Considering Priority in Overlay Multicast Protocols under Heterogeneous Environments

Michael Bishop and Sanjay Rao
Purdue University

Kunwadee Sripanidkulchai
National Electronics and Computer Technology Center, Thailand

***Abstract*— Hosts participating in overlay multicast applications have a wide range of heterogeneity in bandwidth and participation characteristics. In this paper, we highlight and show the need to systematically consider prioritization as a key criterion in the design of protocols for overlay multicast. We identify trade-offs in the design of prioritization heuristics in two important contexts. The first part of the paper considers prioritization strategies in the context of heterogeneity in node outgoing bandwidth and node stay time durations, and a lack of correlation between the two dimensions. The second part of the paper considers bandwidth allocation and prioritization policies with multi-tree data delivery in environments with heterogeneity in outgoing bandwidth and a certain degree of altruistic behavior. We conduct a systematic study of the trade-offs using both real trace data, and sensitivity studies using synthetic workloads. To the best of our knowledge, this is the first work to identify and study these trade-offs, and to demonstrate the potential benefits of the resulting prioritization heuristics.**

I. Introduction

Overlay multicast has emerged as an alternative to IP Multicast in recent years. In this architecture end systems participating in a multicast group self-organize into efficient overlays for delivering data, and no support from network infrastructure is required. There has been significant effort in recent years in validating the architecture [1], [2], [3], [4], [5], designing protocols [6], [7], [8], [1], [2], [3], [9], [10], [11], [12], [13], [14], and deploying real systems [15], [16], [17].

Much effort in overlay multicast has been towards tackling issues like scalable group management and robust delivery through redundant structures. In this paper, we focus on a key orthogonal issue for protocols: heterogeneity in characteristics of end systems and the implications such heterogeneity may have on the design of protocols for overlay multicast. To motivate the issues, consider Table I, which summarizes the characteristics of hosts in real broadcasts using an operational overlay broadcasting system [15]. It may be observed that there is heterogeneity in the outgoing bandwidth of nodes (due to the presence of nodes such as DSL, cable modem and Ethernet), as well as heterogeneity in the duration for which nodes stay. For example, in the *Slashdot* broadcast, about 27% of hosts are behind high-speed access links, while about 73% behind low-speed ones. Further, the median stay time of nodes is 3 minutes, while the mean stay time is 14 minutes. In the Sigcomm broadcast, 48% of hosts are behind low-speed links and the median and mean session duration are eight minutes and 35 minutes.

In this paper, we argue that such heterogeneity in node characteristics makes it important to carefully consider prioritization in protocol design. In particular, overlay trees must be constructed in a manner as to enable nodes that are more critical to the system (for example, nodes that contribute more bandwidth or are more likely to yield stable performance) to be placed at higher locations in the overlay tree. We believe such carefully constructed prioritization policies can serve two goals. First, they can improve the performance of the entire set of hosts. Second, they can ensure nodes that contribute more to the system can receive better performance, which in turn could provide an incentive for them to contribute more.

These observations motivate us to conduct a systematic study of various prioritization strategies for constructing overlays in the presence of heterogeneity. These strategies assume the same base protocol for overlay construction, however they differ with regard to how nodes are prioritized relative to each other. Nodes with higher priority occupy higher positions in the tree.

The first part of this paper focuses on data delivery using single trees. Given that the performance of a node depends on its depth in the tree and the stability of its ancestors, we consider prioritization heuristics based on the outgoing bandwidth and stay time duration of nodes. Our study is set in the context of a large and interesting class of fixed duration broadcasts, where evidence from several measurement studies have indicated nodes with a higher age tend to stay longer in the group [18], [15]. Further, given indications from real data that outgoing bandwidth and stay times are not correlated (for example, the correlation coefficient is -0.01 for the Slashdot trace), we systematically evaluate a family of heuristics that choose different operating points to trade off node outgoing bandwidth and node stay time. Our results indicate clear-cut advantages for degree-based prioritization and show potential benefits for age-based prioritization, but indicate that combining degree- and age-based prioritization does not significantly improve performance over degree-based prioritization alone.

The second part of the paper considers data delivery using multiple trees [7], [11]. Here, the multimedia stream is encoded into many sub-streams, and each sub-stream is distributed along a particular overlay tree. The quality experienced by a receiver depends on the number of sub-streams that it receives. The performance with multi-tree data delivery depends both on how the outgoing bandwidth of a node is allocated across various trees, as well as the prioritization

TABLE I

CONSTITUTION OF HOSTS IN DIFFERENT BROADCASTS, CONDUCTED USING AN OPERATIONALLY DEPLOYED BROADCAST SYSTEM BASED ON OVERLAY MULTICAST.

Event	Low Speed 100Kbps	High Speed 10Mbps	Mean Session Duration	Median Session Duration	Total Size	Peak Group Size
Sigcomm2002	48%	52%	2124	465	110	44
Slashdot	73%	27%	826	197	1023	136
GrandChallenge	82%	18%	969	358	690	176
Rally	75%	25%	1200	465	1720	435
SOSP2003	48%	52%	1032	144	260	54

heuristics within a tree. We evaluate combinations of policies for allocating outgoing bandwidth and heuristics for prioritization. Our results indicate that employing prioritization heuristics optimized for single tree scenarios is insufficient, and it is important for a prioritization heuristic to consider the overall contribution of a node across all trees rather than its contribution in the particular tree alone.

In this paper, we make the following contributions:

• We show by performance evaluation through real traces that carefully designed prioritization policies that leverage heterogeneity in node characteristics have the potential to significantly improve the performance of an overlay multicast protocol, both in terms of improving the overall performance of nodes, as well as performance of nodes that contribute more to the system.

• We have identified and studied key trade-offs in the design of prioritization policies in two different contexts. To our knowledge, this is the first formulation of these trade-offs, and we believe that both the formulation and the results of our study are novel contributions. Finally, we believe ours is the first study to consider the performance of multi-tree protocols in environments with heterogeneous node degree constraints.

The rest of the paper is organized as follows. Section II describes our evaluation framework and methodology. Section III presents a study of prioritization heuristics in the single tree context. Section IV presents the background and study of heuristics in the multi-tree context. Section V presents conclusions.

II. EVALUATION FRAMEWORK

Our evaluation is motivated by video broadcasting applications. Such applications involve data delivery from a single source to a set of receivers. Further, they are non-interactive, and do not place a tight constraint on the end-to-end latency. We assume a constant bit rate (CBR) source stream, and assume only nodes interested in the content at any point in time are members of the distribution tree and contribute bandwidth to the system. We assume that all receivers can receive the full data rate.

A. Factors Impacting Performance

In an overlay multicast application, the performance seen by a node depends on two factors: (i) the frequency of disruptions due to the failure of an ancestor or due to congestion on an upstream link, and (ii) the time it takes a protocol to recover from the disruptions. The frequency of disruptions that a node experiences in turn depends on: (i) the depth of the node (i.e. the number of ancestors the node has) and (ii) the quality of the ancestor nodes and links.

A key factor affecting the node depth is the outgoing bandwidth limit of each host, which determines its *degree* in the overlay multicast tree, i.e., the maximum number of children to which it can forward the stream. The average degree of nodes in the tree and the distribution of their degree can impact the depth of constructed trees.

There are two dimensions that affect ancestor quality. First, the time for which a node stays is important as a large number of unstable or short-lived nodes can worsen overall performance. Second, network bandwidth limitation on upstream overlay links can impact application performance. In this paper, we focus on node stay time durations and ignore losses due to network bandwidth limitation. Further, in a multi-tree data delivery framework, the quality of all ancestors in all trees determines a node's performance.

B. Methodology and Models

Our study is conducted using simulations based on a collection of real-world traces obtained from the operational deployment of an overlay broadcasting system using End System Multicast [15]. The details of the traces are summarized in Table I. The traces includes the join/leave patterns of different nodes, as well as estimates of the outgoing bandwidth of nodes. The primary trace we use is the *Slashdot* trace which is from a broadcast to the Slashdot community. It includes over a thousand participants with a peak group size of over a hundred nodes. We use a two-hour segment at the start of the broadcast, which represents the maximum activity. *Sigcomm2002* and *SOSP2003* are broadcasts of conferences, and thus have a much larger fraction of hosts behind high-bandwidth university machines. *GrandChallenge* is a broadcast of the DARPA Autonomous Vehicle competition, and *Rally* refers to a broadcast of an election campaign. In addition to experiments with real traces, we have conducted simulation experiments with synthetic workloads to study the sensitivity of our results to various environment parameters (e.g. constitution of hosts), and larger group sizes. Details regarding the models used is provided in Section IV-E.

