Scaling End-to-end Multicast Transports with a Topologically-sensitive Group Formation Protocol

Sylvia Ratnasamy
University of California, Berkeley
sylviar@cs.berkeley.edu

Steven McCanne
University of California, Berkeley
mccanne@cs.berkeley.edu

Abstract

While the IP unicast service has proven successful, extending end-to-end adaptation to multicast has been a difficult problem. Unlike the unicast case, multicast protocols must support large and heterogeneous receiver sets. While proposed approaches to scaling multicast transports attempt to localize problems and/or organize receivers into a hierarchy through a divide-and-conquer approach, this approach succeeds only if the resulting hierarchy is congruent with the underlying routing tree topology. This implies the need for some level of topological information at the end systems which the IP multicast service deliberately hides.

In this paper, we present a Group Formation Protocol (GFP) whereby receivers dynamically organize themselves into a multi-level hierarchy of multicast groups that corresponds to the underlying routing tree. GFP can serve as a core component across a wide range of multicast applications and protocols such as local recovery for reliable multicast, self organized transcoding, self organizing web caches, the optimal and dynamic placement of proxies, repeaters, designated receivers, recorders and so forth.

1 Introduction

The core foundation for the great technical success of the Internet is quite arguably the *end-to-end design principle* [24]. When applied to internetworking of heterogeneous components and end-systems, this principle naturally leads to one of the most critical Internet design choices, namely its *best effort* network-layer delivery semantics. In this framework, end-to-end reliability cannot be wholly and correctly implemented at the network layer so network components are deliberately allowed to drop, randomly delay, replicate, or reorder packets, and richer services like reliable, sequenced delivery must be built on top of this unreliable delivery model. But because the requirements imposed on a best-effort network are so minimal, a seemingly very difficult problem (i.e., interconnecting millions of heterogeneous components across a world-wide internetwork) becomes much simpler: the federation of packet routers spread across the network need only make a best effort attempt at delivering packets toward their ultimate destination.

This end-to-end approach to network design cuts across the Internet architecture and has proven enormously successful in the decomposition of the TCP/IP protocol suite. The TCP/IP division of labor between a *reliable* closed-loop transport service and an *unreliable* open-loop network service has enabled the incremental engineering and deployment of very large-scale networks and ultimately led to the success of the TCP-based global Internet [4].

Naturally then, when Deering proposed IP Multicast [5], an extension to the Internet architecture to support efficient multi-point packet delivery, he appealed quite deliberately to this very same end-to-end design methodology. Like unicast IP, the IP Multicast service provides only best effort packet delivery and richer multicast services must be built on top of this unreliable delivery model. But unlike unicast, in which a single source and receiver intercommunicate via a single (possibly dynamic) path through the network, in a multicast *session* with N hosts, there are roughly N^2 paths, one between every source host and every other recipient host.

This multiplicity of paths imposes great difficulty on the design of end-to-end multicast transports. The hallmark characteristic of end-to-end transport protocols like TCP is their ability to dynamically adapt to the network path and thereby match the load imposed by the end-systems to the resources available within the network. But "resource availability" is ill-defined in multicast because there is no single path and no single measure of available resource.

In addition to the challenges imposed by *heterogeneity*, multicast communication also induces challenging problems that center upon *scale*. In a large multicast session, there are many and diverse sources of data and control and a naive protocol design can easily generate uncontrolled traffic transients and effects like the well-known *feedback implosion* problem, where a large number of receivers simultaneously generate feedback messages addressed to a single source. Fortunately, many of these problems of scale and heterogeneity are fairly well understood and a great deal of existing research addresses them [25, 18, 7, 27, 15, 11, 16]. Yet despite all this research, end-to-end transport protocols for multicast that gracefully accommodate heterogeneity, perform well at very large scale, and include proven, Internet-compatible congestion control strategies still remain quite far from our grasp.

Given that no clear-cut, viable solution has emerged from all this work on multicast transports, we believe it is critical to better understand the fundamental difficulties that are carried within this problem space. A common trait of many proposed end-to-end multicast is some scheme for exploiting the structure and topology of the underlying multicast distribution tree (e.g., Liu's work on local recovery [16] attempts to cluster receivers in common sub-trees while RLM [18] induces receivers that share common paths to subscribe to equivalent rate signals). Essentially, these prior works attempt to tackle both heterogeneity and scale simultaneously with a *divide and conquer* approach, e.g., by creating multiple loss recovery groups arrange in a hierarchy to localize the impact of control traffic. In this way, one large, heterogeneous, and difficult problem is reduced to many small, homogeneous, and simpler sub-problems. The goal of our work is to assess the raw difficulty of hierarchically decomposing a multicast distribution tree in this fashion in order to determine the viability of end-to-end transports built on this principle.

Given the simplicity of the IP Multicast service model, how could we realistically perform this hierarchical decomposition? We can think of the multicast service as a simple cloud-based model, where a source injects a packet into the cloud and each receiver receives a copy of that packet. In other words, the service model deliberately hides the underlying topology from the end-systems, which means that we have no easy mechanism for building an efficient hierarchy. If we could discover the underlying topology, our problem would become almost trivial. That is, if we knew this topological structure, we could carry out a variety of topologically-sensitive protocol optimizations. For example, with explicit knowledge of the underlying multicast distribution tree, we could form local recovery groups for reliable multicast; place "designated receivers" (DRs) in RMTP [15]; optimally position video transcoders in SOT [11]; or optimally overlap multicast-based Web caches [10, 19].

