McCanne and E. Brewer. An architecture for Internet content distribution as an infrastructure service. 2000. Unpublished, available at http://www.cs.berkeley.edu/yatin/papers.

[2] Y. Chu and S. G. Rao and H. Zhang. A case for end system multicast. *IEEE journal on selected areas in communication (JSAC), special issue on networking support for multicast*, 2002.

[3] B. Zhang and S. Jamin and L. Zhang. Host multicast: a framework for delivering multicast to end users. *Proceeding of IEEE INFOCOM*, 2002.

[4] P. Francis. Yoid: Extending the Multicast Internet Architecture. 2000. White paper http://www.aciri.org/yoid/.

[5] M. Yajnik and J. Kurose and D. Towsley. Packet loss correlation in the mbone multicast network. *Proceeding of IEEE Global Conference on Communications*, 1996.

[6] R. Cáceres and A. Adams and T. Bu and N. G. Duffield and T. Friedman nad J. Horowitz and F. Lo Presti and S. B. Moon and V. Paxson and D. Towsley. The Use of End-to-End Multicast Measurements for Characterizing Internal Network Behavior. *IEEE Communications*, 38(5), 2000.

[7] C. Labovitz and A. Ahuja and A. Bose and F. Jahanian. Delayed internet routing convergence. *Proceeding Of ACM SIGCOMM*, pages 175-187, 2000.

[8] NLANR active measurement project. http://watt.nlanr.net/.

[9] Birman, Kenneth P. and Hayden, Mark and Ozkasap, Oznur and Xiao, Zhen and Budiu, Mihai and Minsky, Yaron. Bimodal multicast. *ACM Trans. Comput. Syst.*, 17(2):41--88, 1999.

[10] S. Banerjee and B. Bhattacharjee and C. Kommareddy. Scalable application layer multicast. *Proceeding Of ACM SIGCOMM*, 2002.

[11] S. Wu and S. Banerjee. Improving the Performance of Overlay Multicast with Dynamic Adaptation. *Proceedings of IEEE Consumer Communications and Networking Conference (CCNC), Las Vegas, NV*, 2004.

[12] S. Wu and S. Banerjee and Xiaobing Hou and Richard A. Thompson. Active Delay and Loss Adaptation in Overlay Multicast. *Proceeding of IEEE International Conference on Communications*, 2004.

[13] D. Pendarakis and S. Shi and D. Verma and M. Waldvogel. ALMI: an application level multicast infrastructure. *Proceedings of the 3rd USENIX Symposium on internet technologies and systems (USITS 2001)*, 2001.

[14] Zhi Li and Prasant Mohapatra. HostCast: A New Overlay Multicasting Protocol. *Proceeding of IEEE International Conference on Communications*, 2003.

[15] X. Zhang and X. Li and Z. Wang and B. Yan. A Delivery Tree Building Approach for Application Layer Multicast. *International Symposium on Computer Science and Computational Technology*, pages 630--634, 2008.

[16] A. Mourad and M. Ahmed. A Scalable Approach for Application Layer Multicast in P2P Networks. *Pervasive Computing and Communications, IEEE International Conference on*, 0:498-503, 2008.

[17] R. Moreno-Vozmediano. Application Layer Multicast for Efficient Grid File Transfer. *International Journal of Computer science and Applications*, 6(1):70-84, 2009.

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

[19] Suman Banerjee and Seungjoon Lee and Bobby Bhattacharjee and Aravind Srinivasan. Resilient Multicast Using Overlays. *ACM SIGMETRICS 2003, the International Conference on Measurement and Modeling of Computer Systems*, San Diego, California, 2003.

[20] S. Floyd and M. Handley and J. Padhye and J. Widmer. Equation-based Congestion Control for Unicast Applications. *Proceeding Of ACM SIGCOMM*, pages 43-54, 2000.

[21] D. Andersen and H. Balakrishnan and F. Kaashoek and R. Morris. Resilient Overlay Networks. *Proceeding of the 18th ACM SOSP, Banff, Canada*, 2001.

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

[23] GT-ITM: Georgia Tech Internetwork Topology Models. http://www.cc.gatech.edu/fac/Ellen.Zegura/graphs.html.