In this paper, we model access-bandwidth limitation of hosts, but do not model network-bandwidth limitation. Our simulation experiments presented in this paper assume that each host is capable of forwarding as much bandwidth as it receives. This corresponds to a scenario where the source rate is low enough that all hosts are capable of forwarding the source rate. We believe, however, that all our discussions and results in this paper hold true in a more general scenario where hosts (behind asymmetric bandwidth connections) forward less data than they receive.

Our simulations do not consider the underlying topology model and network delays. Instead, we consider a uniform-delay network model throughout this paper. We believe this assumption is reasonable given that our focus is on bandwidth-sensitive and non-interactive applications; our performance metrics relate to loss seen by the application. Having said this, one aspect that our study does not consider is potential correlation between delay and throughput. For example, it may be desirable to avoid high delay transoceanic links even in a non-interactive application, as these links may be more prone to network-bandwidth limitations.

C. Protocol and Prioritization

We conduct our simulations using a simplified centralized protocol based on the one used in [15]. The join, leave, and parent selection functions in our simplified protocol are also common across many of the existing overlay multicast protocols. When a node joins, it looks for a parent with the lowest possible depth to which it could attach itself. If a node leaves, then each of its disconnected children repeats the same process.

We incorporate priority into the protocol as follows. When a node A is looking for a parent (either because A has joined or the parent of A has left), then from among all parents that A is *eligible* to choose, A chooses the one of minimum possible depth. A is *eligible* to choose B as a parent if B is not saturated or if B has a child C that has a lower priority than A. In the latter case, A preempts, or displaces, the child C and takes its spot, leaving C to find a new parent.

Comparing the priorities of A and C is done in a manner specific to the particular prioritization algorithm used. For example, in a purely age-based scheme, a node with a higher age has a higher priority, while in a purely degree-based scheme, a node with a higher degree has a higher priority.

One potential concern with a preemption scheme such as the one we consider is the cost associated with such preemption. However, we believe in practice preemption can be implemented in a graceful manner so as to have negligible impact. Thus, when a node A preempts a node C with parent B, B continues forwarding data to C until it successfully finds a new parent. In addition, as discussed in Section II-D, we will consider a metric that captures the interval between ancestor changes (where a change may include a preemption).

In this paper, we focus on one specific approach to prioritization, where a node with higher priority occupies a higher position in the tree. Further, we focus on a particular mechanism – preemption – to achieve such prioritization. In general, one can conceive of other mechanisms for achieving prioritization, such as providing greater degree of redundancy to nodes that contribute more to the system.

D. Metrics

Our evalutions of a scheme summarize both the performance of the entire set of hosts as well as the performance of nodes in a higher degree class. This is because we wish to evaluate both the performance of the entire set of hosts and the performance of hosts that contribute more to the system. We now discuss our metrics to capture the performance of an individual host.

When a host leaves, it causes all of its descendants to become disconnected from the overlay and stop receiving data until they find new parents and are reconnected to the tree. To capture stability of the overlay we look at two metrics:

•**Interval between ancestor change:** This metric captures the typical performance of each host. The longer the interval, the better the performance. If a host sees only one ancestor change during its session, the time between ancestor change is computed as its session duration. If a host sees no ancestor changes at all, the time between ancestor change is infinite. In our experiments, we consider ancestor changes that may be caused by either the ancestor leaving the group, or an ancestor being preempted. Given that preemption can be implemented in a graceful manner, our results for preemption-based schemes are a conservative estimate of their performance.
•**Stream loss rate:** We compute the loss rate that hosts see by assigning a penalty to each disruption. A preemption is considered to have minimal penalty as it can be a graceful hand-off and is assigned a penalty of a couple of round-trip times (typically around 100 milliseconds). When a parent departs, we assume that a child is disconnected for T seconds before attempting to reconnect to a new parent. Our simulations assume T is 5 seconds. This is because in the case of an abrupt departure of a parent, a child cannot be overly aggressive in shifting to a new parent – when a node A switches to a node B, it takes $1 - 2$ seconds to get good performance assuming a congestion-control scheme involving a slow start phase (e.g. TFRC or TCP) is used on the data path.
Discussion of metrics: We discuss a few key aspects of our metrics and the care that must taken in interpreting them.
• While comparing the performance of a scheme with prioritization versus a scheme without prioritization, we will consider both the time between ancestor changes and the loss rates obtained. The time between ancestor changes is a conservative estimate of performance of a prioritization scheme as it does not distinguish between whether the ancestor change is due to preemption or ancestor departure. The loss rate model, however, assumes a low cost for preemptions and a higher cost for ancestor departures. We consider a prioritization scheme to be clearly better than a scheme without prioritization if it results in an increased time between ancestor changes. For example, in Section III, we consider degree-based prioritization a clear win over no prioritization as it results in a higher time between ancestor changes, while we consider the benefits of age-based prioritization not to be as clear cut as it results in a minor improvement in time between ancestor changes, but significant improvement in loss according to our model.
• Our magnitude of loss rates is sensitive to the detection time parameter. While we believe our choice of detection time to be reasonable, we have conducted sensitivity analysis to the

detection time (for example, Section IV). Further, while our model of ancestor leaves directly impacts the magnitudes of resulting loss rates, we believe the overall loss rate trends are still relevant. We note that a few factors can result in smaller loss rates in practice. First, some node departures may be graceful, where an ancestor informs its descendants about its departure and continues to forward packets to its children for a while. Second, we do not model receiver-side buffering and packet recovery through retransmission, which may help absorb some of the performance degradation caused by node departures.

III. Single Tree Protocols

In this section, we consider strategies for constructing overlay trees in the presence of heterogeneity in outgoing bandwidth of nodes and the duration that nodes stay. While it is feasible to construct overlay trees in a manner that does not consider such heterogeneity, we are motivated to investigate whether there are benefits to prioritizing nodes that stay longer or have a higher outgoing bandwidth by placing them at higher locations in the tree. Prioritizing nodes with a higher outgoing bandwidth has the potential to reduce overall tree depth, and consequently the disruption rate and overall loss that nodes see. Prioritizing nodes with a higher stay time decreases the probability that an ancestor of a node leaves before the node, and thus potentially decreases disruption and loss rates.

We begin by observing that if all nodes have a degree greater than zero, the **optimal strategy** simply organizes nodes in the decreasing order of their *remaining stay times* (RST). As such, all the ancestors of a node will leave only after it does. Therefore, there will be no disruptions caused by ancestor death. Such a policy is optimal as it provides no loss to any node in the system. Note that node degree does not factor into this decision. There are two key assumptions in such an optimal scheme: (i) RST is known in advance, and (ii) preemption based on RST has no cost. While it may be possible to ensure that preemption has minimal cost, the optimal algorithm is not realizable because RST may not be known in advance.

While the optimal strategy is not realizable because the RST may not be known in advance, we believe that in a large class of interesting fixed-duration broadcasts, the age of a node could be used as a good predictor of the RST – that is, a node that has a higher age is likely to stay longer. Our belief is based on measurement studies on group dynamics characteristics observed in overlay multicast deployments [15], Mbone measurements [19] and content delivery networks [18]. For example, to examine the feasibility of such prediction, we consider the predictor accuracy of age for the *Slashdot* trace. We consider nodes that are present in the system at various snapshots and count the number of pairs of nodes in which the node with higher age also has a higher remaining stay time. The results show prediction accuracies of around $70-75\%$ at various snapshots in the trace. In the rest of the study we will focus on this class of broadcasts where age could be a good predictor of RST, and do not consider for example periodic broadcasts where nodes are likely to stay in the system for similar periods of time.

The questions that motivate our work are:

• Does a scheme that considers prioritization significantly improve performance over a scheme that does not? We consider the impact both on all nodes in the system and on nodes of the higher degree class which contribute more.
• What are the relative benefits to considering node degree alone, node age alone, and combinations of the two?
• Which combinations of node degree and node age perform the best? What factors does this depend on?