But IP deliberately hides topology and we have no easy mechanism to discover it, which begs the question: should we extend the IP service model, or in particular the IP multicast service model, with mechanisms for explicitly acting upon or discovering the network's underlying topological structure? Though we believe this question will remain open for some time, in this paper we take the end-to-end position and argue that much of the structure of an underlying multicast distribution tree can be inferred through observations made from only within end-systems. Toward this end, in prior work [22], we investigated the question of how well one could infer a multicast distribution tree strictly from end-to-end observations, and found that with a relatively simple model based on shared loss patterns and probabilistic analysis that eliminated "false sharing" signals, we could do remarkably well.

Though our tree inference algorithm performs well in theory, it assumes a global exchange of information among all the participants in a multicast session, wherein an omniscient, centralized agent has complete knowledge of each receiver's loss patterns from the beginning of time. Though unrealistic in practice, we adopted this hypothesis to explore the fundamental limitations of an inference-based approach. The next challenge, which we tackle herein, is how we might leverage this ability to infer multicast tree topology in a practical protocol framework.

Toward this end, we propose herein a core protocol building block that can be used across a wide variety of end-to-end multicast transports to address the performance problems imposed by heterogeneity and scale. We describe, simulate, and evaluate a distributed Group Formation Protocol (GFP) that leverages our earlier work on multicast tree inference to produce a topologically-sensitive protocol primitive.

2 Inferring Structure from Shared Loss

State of the art end-to-end multicast transports exploit the structure and topology of the underlying multicast distribution tree in an attempt to tackle both heterogeneity and scale. For example, landmark works by Liu [16] and Kouvelas [11] leverage the fact that the multicast tree structured delivery model induces correlations in the end receiver packet losses to cluster together co-located receivers through "shared loss thresholding". In these schemes, every receiver has an associated *loss fingerprint* or *lossprint* which is simply a list of sequence numbers of packets lost by that receiver. Receivers exchange lossprints over the multicast channel and monitor their local loss rate. If a receiver experiences a sufficiently high rate of loss, it initiates the formation of a new multicast group. Other receivers join the new group if their lossprints share enough similarity with the group initiator's lossprint according to a thresholding rule.

Clustering receivers into groups through shared loss thresholding is thus built on the base assumption that shared loss implies spatial (topological) correlation. However, this assumption can fail in a subtle and misleading way. This is illustrated in Fig 1 where a source transmits to three receivers A, B and C. Receivers A, B and C share link L_1 and only A and B share L_2. If links L_3 and L_4 have high

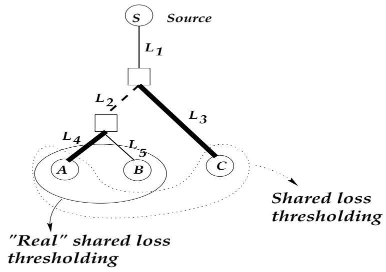

Figure 1. Shared Loss Pathology

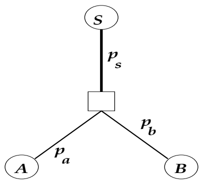

Figure 2. Loss Model

loss rates and L_5 has a relatively low loss rate then it is entirely possible that the probability of seeing shared losses between receivers A and C exceeds that between A and B which would lead us to falsely conclude that A and C are more closely located than A and B. This *shared loss pathology* occurs because high loss rate links can cause physically disparate receivers to have higher shared loss than physically co-located receivers. Given this shared loss pathology, is it possible to somehow extract the true loss correlations between receivers?

In [22] we demonstrated that this can indeed be achieved through the use of a simple loss model that correctly estimates topological proximity based on observed receiver loss patterns. Since this work forms the base for the work described in this paper, we briefly outline the key results.

Shared losses arise in two ways: either due to a shared path or due to mere coincidence. To clarify, consider a pair of receivers A and B (Fig 2) that share the path from the source to their closest common ancestor. Packets lost along this shared path are truly indicative of the underlying tree structure. We call these *real* shared losses. In addition to real shared losses, distinct copies of the same packet can be lost along the independent paths from the closest common ancestor to each receiver. These coincidental shared losses are *not* caused by the underlying tree structure. We call these *false* shared losses. The failure modes that arise in the use of net shared loss as metric for spatial locality are due to this "false sharing". *Real* shared losses, on the other hand, are a robust metric of spatial locality because the greater the extent of the shared path between a pair of receivers, i.e. the more closely located a pair of receivers, the greater will be the real shared losses between them. At the end host however, a real shared loss is indistinguishable from a false one. However we can overcome this problem by utilizing lossprints and the simple loss model of Fig 2 to separate out the real shared loss from the total shared loss observed between a pair of receivers.

Packet losses are modeled as being independent. The p's next to each link in Fig. 2 represent the individual link loss probabilities. In particular, p_s represents the probability of seeing real shared loss between receivers A and B. The above model yields the following equations:

$$\begin{aligned} P_{ab} &= p_s + (1-p_s)p_a p_b \\ P_{a\bar{b}} &= (1-p_s)p_a(1-p_b) \\ P_{b\bar{a}} &= (1-p_s)(1-p_a)p_b \end{aligned}$$

where, P_{ab} is the probability of a shared loss (whether real or false) between A and B and $P_{a\bar{b}}$ and $P_{b\bar{a}}$ are the probabilities of a loss at one receiver but not the other. The left hand side P's can be estimated from the receiver lossprints. Solving the above equations we can derive the real shared loss probability p_s. Using real shared loss probabilities, we found that by assuming global knowledge of receiver lossprints it was possible to infer the logical topology of the multicast routing tree. A detailed description of the tree inference algorithm and its performance can be found in [22].

3 A Group Formation Protocol

In a practical protocol, the tree inference algorithm's assumption of global knowledge of receiver lossprints would lead to infeasible control traffic overhead. Fortunately, the use of subgroups in multicast applications and protocols is an optimization that need not be perfect at all times to dramatically enhance performance. Thus, we can aim for a coarse grained clustering of co-located receivers without requiring detailed knowledge of the exact routing tree topology. In this section we apply the lessons learned from the extreme case analysis of Section 2 to the design of a practical Group Formation Protocol (GFP) that achieves this coarse-grained clustering i.e. we relax the exchange of information at the cost of decreased topological accuracy and yet do so in a controlled fashion. .

The goal of GFP is to have the receiver set self-organize into a multi-level hierarchy of multicast groups wherein individual groups correspond to the homogenous sub-regions within the heterogeneous multicast tree and the hierarchy imposed is congruent with the underlying multicast routing tree. The metric used to identify group structures is the real shared loss probabilities introduced in Section 2. We have implemented a prototype protocol to explore the question of whether this basic loss based approach is feasible. A number of variations and enhancements to this basic approach are conceivable and would provide different tradeoffs.

3.1 Terminology

In order to facilitate the explanation of the detailed working of GFP, we first introduce some requisite terminology:

- Receiver LossPrint (LP_i) : The ordered list of packets lost by receiver *I*.
- Group LossPrint (LP_g) : The ordered list of packets lost by the group as a whole as reported by the group's representative.
- Receiver's Working LossPrint ($LP_i - LP_g$) : The ordered list of packets lost by a receiver *within* its current group which is estimated as the losses seen by a receiver but not by its group representative.
- Receiver's Working Loss Rate (LR_i) : Receiver I's loss rate calculated with respect to its working lossprint. LR_i estimates the loss rate experienced by receiver I *within* its current data group G_i.

A lossprint further includes the sequence number of the last packet received.

3.2 Group Formation

The initial multicast group structure associated with GFP consists of a single, well known, pervasive control group and a single data group. Using GFP the single data group evolves into multiple topologically localized data groups. At all times, a receiver is a member of a single data group and the common control group.

In our prototype protocol, the source transmits probe packets onto the common control group at a low rate that can be handled by even the lowest bandwidth path in the tree. An alternative future implementation could, in some way, use the actual data transmission rather than inject additional probe traffic. The losses experienced by the receivers on this control group drive the group formation process. While the group formation process is driven by the losses experienced on the common control group, the protocol messages (e.g.: group initiation messages) sent out by a receiver are multicast only to its local data group rather than globally. GFP is thus run independently within each data group but always with respect to the losses observed on the control group on account of which the derived structure corresponds to the routing tree rooted at the source. Figure 3 illustrates the overall operation of GFP. A receiver *I* initiates the formation of a new data group G_1 because it experiences significantly high loss caused by L_1. Receivers downstream from L_1 join G_1. G_1 lies at the first level in the hierarchy of data groups. The existence of an additional bottleneck causing loss (L_2) within G_1 causes receiver J to subsequently initiate the formation of a new group G_2. This group initiation message is multicast to G_1. The subset of receivers in G_1 downstream from L_2 now leave G_1 and join G_2. G_2 lies at the second level in the hierarchy of data groups. This process continues until every receiver settles into a stable state data group *within* which there are no significant bottlenecks. We first describe in detail how GFP operates within a single data group,how representatives are elected and finally how the pieces fit together to yield a multi-level hierarchy.

3.2.1 Group formation within a single data group

A receiver I builds its individual lossprint LP_i from the losses experienced on the control group. Every local data group has a group representative (described in Sec.3.2.2) that periodically multicasts a group lossprint (LP_g) onto its data group. A receiver factors out the reported group lossprint from its individual lossprint to obtain its working lossprint ($LP_i - LP_g$) that estimates the losses suffered by the receiver within its current data group.

Every receiver sets a timer as a linearly increasing function of its working loss rate. On timeout, a participant may initiate the formation of a new multicast group, join a previously proposed group, or take no action. A receiver may initiate the formation of a new data group only if its *working* loss rate exceeds a predefined loss rate threshold LR_{min} Group initiation is done by multicasting an INIT(G) message onto the receiver's current data group proposing the formation of a new data group G. The INIT(G) message includes the group initiator's working lossprint.

For every incoming INIT message, a participant I computes $p_s(i,g)$, the real shared loss probability between itself (receiver I) and the proposed group G using the model described in Section 2. $p_s(i,g)$ estimates the loss that receiver I shares with the proposed group G along the common path from the source to the closest common ancestor of I and the group's initiator. Until timer expiry, a receiver I tracks the proposed group G_{max} with which it shares the maximum value of $p_s(i,g)$, i.e. p_s^{max} and G_{max}. On timeout, if the receiver's value of p_s^{max} exceeds a predefined threshold *THRESH*, it joins the group G_{max}. If not, it initiates the formation of a new group G_{new} provided its working loss rate LR_i exceeds LR_{min}.