We study these questions using trace-based simulations. Traces provide us with a realistic idea of the importance of the issues, and provide us realistic workloads which enable us to evaluate the effectiveness of age-based prioritization schemes.

A. Prioritization Algorithms Considered

We now describe the schemes we evaluated. The schemes we consider include not having any prioritization at all, prioritization based on degree alone, and age of nodes alone. In addition, to explore the trade-offs in considering both degree of contribution and age, we look at a spectrum of algorithms that combine the two with degree always being prioritized over age at one end of the spectrum and age prioritized over degree at the other end.

We now describe the specific schemes we considered:

• *No-Preemption:* This scheme does not prioritize nodes by their degree or age. It simply creates as packed a tree as possible by having a node that is looking for a parent pick a free slot at the lowest possible depth. Packing the tree in this fashion does help minimize the number of affected descendants when an ancestor higher up at the top of the tree leaves.
• *Preempt-Degree:* The key difference between this scheme and the previous scheme is that a node with higher degree may displace or preempt nodes with lower degree to achieve even better depth.
• *Preempt-Age:* In this scheme, nodes that are older may preempt nodes that are younger.
• *Degree vs. Age hybrids:* To better understand the trade-off between degree and age, we look at the following family of algorithms. We compute a DegreeRatio and an AgeRatio for A vs. B, as $Degree(A)/Degree(B)$, and $Age(A)/Age(B)$. If both ratios are above 1, A has higher priority. If only one of DegreeRatio or AgeRatio is above 1 and the other is below 1, A has priority only if $DegreeRatio > AgeRatio^{-p}$, where p is a parameter. With a small p, for example 0.01, a node with a higher degree is almost always prioritized (though age may be considered if the degree of nodes are equal). With a large p, however, a node with a higher age is almost always prioritized (though degree may be considered if nodes have similar ages). For example, for $p = 0.01$, making up a DegreeRatio of 2 requires an AgeRatio of 2^{100}, while for

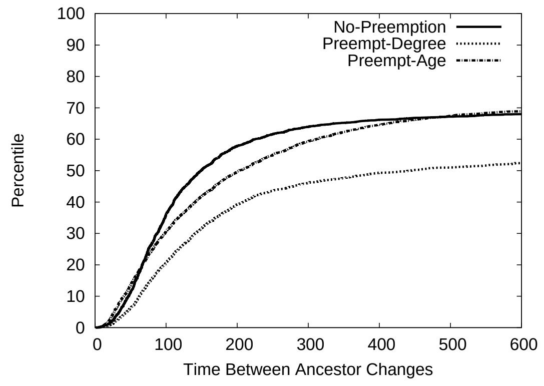

Fig. 1. Cumulative distribution of average time between ancestor changes of all hosts for the *Slashdot* trace. The top-most curve is *No-Preemption*. The middle curve is *Preempt-Age*. The lowest curve is *Preempt-Degree*.

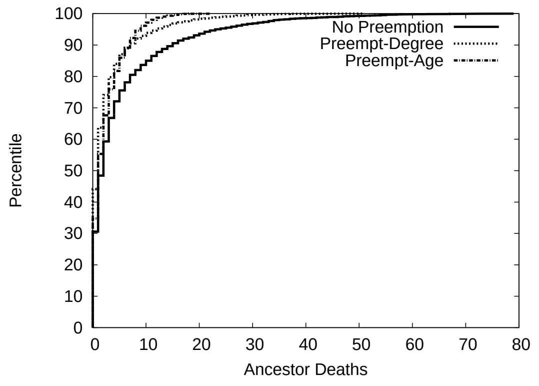

Fig. 3. Cumulative distribution of number of ancestor deaths for all hosts in the *Slashdot* trace. The top-most curve is *Preempt-Age*. The middle curve is *Preempt-Degree*. The lowest curve is *No-Preemption*.

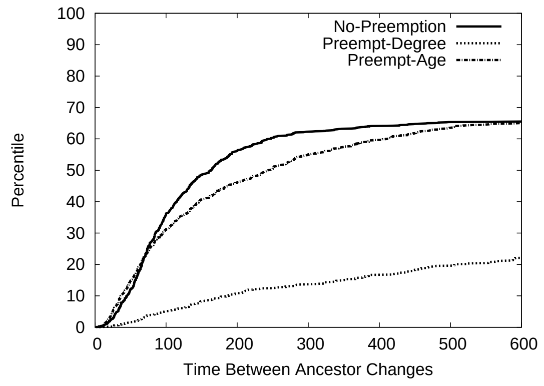

Fig. 2. Cumulative distribution of average time between ancestor changes of hosts with degree 2 for the *Slashdot* trace. The top-most curve is *No-Preemption*. The middle curve is *Preempt-Age*. The lowest curve is *Preempt-Degree*.

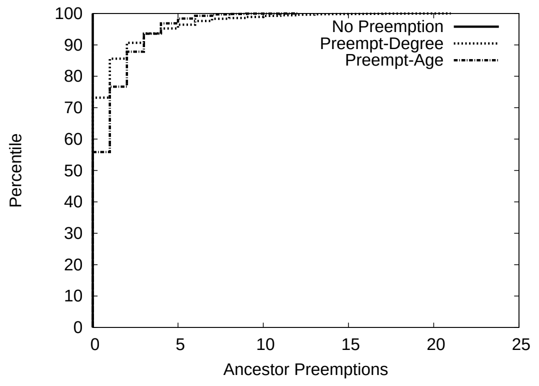

Fig. 4. Cumulative distribution of number of ancestor preemptions for all hosts in the *Slashdot* trace. The curve against the axis is *No-Preemption*. The top curve is *Preempt-Degree*. The bottom curve is *Preempt-Age*.

$p = 100$. making up a DegreeRatio of 2 only requires an AgeRatio of 1.0069.

B. Evaluation with real-world trace

We evaluate the performance of the above algorithms using the Slashdot trace from a real deployment of overlay multicast. We use the node capabilities and join and leave patterns from the trace. To assign a degree to a node, we use the following policy: we assume low-speed hosts have a degree of 1, and high-speed hosts have a degree of 2. We have explored policies that assign higher degrees to high-speed hosts – the overall trends of our results are the same. However, we find that the performance of all schemes, including the naive ones, improves.

Age Alone or Degree Alone: Fig. 1 depicts the cumulative distribution of the average time between ancestor changes for all hosts for the three basic algorithms: *No-Preemption*, *Preempt-Degree*, and *Preempt-Age*. The y-axis is the CDF and the x-axis is the average time between ancestor changes in seconds. A larger time between ancestor changes is more desirable as it indicates fewer disruptions; an infinite time between ancestor changes indicates no disruptions throughout the session. We truncate the x-axis at 600 seconds (10 minutes) as hosts with larger time between ancestor changes already see good performance. The *Preempt-Degree* algorithm has the best performance with 50% of the hosts seeing ancestor changes less frequently than 1 in 10 minutes. *No-Preemption* and *Preempt-Age* however involve fewer than 30% of hosts seeing ancestor changes less frequently than 1 in 10 minutes.

Now, we look at the time between ancestor changes for the degree-2 nodes as depicted in Fig. 2. First, note that the performance for degree-2 nodes for the *No-Preemption* and *Preempt-Age* algorithms are similar to the distribution for all nodes as shown in the previous figure. However, for the *Preempt-Degree* algorithm, the performance for degree-2 nodes is much better. Almost 80% of degree-2 nodes saw ancestor changes less frequent than 1 in 10 minutes. The *Preempt-Degree* scheme not only provides good overall performance, but also differentiates between classes of nodes and gives better performance to the better class.

There are two causes of ancestor changes: (i) ancestor death and (ii) preemption. The performance reduction induced by preemptions (Section II-D) can be kept small, while ancestor deaths incur a much larger penalty. Thus, the time between ancestor changes is a conservative estimate of the performance

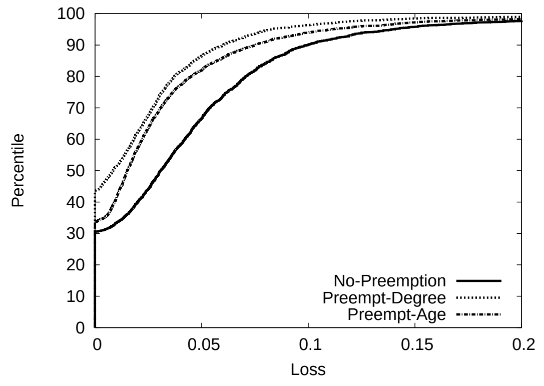

Fig. 5. Cumulative distribution of loss rate of all hosts in the *Slashdot* trace. The top-most curve is *Preempt-Degree*. The middle curve is *Preempt-Age*. The lowest curve is *No-Preemption*.