In summary, on timeout, a receiver I which is currently a member of the data group G performs the following check:

If ($p_s^{max} >$ *THRESH*)
    leave G
    join G_{max}
else-if ($LR_i > LR_{min}$)
    # I initiates the formation of G_{new}
    Multicast *INIT* message (*INIT* $/G_{new}/(LP_g - LP_i)$) onto G
    leave G
    join G_{new}

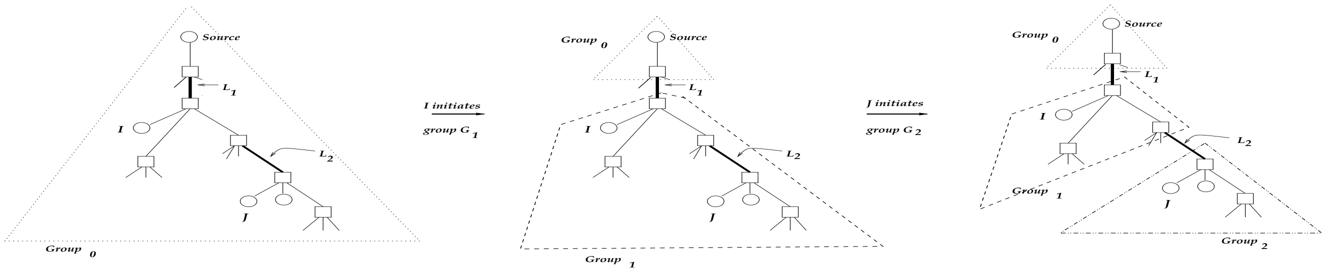

Figure 3. Evolution of a multi-level hierarchy

else
do nothing

3.2.2 Selection of Group Representatives

This section describes the election of group representatives and their role in GFP. The ideal group lossprint would consist of exactly those losses incurred along the shared path between the source and the closest common ancestor of all the group members. While the ideal group lossprint could be computed as the intersection of each receiver's lossprint, this requires global knowledge of receiver lossprints which is infeasible in practice. Instead, GFP elects a single representative per data group that is responsible for announcing the group lossprint to its entire data group We choose the lowest loss rate receiver within the group to serve as the representative because such a receiver's lossprint most closely matches the ideal group lossprint. Further, the lowest loss rate member of the group is the best connected to the source and is likely to be the further upstream in the underlying routing tree. As shall be described in Section 5 such a receiver plays a key role in achieving reliability and congestion control in a number of proposed multicast applications and protocols. To ensure that the initiator of a new data group has the lowest loss rate within that group timer values are selected as a linearly increasing function of the receiver working loss rates. Thus, the initiator can serve as the group representative thereby greatly simplifying the election of group representatives.

The use of a representative-based model induces a single point of failure. We rely on the application of "soft state" principles to achieve robustness in the face of representative crashes. A representative periodically multicasts *REP* messages announcing the group lossprint to its data group. Every group member sets a timer with value directly proportional to its working loss rate. If a receiver sees a *REP* message before its timer expires, it cancels its timer. If not, then on time out, a receiver assumes that the representative has stopped or crashed and starts transmitting *REP* messages. This ensures that when a representative crashes, the next lowest loss rate receiver is promoted to representative.

In addition to periodic *REP* messages, group representatives periodically multicast *INIT* messages onto the common control group at a very low rate. *INIT* messages contain the representative's local data group address and the representative's lossprint LP_r. As described in sections 3.3 and 3.4, these messages allow for adaptation to changing routing topologies, incorrect grouping decisions etc. Further, these *INIT* announcements allow late joiners in the session to learn about existing groups and find their correct data group through the normal application of the GFP procedure described in the previous section. As in any "announce-listen" [1, 21] type protocol, the rate at which these periodic *INIT* control messages are announced involves a trade-off between the amount of the receiver's bandwidth allocated to control traffic and the latency with which receivers react to dynamic changes in routes, recover from incorrect grouping decisions etc.

3.2.3 Evolution into a multi-level hierarchy

We now describe how the group formation process within a single group and the representative election process work together to yield a multi-level hierarchy. The initial data group in which all receivers start out constitutes the first level in the hierarchy of data groups. There is no group representative and the group lossprint is empty. Thus, a receiver's initial working lossprint is its own lossprint itself. The presence of bottlenecks within the original data group that are not shared by the entire group causes subsets of receivers below common bottlenecks to break off and form new data groups by the application of the process described in section 3.2.1. These new groups form the second level members of the hierarchy.

On formation of a new data group, the group initiator serves as the group representative and periodically multicasts its individual lossprint onto the new data group. The representative's lossprint serves as the group lossprint. As the group initiator (representative) is itself downstream of the common bottleneck, the group lossprint now includes the losses suffered along the bottleneck. Consequently, the working lossprints computed by the new group members no longer reflect the losses suffered along the shared bottleneck thus yielding lower working loss rates. If however, there still exist lossy paths *within* a new data group then the subset of receivers below such paths will still have significantly high working loss rates and will break off into a new data group one level lower in the hierarchy. If there are no more bottlenecks within the data group, the receiver working loss rates will be below LR_{min}, no new groups will be initiated and the group structure will stabilize.

GFP is thus run independently within each data group and the multi-level hierarchy evolves in parallel across the receiver set until every receiver settles into its final stable state data group within which there are no significant bottlenecks.

3.3 Recovery from incorrect grouping decisions

The limited number of loss samples per exchanged lossprint could result in occasional incorrect grouping decisions. Hence a recovery process is required to correct bad grouping decisions. The recovery algorithm is a two-step process. First, the receiver detects that it has made a mistake in joining its current data group. Having done so it then chooses a better group to join.

A member I of a group G_{l-1} at level $l-1$ joins a group G_l at level l because the estimated shared loss between the representative of G_l and receiver I along the common path from the representative of G_{l-1} exceeds the value *Thresh*. Thus, a receiver I in a level l group should estimate a shared loss rate of at least l times *Thresh* between itself and its current group representative. For every incoming *REP* message containing group lossprint LP_g, a receiver I with lossprint LP_i recomputes the real shared loss probability $p_s(i, g)$. If receiver I finds that its estimate of $p_s(i, g)$ is repeatedly less than $l * Thresh$, it concludes that it is in an incorrect data group and leaves the group G_l. To decide upon which group to join, the receiver simply runs GFP as normal and listens for new *INIT* messages. By listening to the periodic *INIT* announcements on the control group, the receiver can by the application of the normal GFP procedure discover its appropriate data group.

3.4 Adapting to changes in link loss rates and routing topologies

We expect GFP to operate over large time-scales for multicast sessions that are fairly long-lived. As such, the GFP group structure needn't adapt to transient network congestion and other short-lived variations in network conditions. Yet, GFP must adapt to long-lived variations in network conditions and routes. On account of changing link loss

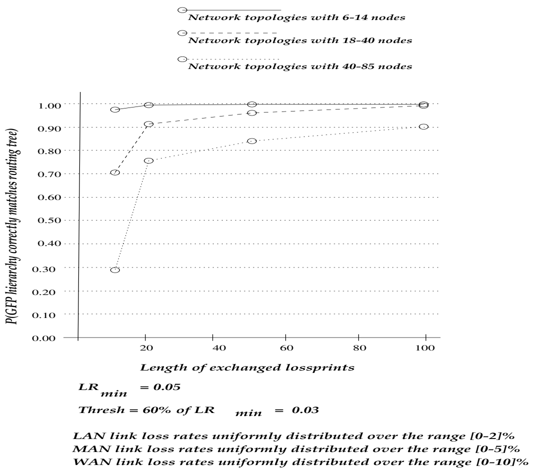

Figure 4. GFP performance for different size topologies

rates and routing topologies, new bottleneck links may appear and existing ones dissapear. The emergence of a new bottleneck link raises no new problems because the downstream receivers will break off into a new data group by the normal application of GFP. The dissapearance of a previously existing bottleneck however implies that two data groups should ideally be merged to control the number of data groups. The process for merging two equivalent groups is as follows:

- For every REP message received on its local data group G, a receiver I computes the real shared loss probability $p_s(i, g)$.
- For every incoming $INIT(G_{new})$ message received on its control group, a receiver I computes the real shared loss probability $p_s(i, new)$.
- If $p_s(i, new)$ is repeatedly within $\pm 10\%$ of $p_s(i, g)$ then the groups G_{new} and G are treated as being essentially equivalent and the receivers switch to the group with the lower IP address.

Thus, GFP builds a hierarchy of multicast groups wherein every receiver *independently* discovers its local group, recovers from incorrect grouping decisions and monitors periodic group announcements such that the group structure as a whole adapts to variations in network conditions.

4 Evaluation

4.1 Accuracy of GFP

We implemented GFP in the VINT network simulator *ns* [17], using the TIERS topology generator [6].

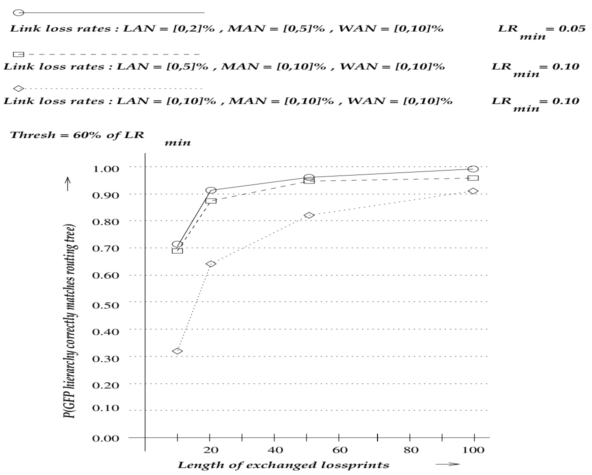

Figure 5. GFP performance for different distributions of link loss rates

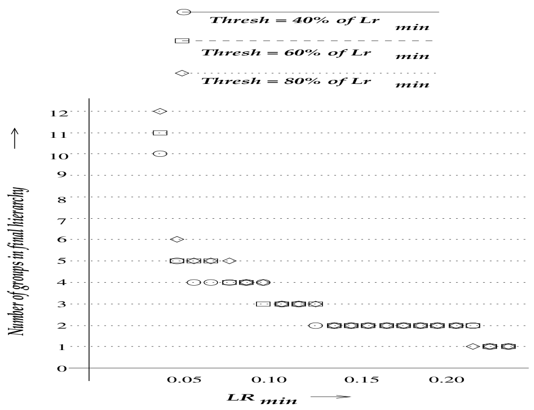

Figure 6. Effect of LR_{min} on number of groups in the final hierarchy

A GFP participant computes the value of $p_s(i, g)$ afresh for every incoming lossprint and makes a group initiation/join decision based on this value. A participant's decision causes the hierarchy to evolve by a single level, i.e. the information conveyed by a single lossprint controls the evolution of the hierarchy by a single layer. It is thus important to understand how much loss information a receiver must acquire and exchange in order to make useful grouping decisions as this impacts the protocol's rate of convergence to a final stable hierarchy.

Figure 4 plots the probability with which the hierarchy built by GFP correctly matches the underlying tree for different lossprint lengths for network topologies of different sizes. The computed hierarchy is said to be *correct* if for every group G in the final hierarchy, the closest common ancestor of all the group G members has only members of G as downstream receivers. As expected, GFP performance improves with increasing lossprint lengths because more information is utilized per grouping decision. As GFP serves as a protocol optimization rather than being critical for correct protocol operation we need not always achieve perfect results and a trade-off can be made between the desired degree of accuracy and the amount of bandwidth allocated to traffic due to *INIT* messages.