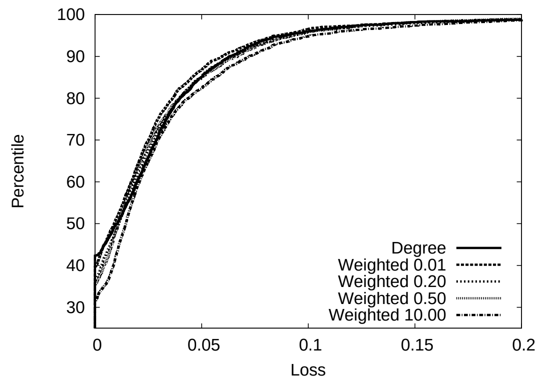

Fig. 6. Cumulative distribution of loss rate of all hosts in the *Slashdot* trace for various degree vs. age hybrid algorithms.

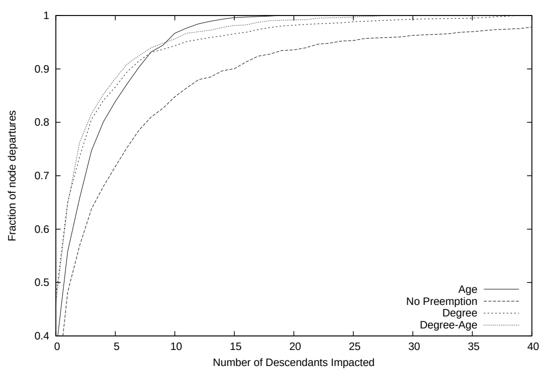

Fig. 7. The x-Axis is the number of descendants impacted by a node departure. A point (x,y) signifies the fraction of departures for which the number of affected descendants is less than x. *Degree-Age* refers to a hybrid scheme where degree is always prioritized over age (p=0).

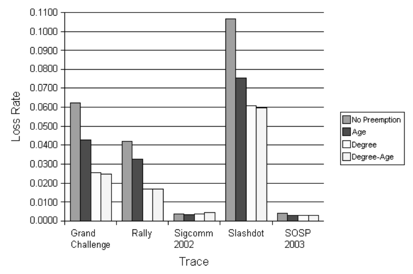

Fig. 8. 90th-percentile of loss rate for each policy across each of our test traces. *Degree-Age* refers to a hybrid scheme where degree is always prioritized over age (p=0).

of a scheme with preemption. Fig. 3 shows the number of ancestor deaths which nodes experience. Note that more ancestor deaths are experienced by all nodes when using the *No-Preemption* algorithm. The other two algorithms are similar, though most nodes experience fewer ancestor deaths under *Preempt-Degree* than under *Preempt-Age* . This improvement in the number of ancestor deaths comes at the cost of preemptions. In Fig. 4, we examine the number of preemptions experienced by each node. In the *No-Preemption* algorithm, all nodes experience zero ancestor preemptions for obvious reasons. Note that with *Preempt-Degree* , almost 75% of nodes never experience an ancestor preemption, while with *Preempt-Age* only 55% of nodes are spared.

Fig. 5 studies how the time between ancestor changes impacts user performance by considering the performance of the schemes with respect to the loss rate metric discussed in Section II-D. The x-axis is the average loss rate for all hosts, and the y-axis is the cumulative percentage. The *Preempt-Degree* algorithm performs better than *No-Preemption* – 90% of the hosts have a loss rate of 6% or less with *Preempt-Degree,* while with *No-Preemption* 90% of the hosts experience a loss rate of up to 12%. While this trend is expected because *Preempt-Degree* performs better even with respect to the time between ancestor changes, we note that the actual magnitudes are sensitive to our loss model.

Fig. 5 also shows that *Preempt-Age* performs much better than *No-Preemption*, though not as well as *Preempt-Degree*, with about 90% of the hosts having a loss rate of 8% or less. The *Preempt-Age* algorithm is effective at minimizing loss even though the time between ancestor changes is not significantly reduced. This is because most of the ancestor changes in the *Preempt-Age* algorithm are caused by preemptions which have a much smaller cost than ancestor death in our loss model.

Our results so far have indicated clear-cut advanges to degree-based prioritization. Degree-based prioritization clearly reduces the time between ancestor changes, and consequently loss, for the entire class of nodes, and leads to even more significant improvement for nodes of the higher degree class. The benefits of age-based prioritization, on the other hand, are less clear-cut. It does not significantly reduce the time between ancestor changes, because it results in an increased number of preemptions. However, it still results in significant improvements in loss rates, under the assumption that preemptions have a much smaller cost than ancestor death.

Hybrid Schemes: We now examine the performance of hybrid

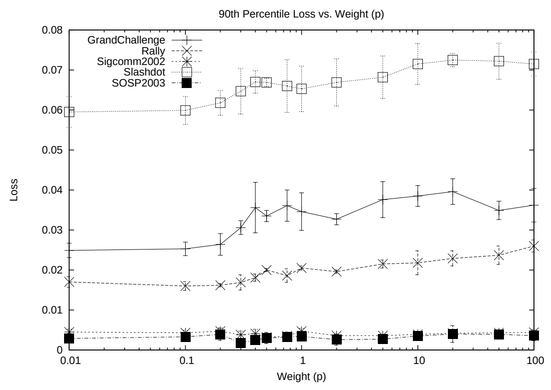

Fig. 9. 90th-percentile of loss rate for different degree vs. age hybrids

degree vs. age algorithms in Fig. 6. The y-axis is the average loss rate and the x-axis is the cumulative distribution across all hosts in the *Slashdot* trace. There are five curves – four of these curves represent different points in the hybrid space, corresponding to p values of 0.01, 0.2, 0.5, and 10. The fifth curve is *Preempt-Degree*, used for comparison. Recall that while choosing between a pair of nodes with the hybrid schemes, one with a higher degree and the other with the higher age, a small p value is biased toward degree whereas a large p value is biased toward age.

Fig. 6 indicates that all five algorithms have roughly similar performance and are almost indistinguishable. In short, there is almost no improvement in performance when age is combined with degree (for various combinations) as compared to using degree alone. It is interesting that while *Preempt-Age* can reduce loss over *No-Preemption*, combining degree and age does not reduce loss compared to *Preempt-Degree*. To explore this further, consider Fig. 7. The x-axis represents the number of descendants impacted by a departure, and a point (x, y) implies that a fraction y of the departures impact x or less descendants. The higher the curve, the more effective the scheme in minimizing the impact of departures on descendants. We see that *Preempt-Age* offers a significant improvement over *No-Preemption* — a given death is much less likely to affect a large number of other nodes. It is also clear that *Preempt-Degree* offers an even more significant advantage. By combining both approaches, *Preempt-Degree-Age* (hybrid scheme with a p of 0, where degree is always prioritized over age) offers only a marginal further advantage over *Preempt-Degree.* One possible hypothesis for this is that the main benefit of a hybrid degree vs. age scheme over degree alone is in minimizing the impact of departures of high-degree short-lived nodes, which are much smaller in number.

Overall our investigation with the Slashdot trace reveals that degree-based prioritization offers the best and most clear-cut benefits. While age-based prioritization may improve performance over not having any prioritization at all, combined age- and degree-based prioritization does not improve performance over using degree-based prioritization alone.

Sensitivity to trace: We have attempted to confirm these findings by using other traces. Fig. 8 plots the 90th-percentile of the loss rate taken across the set of hosts for *No-Preemption*, *Preempt-Degree*, and *Preempt-Age*. We also consider the performance of the hybrid degree vs. age scheme for a p value of 0 (degree is always prioritized over age). We observe several points. First, for the *SOSP2003*, and *Sigcomm2002* traces, even the naive *No-Preemption* scheme performs very well – and the benefits from any form of prioritization are marginal. This is because a large fraction of nodes in these broadcasts are behind well-connected university machines and the heterogeneity is therefore limited. Second, for all other traces – *GrandChallenge, Rally,* and *Slashdot* – degree-based prioritization clearly improves performance over *No-Preemption*. Third, age-based prioritization does help in each of these traces, however does not perform as well as degree-based prioritization. Finally, the hybrid scheme *Preempt-Degree-Age* (combination of degree and age with p=0, where degree is always prioritized over age) has a very similar performance to using degree-based prioritization.

Fig. 9 considers whether use of other hybrid degree vs. age schemes with different p values can improve performance over *Preempt-Degree*. The figure plots the 90th-percentile of the loss rate taken across the set of hosts for the degree vs. age hybrid schemes. The x-axis is the parameter p which represents a different point trading off degree and age. Each curve represents a different trace. For example, the top curve corresponds to the Slashdot trace. Overall, we see that the best performance is with small values of p, and choosing larger p values can sometimes result in degraded performance.

Overall, our results are consistent with our observations for the *Slashdot* trace, and indicate clear benefits for degree-based prioritization, and indicate that combining degree- and age-based prioritization does not lead to significant benefit. We have also conducted experiments with synthetic workloads, and the results show similar trends. We choose not to elaborate on these results. Our conclusions depend on the accuracy of age as a predictor for stay-time; in generating a synthetic trace, the accuracy of this prediction will depend directly on properties of the join-leave pattern specified. Therefore, we consider results from synthetic traces not as interesting for this section and confine our discussion to real traces.

C. Summary

We summarize our findings as follows:

• Our experiments have clearly revealed the advantages of prioritization based on degree. In three of our traces with significant heterogeneity, degree-based prioritization clearly reduces the time between ancestor changes for the entire class of nodes, and leads to even more significant improvement for nodes of the higher degree class. The benefits are limited in environments with a large fraction of nodes in the higher degree class (such as *SOSP2003* and *Sigcomm2002*), because the naive *No-Preemption* scheme itself performs well.

• The benefits of age-based prioritization are less clear-cut, even though age does have a strong correlation with Remaining Stay Time. It does not appear to significantly reduce the time between ancestor changes, perhaps because it results in an increased number of preemptions. It could, however, still result in significant improvements in loss rates under the assumption that preemptions have a much smaller cost than ancestor death.

• We considered the benefits of combining degree- and age-based prioritization choosing a range of hybrid degree vs. age algorithms. Our experiments do not reveal significant improvement in loss rate, and can potentially result in an decreased time between ancestor changes.

IV. Multiple-Tree Protocols

Rather than using a single tree, CoopNet [11] and Splitstream [7] have investigated delivering data using multiple trees. In these protocols, the source encodes the stream into many sub-streams and distributes each sub-stream along a particular overlay tree. The quality experienced by a receiver depends on the number of sub-streams that it receives.

There are two key advantages of the multi-tree solution. First, the overall resiliency of the system is improved, as a host is not completely disrupted by the failure of an ancestor on a given tree. Second, multi-trees have the attraction that they can be employed in more extreme environments where all hosts are behind limited bandwidth connections like DSL and cable modems. For example, consider a scenario where the source rate is 100 Kbps, and all participating hosts have an outgoing bandwidth of 100 Kbps. A single tree solution will result in a linear chain of participating hosts. In a multiple-tree solution, however, there is greater flexibility by having hosts allocate their bandwidth unevenly across trees. For example, Splitstream considers the interior disjoint policy where each host allocates its entire bandwidth in one tree, and does not allocate any bandwidth in the other trees. Thus, if two trees were used, 50% of the hosts would be interior nodes supporting two children (at 50 kbps each) in one tree, and leaf nodes in the other tree.

The multi-tree solutions so far have been constructed assuming hosts are homogeneous, and each host contributes as much as it receives. While this model is attractive from a fairness perspective, given the heterogeneity inherent in the Internet (Table I), such a model does not fully utilize the bandwidth resources available at hosts with a higher outgoing bandwidth capacity. In this paper, we explore a model where some hosts may contribute more than they receive. We call such hosts *high contributors.* Experience with real-world peer-to-peer systems [20], [16], [15], [17] indicates that it is not uncommon to see such behavior among peers. Further, in this paper, we require that high contributors be provided better performance, as they are contributing more to the system. In the rest of the paper, we refer to the *contribution parameter* of a host as the ratio of the total outgoing bandwidth that a host can supply across all trees to the total bandwidth a host receives across all trees. Note that in the case of a single tree solution, the contribution parameter is the degree of the host in the tree.

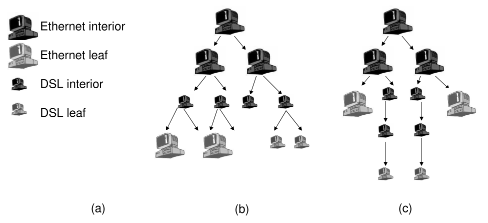

Fig. 10. Prioritization schemes with the interior disjoint policy for multi-trees. (a) Composition of hosts, and the outgoing bandwidth they allocate in each tree of the multi-tree framework; (b) Individual-tree-optimized prioritization; (c) Overall-host-optimized prioritization.

A. Multi-trees in heterogeneous environments

In the rest of this paper, we consider operation of multi-trees in such heterogeneous environments. We consider two key components: (i) Bandwidth Allocation Policy; and (ii) Prioritization Policy. While there has been some attention paid to the Bandwidth Allocation Policy [7], there has been relatively little attention paid to the prioritization policy, and to the combination of the two policies. We now describe the components:

• *Bandwidth Allocation Policy:* This dictates how a host allocates its outgoing bandwidth across multiple trees. We consider the *Interior Disjoint Policy* [7], where each host allocates all its outgoing bandwidth in one tree and does not allocate any outgoing bandwidth in the other trees. This provides resilience because when a host leaves the system, only the tree in which it is an interior node is impacted. We also consider the *Uniform Allocation Policy*, where each host allocates its outgoing bandwidth uniformly across all the trees. In this policy, when a host leaves the system, it may impact all trees. However, it may provide better performance to hosts with higher priority as discussed next.

• *Prioritization Policy:* This decides how nodes are prioritized in any given tree once a particular bandwidth allocation policy has been chosen. This choice of policy may impact the quality of individual trees in terms of depth and stability.

With a *Uniform Allocation*, the choice of prioritization policy is straightforward. Since the characteristics or importance levels of nodes are preserved in each tree, we simply consider a good prioritization policy designed in the single-tree setting. However, with the *Interior Disjoint* policy, the prioritization is more involved.

Fig. 10 illustrates the different prioritization schemes for the *Interior Disjoint Policy*. Consider two different classes of hosts – low contributors which we label as DSL and high contributors which we label as Ethernet. Consider a scenario involving T trees, with a source rate S, and a video rate S/T

being sent along each tree. Assume that DSL hosts contribute as much as they get, and donate a bandwidth S to the system, perhaps constrained by their outgoing bandwidth. Assume that Ethernet hosts donate a bandwidth $K * S$ to the system, where K is the *contribution parameter*. With an *Interior Disjoint Policy*, Ethernet hosts have a degree $K * T$ in the tree that they are an interior node, and 0 in each tree that they are a leaf node. Similarly, DSL hosts have a degree T or 0, depending on whether or not they are an interior node in the tree. Thus, any given tree has four classes of nodes – Ethernet interior, Ethernet leaf, DSL interior and DSL leaf as illustrated in Fig. 10(a).

We now consider two approaches for prioritizing nodes within each tree of the multi-tree dissemination framework:

- *Individual-tree-optimized:* In this approach, we simply prioritize nodes in each tree in the same manner as one would in a single tree, independently of other trees. Thus, we use only the degree allocated in a given tree to determine the node's priority for that tree. For example, in Fig. 10(b), the order of priority is (i) Ethernet interior, (ii) DSL interior, and (iii) Ethernet leaf and DSL leaf. While this approach may result in the best overall performance for nodes in individual trees, it may result in poorer performance for hosts that are overall more important to the system such as Ethernet hosts. In particular, the Ethernet hosts are rewarded only in the trees in which they are interior nodes, but potentially penalized in all other trees as leaf nodes.
- *Overall-host-optimized:* In this alternate approach to prioritization, we consider the overall contribution of the host across all trees while applying prioritization heuristics. For example, in Fig. 10(c), Ethernet hosts are prioritized over DSL hosts in all trees, irrespective of their status as an interior node or leaf node in that particular tree. A concern with this approach, however, is that the depth of each tree can increase as leaf nodes are occupying higher regions in the tree, thereby resulting in degradation in overall performance.