The ability to extract topological information depends on the distribution of link loss rates along the paths from the source to the different receivers. A typical transmission might originate from a low loss rate environment such as a LAN, traverse a few shared wide-area, lossy links (eg: trans-atlantic, trans-continental links) and be ultimately delivered to end receivers on well-connected LANs. Intuitively, GFP should perform well in such a setting on account of the significant shared loss along the wide area path. At the same time, GFP should still perform satisfactorily in a worst-case scenario where the range of link loss rates is not clearly demarcated between the shared and non-shared path. Figure 5 illustrates GFP performance for different ranges of link loss rates for the three different types of links generated in TIERS topologies: wide area (WAN) links, local area (LAN) links and intermediate MAN links. The data indicates that the best performance is obtained for cases where the difference in loss rates between different link types is the most pronounced and performance degrades as this difference is reduced thus corroborating our intuition.

4.2 Effect of protocol parameters

The two key parameters that control the operation of GFP are the minimum working loss rate threshold LR_{min} and *Thresh* used in making group initiation / join decisions. LR_{min} controls the degree of heterogeneity with local data groups and *Thresh*, the shape of the hierarchy.

The value of LR_{min} controls the permissible degree of heterogeneity within a single data group. If there exists a path within a data group with loss rate greater than LR_{min} then a receiver downstream from that path initiates the formation of a new group. The hierarchy settles when the loss rates of receivers within any single group is less than LR_{min}. For a given tree topology, LR_{min} also influences the number of members in each group and the total number of data groups in the final hierarchy. A high value of LR_{min} results in fewer data groups with a larger more heterogeneous member set while a lower value results in a larger number of smaller more homogeneous groups. Fig 6 illustrates this trade-off. For a topology with 1 WAN, 4 MANs and 8 LANs (with a total of 40 nodes) the graph plots the

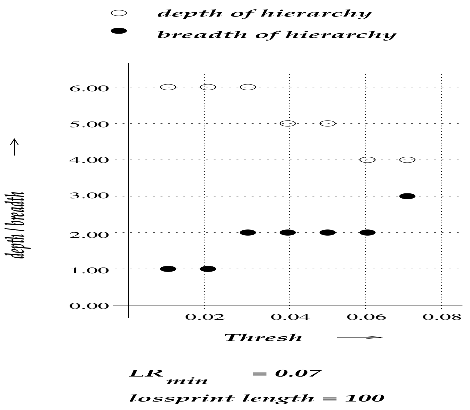

Figure 7. Effect of $Thresh$ on the shape of the final hierarchy

number of groups (including the original data group) in the final stable hierarchy for increasing values of LR_{min}. Link loss rates are set to 4%, 6% and 10% for LAN, MAN and WAN links respectively. As expected, the number of groups decreases with increasing values of LR_{min}. *Thresh* controls the shape of the final hierarchy. A high value of *Thresh* yields a hierarchy that is more "broad" than "deep" while a lower value of *Thresh* will yield one that is more "deep" than "broad". Figure 7 plots simulation results obtained by running GFP on a binary tree topology of depth seven with a single LAN at every level of the tree. All the link loss rates were set to 5%. The graph plots the depth and breadth in the final hierarchy for increasing values of *Thresh*. The depth of the hierarchy is the number of levels in the final stable state GFP hierarchy. The breadth of the hierarchy is the maximum number of data groups at any given level in the final hierarchy.

5 GFP as a reusable component

Several research multicast protocols and applications rely on the identification of sets of co-located receivers to enhance loss recovery, congestion control and so forth. In this section, using a few existing applications and protocols as examples, we describe how GFP serves as an important reusable building block across a wide range of these proposed schemes. While we haven't implemented the proposed schemes or worked out the finer details, we believe the basic approach should work well in practice.

Loss recovery algorithms for reliable multicast has proven difficult to scale to a large number of receivers. For example, the global loss recovery component of SRM multicasts retransmission requests and replies to the entire group and thus scales problematically. To solve this problem a number of schemes attempt to achieve *local recovery*. The key idea behind local recovery is to identify loss neighborhoods of receivers that share similar loss patterns and confine error recovery to this neighbourhood without disturbing the rest of the tree.

Liu et al. [16] propose the the use of separate local multicast groups in order to confine the scope of error recovery traffic in SRM. Multiple local groups are associated with a source where each group is responsible for error recovery of one or more lossy links. For a specific source, lossy links are related to one another as either ancestors/descendant or siblings and hence membership of the local groups should ideally be either perfectly disjoint or nested. Shared loss thresholding is used to form local recovery groups. The GFP data groups are essentially the same as Liu's local recovery groups and avoid the shared loss pathology incurred by their use of shared loss thresholding. The desired nested effect can be achieved using GFP if receivers incrementally join new proposed data group without leaving their current ones.

The Reliable Multicast Transport Protocol [15] addresses the loss recovery problem by organizing members into a hierarchy. Internal nodes in the hierarchy called Designated Receivers (DRs) cache data packets for later retransmission. Receivers are grouped into local regions each of which is assigned a DR. Acknowledgments are sent not to the source, but to the DR in th region. RMTP therefore provides both implosion avoidance and local recovery. However for RMTP to perform well, the hierarchy of members must be very closely correlated to the underlying multicast distribution tree and DRs need to be optimally distributed over the tree. The dynamic configuration of receivers and DRs is outlined as future work. The integration of GFP with RMTP enables this dynamic configuration of receivers into the required hierarchy with GFP representatives playing the role of DRs.

A number of emerging research proposals advocate the use of application aware agents or *proxies* to serve as intermediaries between a source and receivers as a means of coping with the heterogeneity inherent in multicast sessions on account of varied network and end host capabilities.