In the rest of this section, we describe details of specific schemes we considered, our evaluation results with the trace, and sensitivity.

B. Specific schemes considered

Based on our discussion above, we considered the following three policies:

- *Uniform-Tree:* Uniform bandwidth allocation policy, with tree-optimized prioritization. Further, given our discussions in Section III, in each tree we prioritize nodes with a higher degree. If there is a tie, we then consider the age of nodes.
- *ID-Tree:* Interior Disjoint bandwidth allocation policy with the prioritization optimized for the particular tree. Again, we prioritize nodes with higher degree and we consider age if there is a tie.
- *ID-Host:* Interior Disjoint bandwidth allocation policy with the prioritization optimized based on the overall contribution of the host across all trees.

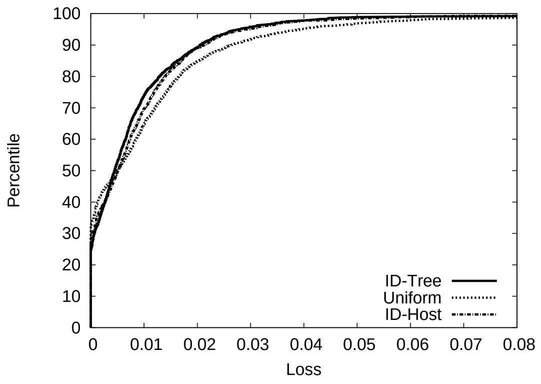

Fig. 11. Cumulative distribution of loss rate of hosts with various policies for the *Slashdot* trace, and assuming multi-tree delivery with four trees.

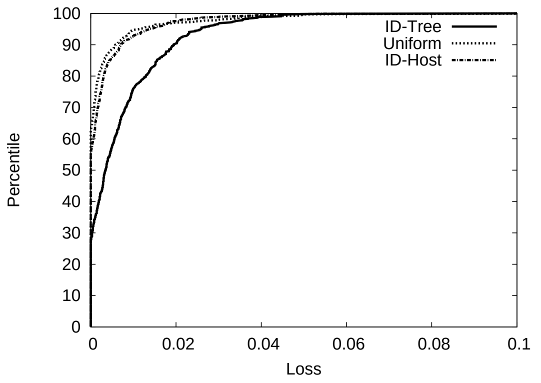

Fig. 12. Cumulative distribution of loss rate of hosts with the higher contribution parameter of 2 with various policies for *Slashdot* trace. Multi-tree delivery with four trees is assumed. The lowest curve is *ID-Tree*. The top two curves are *ID-Host*, and *Uniform*.

C. Evaluation with real-world trace

In this section, we present results evaluating the above policies with the *Slashdot* trace. As in Section III, we assume the ratio of the bandwidth supplied to the bandwidth consumed across all trees is 1 for low contributors, and is 2 for high contributors (i.e., contribution parameter is 2). In the rest of the section, we refer to these hosts with the higher contribution parameter as the high contributors. We evaluate the performance of various schemes with regard to the performance of all hosts, as well as that of high contributors alone.

In our setting, the ratio of bandwidth supplied to bandwidth consumed is 1 for the low contributors, and 2 for the high contributors. This corresponds to a scenario where the source rate is low enough that all hosts are capable of forwarding as much data as they receive – for example, a source rate of 100 Kbps if hosts behind low access bandwidth connections like DSL or cable-modem can sustain that rate. Further, some of the bandwidth that hosts are capable of supporting is unutilized. In general it is possible to envision a scenario with a higher source rate, with low contributors (typically behind asymmetric connections) supplying lower bandwidth than they receive. We believe our discussion and results remain valid.

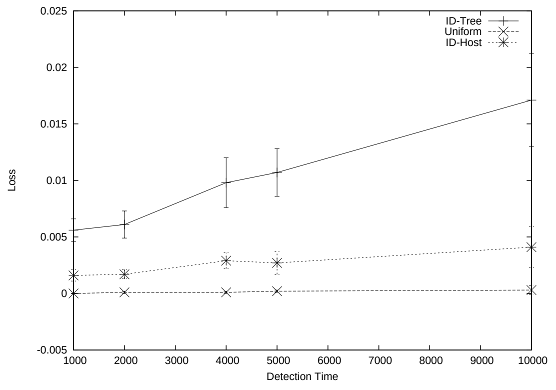

Fig. 13. Sensitivity of 90th-percentile loss rate of hosts with a high contribution parameters to ancestor death detection time.

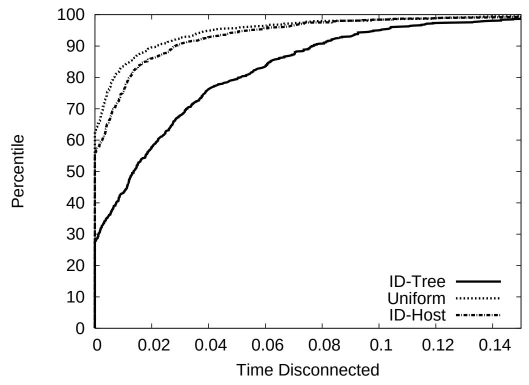

Fig. 14. Cumulative distribution of fraction of session for which a host is disconnected in one or more trees, for hosts with a contribution parameter 2 for *Slashdot* trace. Multi-tree delivery with four trees is assumed. The lowest curve is *ID-Tree*. The top two curves are *Uniform,* and *ID-Host*.

The primary metric we consider is the average loss rate experienced by a host, obtained by sampling the loss rate of the host at various instants, and computing the mean. Our sampling rate is once every 200 milliseconds – this parameter was chosen to ensure it was sufficiently smaller than the reconnection times we assume (5 seconds), yet large enough not to be computationally intensive. In a single tree, the loss rate of a host at a particular instant is either 0% or 100% depending on whether the host is connected to the source, or not. With multi-trees, we assume the loss rate at an instant is $i/T * 100\%$, where i is the number of trees in which the host is disconnected at that instant.

Fig. 11 depicts the cumulative distribution of the loss rate of all hosts for the three policies, using the *Slashdot* trace, and assuming multi-tree data delivery with four trees. The performance of all three schemes are comparable. The performance of *ID-Tree* and *ID-Host* are almost indistinguishable, while *Uniform* performs slightly worse. For example, roughly 30% of hosts see no loss in all schemes, and 90% of hosts see a loss of less than 2% with *ID-Host,* and less than 4% with *Uniform*.

Fig. 12 depicts the cumulative distribution of loss rate for only the high contributor hosts. Recall that we would like these hosts to have better performance than the rest of the hosts.

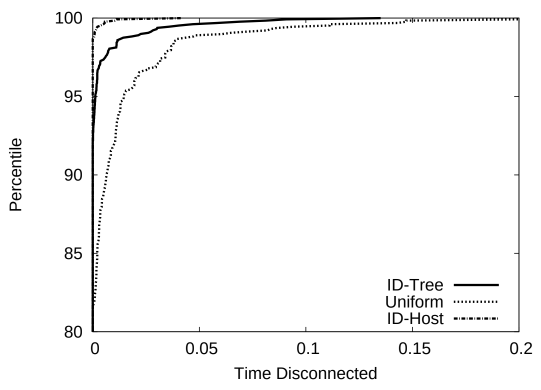

Fig. 15. Cumulative distribution of fraction of session for which a host is disconnected in two or more trees for hosts with a contribution parameter 2 for the *Slashdot* trace with Multitree delivery. The lowest curve is *Uniform*. The top two curves are *ID-Tree* and *ID-Host*.

With the *ID-Tree* scheme, the high contributor hosts have the same performance as all other hosts, and the curve is similar to corresponding curve in Fig. 11. However, the performance of these hosts is significantly improved in the *ID-Host* scheme. For example, with *ID-Host* 60% of the hosts see no loss, and 90% of hosts see a loss rate of less than 1%. On the other hand, only 30% of hosts see no loss, and 60% of hosts see loss less than 1% with *ID-Tree*. We also observe that *Uniform* does comparably to *ID-Host*, and better than *ID-Tree*.

As per our discussion in Section II-D, our loss rate model assumes that when a parent departs, the child is disconnected for T seconds before attempting to join a new parent. T models the detection time, assuming the parent makes an abrupt departure. Our loss rate numbers so far use a default value of T of 5 seconds as discussed in Section II-D. Fig. 13 studies the sensitivity to T. The x-axis is the detection time. The y-axis is the 90th-percentile of the loss rates taken for the high-contributor hosts for various schemes. For all T values, the loss rate with *ID-Host* is very small, however, the loss rate of *ID-Tree* decreases with lower T values. Thus, while the magnitude of performance improvement is sensitive to the value of T used, the overall trend remains consistent.

Our results so far sample the loss rates with multi-tree solutions by assuming that a loss in i trees at any instant corresponds to a loss of $i/T * 100$. Another metric of interest is the fraction of the session where a host is disconnected in some number of trees. In scenarios with limited redundancy, a loss in any tree translates to some degradation in quality, while in scenarios with redundancy, a loss in a small number of trees can be tolerated. Fig. 14 depicts the the cumulative distribution of the percentage of the session for which hosts are disconnected in one or more trees for the high contributors. The benefits of host prioritization are significant. For *ID-Host* and *Uniform*, over 90% of hosts are disconnected in one or more trees for less than 4% of the session. However, with *ID-Tree*, the 90th-percentile value is up to 12% of the session. Fig. 15 shows the cumulative distribution of the fraction of the session in which high-contributor hosts are disconnected

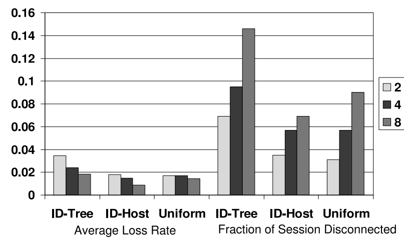

Fig. 16. The 90th-percentile loss rate and fraction of session for which hosts are disconnected in one or more trees as a function of the number of trees used in the multi-tree schemes. Only hosts with a higher contribution parameter are considered.

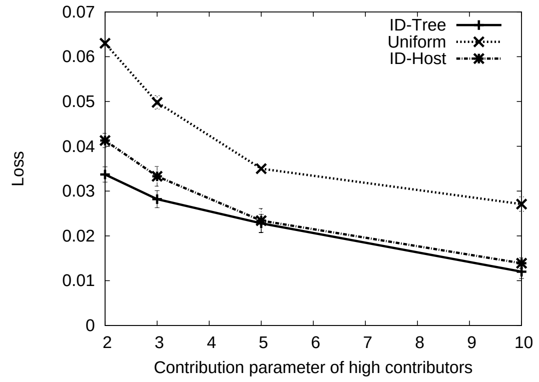

Fig. 17. 90th-percentile loss rates of all hosts in the *Slashdot* trace, varying the contribution parameter of the high-contributor hosts. Low contributors have a fixed contribution parameter of 1.

in two or more trees. The benefits for *ID-Host* are less significant, but still hold. Of high-contributors, 82% are never disconnected in multiple trees with *Uniform*, 93% with *ID-Tree*, and 98% with *ID-Host*.

Fig. 16 considers the impact of the number of trees on the performance by presenting the 90th-percentile performance in terms of both loss rate and fraction of session for which hosts are disconnected in one or more trees. Only the high contributors are considered. For all three schemes, the average loss improves with an increase in the number of trees, but the fraction of period for which hosts are disconnected increases. This is expected because increasing the number of trees with multiple trees has the potential to improve the average performance due to the greater resiliency. But, with a larger number of trees, it is also more likely that some tree is disconnected. In addition, the *ID-Host* policy performs the best in both metrics consistently across the board.

Our results so far have assumed that high-contributor hosts have a contribution parameter K of 2 – they contribute 2 times what they receive across all trees. In the following experiments, we consider the impact of varying K. Note that we only vary the contribution of high contributors. Low contributors have a fixed contribution parameter of 1. Fig. 17

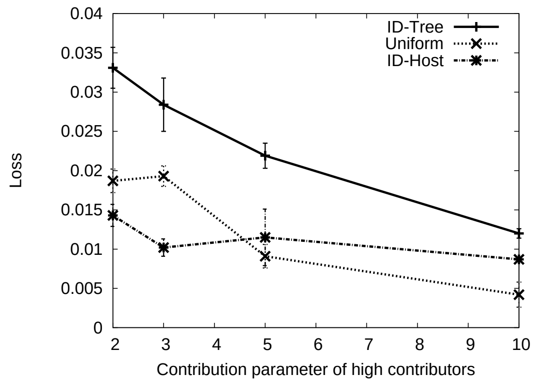

Fig. 18. 90th-percentile loss rates of high contributors in the *Slashdot* Trace, varying the contribution parameter of the high-contributor hosts. Low contributors have a fixed contribution parameter of 1.

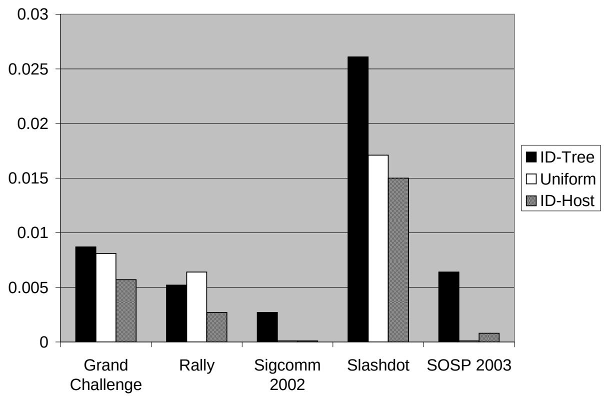

Fig. 19. 90th-percentile of loss rates for high contributors in various traces.

depicts the 90th-percentile loss rate of all hosts against the contribution parameter K of the high contributors. Each point represents the mean of five runs along with 95% confidence intervals. For all values of K, *ID-Tree* performs the best, and *Uniform* performs significantly worse. *ID-Host* performs slightly worse than *ID-Tree*, and is almost indistinguishable as K increases. Examining high-contributor performance in Fig. 18, *ID-Host* and *Uniform* perform better than *ID-Tree* for most K values, though the difference decreases for larger K. We have also conducted experiments fixing the contribution parameter of the high-contributor hosts, and varying the low-contributor hosts' contribution parameter, and our results are similar.

D. Sensitivity to trace

Our results so far demonstrate that with the *Slashdot* trace, *ID-Host* provides better performance to hosts that contribute more, at a modest cost to average performance of the system. Fig. 19 illustrates the performance of the high contributors with various schemes over the entire set of traces. The performance of all schemes is better in other traces compared to Slashdot. This is due to both the composition and the dynamics associated with the traces. As Table I indicates, for both the *Sigcomm2002* and the *SOSP2003* broadcasts, over half the nodes are high contributors. While the *GrandChallenge* and

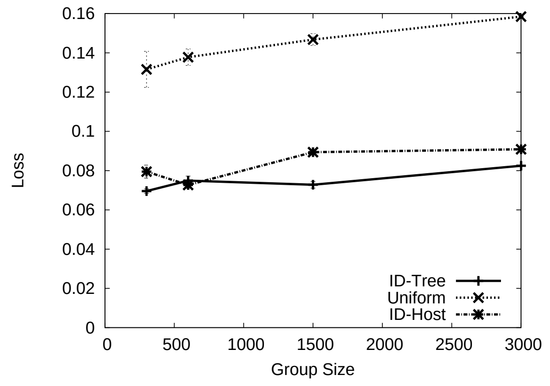

Fig. 20. 90th-percentile loss rates of all hosts, varying the scale of the synthetic workload.

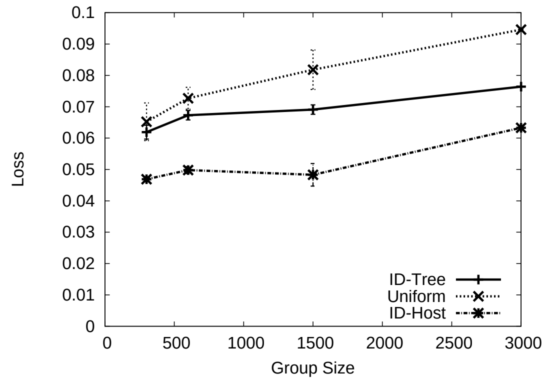

Fig. 21. 90th-percentile loss rates of high contributors, varying the scale of the synthetic workload.