In [3] Chawathe et al. present a general architecture for proxy-based reliable multicast called the Reliable Multicast proXy (RMX) model. Their prototype RMX implementation for a shared whiteboard application for hand-held PDAs relies on the existence of a well-known service cluster that supports the RMX. Algorithms for the dynamic placement of RMX agents is outlined as future work.

Self-Organized Transcoding (SOT) [11] adapts continuous-media applications to varying network conditions by the self organization of groups of receivers with bad reception and provides rate adjustment through the use of transcoders. In SOT, co-located receivers experiencing loss caused by a bottleneck link, elect a representative which will locate an upstream receiver with better reception at the far end of the bottleneck to act as transcoder. The transcoding receiver multicasts a customized version of the stream to a new address and receivers adversely affected by the bottleneck switch to the new group. SOT uses shared loss as the metric for group identification.

Handley [8] proposed a relay-based congestion control architecture for bulk data transfer in which a high loss rate receiver elects itself to act as representative through the use of SRM-style random timers weighted by the number of losses experienced. Representatives initiate the formation of new subgroups and receivers use shared loss in deciding whether to join a proposed subgroup. The elected representative performs an expanding ring search in order to locate an upstream receiver with better reception to act as relay for the new subgroup.

In SOT, RMX, Relay-based congestion control, and in fact, any agent-based scheme, protocol stability requires that *all* receivers within the same loss subtree are co-ordinated in their actions such as switching groups, tuning in to the same proxy/transcoder etc. Further, enlisting such agents raises the problem of placing them intelligently and dynamically throughout the network. GFP enables solutions to both problems: co-ordination is achieved by building a hierarchy of data groups that explicitly corresponds to the different loss subtrees and strategically located representatives enable the intelligent placement of transcoders and proxies.

A multicast-based adaptive web caching infrastructure has been proposed [10, 19] to meet the exponential growth of the Web. Zhang et al.[19] outline an adaptive web caching system in which Web servers and cache servers self organize into overlapping local multicast groups. A request for a web page is successively forwarded through a chain of overlapping cache groups until it reaches a cache group with the required page. GFP yields the desired structure with the group representatives serving as ideal points of overlap. GFP can achieve overlapping hierarchies if group representatives retain their membership in the parent group even after the formation of new lower level groups.

6 Related Work

Several schemes in the research literature rely on the aggregation of receivers in a multicast session for scalability. In this section we compare the underlying aggregation metric / technique used by these schemes with GFP. As group formation is an enabling piece and not the principal goal of much of the work described here, none of these schemes have, to our knowledge, evaluated the accuracy of the group formation process in terms of group structure conforming to the underlying routing structure.

In [8, 11, 16, 19] groups are formed based on the measured *net* shared loss between receivers. Such schemes suffer from the shared loss pathology described in Section 2 overcome by our use of real shared loss probabilities. A number of schemes [26, 23, 12, 9] rely on the use of *time-to-live* or TTL-based scope to limit the reach of traffic. Real-world measurements [26] however indicate that TTL is not a good measure of locality as the number of reachable hosts does not increase linearly with TTL values. Further, TTL count by itself does not provide any information regarding receiver losses nor does it indicate where the bottleneck links lie relative to the different receivers. It might however be possible to combine TTL scoping and loss based approaches such as GFP for enhanced robustness and scalability.

The Tracer protocol [13] uses the MTRACE router function to organize the receivers of a multicast group deterministically into a logical tree structure. The existing implementation of MTRACE could lead to scaling problems due to an implosion of MTRACE queries at the source. Further, this places a heavy load on the source which has to unicast replies back to every receiver. In order to improve the efficiency of tracing in Tracer, Levine et al propose the addition of source-based multicast tracing to IGMP.

Static configuration of receivers as in RMTP [15] , the use of service clusters with well known locations [1, 3] for agent-based solutions etc while good first-cut solutions may not scale well to large receiver groups with multiple service clusters without the dynamic and optimal configuration of receivers.

In [2] the authors explore the use of end-to-end multicast traffic as measurement probes to infer network-internal characteristics and develop an MLE for individual link loss rates based on the end-to-end losses observed by multicast receivers. Their work assumes a priori knowledge of the logical topology of the multicast routing tree and is thus orthogonal and complementary to GFP.

Finally, recent research proposals [20, 14] advocate the inclusion of new forwarding services within the network in order to better support end-to-end transports. These schemes enable the acquisition of topological information at the cost of modifying the existing IP service model and are an alternative approach to our end-to-end solution. It is worthwhile to explore both approaches to better understand the trade-offs involved.

7 Future Work

In future work, we plan to work on reliable multicast solutions that are based on GFP and a comprehensive application of GFP to existing protocols and applications. This would enable us to test GFP in an actual Internet setting and evaluate the performance of

GFP in the face of correlated loss, representative crashes, dynamic link characteristics, fluctuating group membership and other such real-world events. Our assumption of a uniform loss distribution could prove problematic under certain circumstances. In future work, we plan to study the validity and effect of this assumption.

8 Conclusions

In this paper, we studied the fundamental feasibility of a *divide and conquer* style approach to scalable multicast transports. We designed and simulated a practical protocol primitive, namely a Group Formation Protocol whereby receivers self organize into a multi-level hierarchy of topologically localized multicast subgroups. GFP works with the existing IP service model and preserves the forwarding semantics of the current Internet architecture. We believe that GFP can serve as a reusable core protocol building block across a wide range of end-to-end multicast applications and protocols proposed in the research literature.