Rally broadcasts have similar composition as *Slashdot* they have a significantly higher session duration. Across all traces, *ID-Host* performs better than *ID-Tree* when performance of high-contributor nodes is considered. The benefits are more significant in broadcasts like *Slashdot*, which have both a small fraction of high-contributor nodes and significant dynamics. We have also considered the performance of the entire set of nodes. While we do not present results, we observed that *ID-Host* and *ID-Tree* perform similarly, and better than *Uniform*.

E. Scaling Properties

Given the limited peak size of all our real traces, we also evaluated our schemes with synthetic traces. We model node arrival using a poisson process, and node session durations using a pareto process. These choices are based on group dynamics observed in overlay multicast deployments [15], Mbone measurements [19] and content delivery networks [18]. For the pareto distribution, we assume a mean stay time of 300 seconds, and an alpha value of 1 for the pareto distribution. We vary the join rate of the poisson process between 1 and 10 nodes per second, leading to group sizes varying from 300 to 3000 nodes.

Fig. 20 plots the performance of the three schemes as a function of number of nodes. The 90th-percentile loss across the entire set of nodes is considered. *ID-Host* continues to closely track the performance of *ID-Tree* as the population increases. The difference remains small, and both schemes perform better than *Uniform*. Fig. 21 considers the 90th-percentile loss taken across the high contributors. While all schemes perform slightly worse with larger group sizes, *ID-Host* performs better than *ID-Tree* over all group sizes.

V. Conclusions

In this paper, we highlight the need to systematically consider prioritization as a key criterion in the design of protocols for overlay multicast. Our observation is motivated by data that indicates heterogeneity in node characteristics, such as their outgoing bandwidth and stay time durations. We show that considering such prioritization is important, both to improve the overall performance of the system and to ensure that nodes which contribute more to the system receive better performance.

We identify two important contexts where prioritization heuristics can have significant benefit, yet the trade-offs between multiple prioritization heuristics are not clear. We conduct a systematic study of the trade-offs using both real trace data, and sensitivity studies using synthetic workloads. To the best of our knowledge, this is the first work to identify and systematically study these trade-offs, and to demonstrate the potential benefits of the resulting prioritization heuristics.

First, we considered single-tree data delivery given data that not only indicates heterogeneity in node degree, and node stay times, but also indicates a lack of correlation between the contribution of the node, and the duration a node stays [18], [15]. Our study is set in the context of a large class of interesting fixed-duration broadcasts, where evidence from several measurement studies [18], [15], [19] indicates nodes with a higher age tend to stay longer. Our results indicate clear-cut benefits to degree-based prioritization, particularly in regimes where there is a large fraction of hosts behind low-bandwidth connections. Further, while there are some benefits to using age-based prioritization over no prioritization, a systematic study of various hybrid degree vs. age heuristics indicates marginal performance benefits over degree-based prioritization.

We then considered multi-tree data dissemination frameworks. While traditionally multi-tree based approaches have considered homogeneous models, in this paper we considered environments with node heterogeneity, with nodes contributing differently based on their outgoing bandwidth. To our knowledge, this is one of the first works to study the impact of multi-trees under heterogeneous bandwidth environments or to identify and investigate the trade-offs between various prioritization policies in a multi-tree context. Overall, our results indicate that employing prioritization heuristics optimized for single-tree scenarios is insufficient and that it is important for the heuristics to consider the overall contribution of a node rather than its contribution in the particular tree alone. For example, in experiments involving multi-tree delivery using 4 trees with the *Slashdot* trace, 90% of higher degree nodes see a loss rate of less than 1% with *ID-Host*, while only 60% of nodes see loss less than 1% with *ID-Tree*. However, the performance of *ID-Host* and *ID-Tree* is practically indistinguishable when all nodes in the system are considered.

Our results highlight the potential benefits of carefully constructed prioritization policies in improving application performance. We are in the process of incorporating these heuristics into an actual system, and evaluating them over an Internet test-bed. We believe the potential benefits of prioritization are even more significant in such environments, when network losses are considered. Our future work includes investigating mechanisms needed to make these prioritization schemes practical, such as avoiding prioritization of nodes with high delay and mechanisms to automatically identify the contribution of a node to prevent untruthful nodes from being falsely prioritized.

Acknowledgments: We thank Venkata N. Padmanabhan for several suggestions that helped shape this paper, and for comments on drafts of the paper.

References

[1] Y. Chu, S. G. Rao, and H. Zhang, "A Case for End System Multicast," in *Proceedings of ACM Sigmetrics*, June 2000.

[2] P. Francis, "Yoid: Extending the Internet Multicast Architecture," Apr 2000.

[3] J. Jannotti, D. Gifford, K. L. Johnson, M. F. Kaashoek, and J. W. O. Jr., "Overcast: Reliable Multicasting with an Overlay Network," in *Proceedings of the Fourth Symposium on Operating System Design and Implementation (OSDI)*, Oct. 2000.

[4] D. Pendarakis, S. Shi, D. Verma, and M. Waldvogel, "ALMI: An Application Level Multicast Infrastructure," in *Proceedings of 3rd Usenix Symposium on Internet Technologies & Systems (USITS)*, March 2001.

[5] K. Sripanidkulchai, A. Ganjam, B. Maggs, and H. Zhang, "The Feasibility of Supporting Large-Scale Live Streaming Applications with Dynamic Application End-Points," in *Proceedings of ACM SIGCOMM*, 2004.

[6] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable Application Layer Multicast," in *Proceedings of ACM SIGCOMM*, Aug. 2002.

[7] M. Castro, P. Druschel, A. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "SplitStream: High-bandwidth Content Distribution in Cooperative Environments," in *Proceedings of SOSP*, 2003.

[8] M. Castro, P. Druschel, A. Kermarrec, and A. Rowstron, "Scribe: A Large-Scale and Decentralized Application-Level Multicast Infrastructure," in *IEEE Journal on Selected Areas in Communications Vol. 20 No. 8*, Oct 2002.

[9] D. Kostic, A. Rodriguez, J. Albrecht, and A. Vahdat, "Bullet: High Bandwidth Data Dissemination Using an Overlay Mesh," in *Proceedings of SOSP*, 2003.

[10] J. Liebeherr and M. Nahas, "Application-layer Multicast with Delaunay Triangulations," in *Proceedings of IEEE Globecom*, Nov. 2001.

[11] V. Padmanabhan, H. Wang, P. Chou, and K. Sripanidkulchai, "Distributing Streaming Media Content Using Cooperative Networking," in *Proceedings of NOSSDAV*, May 2002.

[12] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker, "Application-level Multicast using Content-Addressable Networks," in *Proceedings of NGC*, 2001.

[13] W. Wang, D. Helder, S. Jamin, and L. Zhang, "Overlay Optimizations for End-host Multicast," in *Proceedings of Fourth International Workshop on Networked Group Communication (NGC)*, Oct. 2002.

[14] S. Q. Zhuang, B. Y. Zhao, J. D. Kubiatowicz, and A. D. Joseph, "Bayeux: An Architecture for Scalable and Fault-Tolerant Wide-Area Data Dissemination," in *Proceedings of NOSSDAV*, Apr. 2001.

[15] Y. Chu, A. Ganjam, T. S. E. Ng, S. G. Rao, K. Sripanidkulchai, J. Zhan, and H. Zhang, "Early Experience with an Internet Broadcast System Based on Overlay Multicast," in *Proceedings of USENIX*, June 2004.

[16] "Tmesh broadcast system," http://warriors.eecs.umich.edu/tmesh/tmeshv.html.

[17] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum, "DONet/CoolStreaming: A Data-driven Overlay Network for Live Media Streaming," in *Proceedings of IEEE INFOCOM*, 2005.

[18] K. Sripanidkulchai, B. Maggs, and H. Zhang, "An Analysis of Live Streaming Workloads on the Internet," in *Proceedings of Internet Measurement Conference*, 2004.

[19] K. C. Almeroth and M. H. Ammar, "Collecting and Modeling the Join/Leave Behavior of Multicast Group Members in the MBone," in *Proceedings of International Symposium on High Performance Distributed Computing (HPDC)*, Aug. 1996.

[20] M. Yang, Z. Zhang, X. Li, and Y. Dai, "An empirical study of free-riding behavior in the maze p2p file-sharing system," in *4th International Workshop on Peer-to-Peer Systems*, 2005.