9 Acknowledgments

This work greatly benefited from discussions with Sally Floyd, Vern Paxson, Nick Duffield, Steve Deering, Francesco LoPresti and members of the MINC research group and received useful feedback from Anthony Joseph and the ICNP reviewers.

References

[1] AMIR, E., MCCANNE, S., AND KATZ, R. H. An active service framework and its application to real-time multimedia transcoding. In *Proceedings of SIGCOMM '98* (Vancouver, BC CANADA, Sept. 1998).

[2] CACERES, R., DUFFIELD, N., HOROWITZ, J., TOWSLEY, D., AND BU, T. Multicast Based Inference of Network-Internal characteristics: Accuracy of Packet Loss Estimation. In *Proceedings IEEE Infocom '99* (New York, NY, Mar. 1999).

[3] CHAWATHE, Y., FINK, S., MCCANNE, S., AND BREWER, E. A Proxy Architecture for Reliable Multicast in Heterogeneous Environments. In *Proceedings of ACM Multimedia '98* (Sept. 1998).

[4] CLARK, D. D. The design philosophy of the DARPA Internet protocols. In *Proceedings of SIGCOMM '88* (Stanford, CA, Aug. 1988), ACM.

[5] DEERING, S., AND CHERITON, D. Multicast routing in datagram internetworks and extended LANs. *ACM Transactions on Computer Systems 8*, 2 (May 1990), 85–110.

[6] DOAR, M. A better model for generating test networks. In *Proceedings of GLOBECOM '96* (London, UK, Nov. 1996).

[7] FLOYD, S., JACOBSON, V., MCCANNE, S., LIU, C.-G., AND ZHANG, L. A reliable multicast framework for light-weight sessions and application level framing. In *Proceedings of SIGCOMM '95* (Boston, MA, Sept. 1995), ACM, pp. 342–356.

[8] HANDLEY, M. J. A congestion control architecture for bulk data transfer, Sept. 1997. Presentation at the Reliable Multicast Research Group meeting.

[9] HOFMANN, M. Enabling group communication in global networks. In *Proceedings of Global Networking* (Alberta, Canada, June 1997).

[10] JACOBSON, V. SIGCOMM '95 Middleware Workshop: How to kill the Internet, Aug. 1995.

[11] KOUVELAS, I., HARDMAN, V., AND CROWCROFT, J. Network adaptive continuous-media applications through self organised transcoding. In *Proceedings of the Network and Operating Systems Support for Digital Audio and Video* (Cambridge, U.K., July 1998).

[12] LEVINE, B. N., LAVO, D. B., AND GARCIA-LUNA-ACEVES, J. The case for reliable concurrent multicasting using shared ACK trees. In *Proceedings of ACM Multimedia '96* (Boston, MA, Nov. 1996), ACM.

[13] LEVINE, B. N., PAUL, S., AND GARCIA-LUNA-ACEVES, J. Organizing multicast receivers deterministically by packet-loss correlation. In *Proceedings of ACM Multimedia '98* (Bristol, UK, Sept. 1998), ACM.

[14] LI, D., AND CHERITON, D. R. OTERS (On-Tree Efficient Recovery using Subcasting): A reliable multicast protocol. In *Proceedings of the Sixth IEEE International Conference on Network Protocols* (Austin, Texas, Oct. 1998), pp. 237–245.

[15] LIN, J. C., AND PAUL, S. RMTP: A Reliable Multicast Transport Protocol. In *Proceedings IEEE Infocom '96* (San Francisco, CA, Mar. 1996), pp. 1414–1424.

[16] LIU, C.-G., ESTRIN, D., SHENKER, S., AND ZHANG, L. Local error recovery in SRM: Comparison of two approaches. *IEEE/ACM Transactions on Networking 6* (Dec. 1998), 686–699.

[17] MCCANNE, S., AND FLOYD, S. *The LBNL/UCB Network Simulator*. Lawrence Berkeley Laboratory, University of California, Berkeley.

[18] MCCANNE, S., JACOBSON, V., AND VETTERLI, M. Receiver-driven layered multicast. In *Proceedings of SIGCOMM '96* (Stanford, CA, Aug. 1996), ACM, pp. 117–130.

[19] MICHEL, S., NGUYEN, K., ROSENSTEIN, A., ZHANG, L., FLOYD, S., AND JACOBSON, V. Adaptive web caching: towards a new global caching architecture. In *Computer Networks and ISDN Systems* (Nov. 1998).

[20] PAPADOPOULOS, C., PARULKAR, G., AND VARGHESE, G. An error control scheme for large-scale multicast applications. In *Proceedings IEEE Infocom '98* (San Francisco, CA, Mar. 1998).

[21] RAMAN, S., AND MCCANNE, S. A Model, Analysis, and Protocol Framework for Soft State-based Communication. In *Proceedings of SIGCOMM '99* (Cambridge, MA, Sept. 1999), ACM.

[22] RATNASAMY, S., AND MCCANNE, S. Inference of multicast routing trees and bottleneck bandwidths using end-to-end measurements. In *Proceedings IEEE Infocom '99* (New York, NY, Mar. 1999).

[23] ROSENSTEIN, A., LI, J., AND TONG, S. Y. MASH: The Multicasting Archie Server Hierarchy. In *Computer Communication Review* (July 1997), ACM.

[24] SALTZER, J. H., REED, D. P., AND CLARK, D. D. End-to-end arguments in system design. *ACM Transactions on Computer Systems 2*, 4 (Nov. 1984).

[25] SCHULZRINNE, H., CASNER, S., FREDERICK, R., AND JACOBSON, V. *RTP: A Transport Protocol for Real-Time Applications*. Internet Engineering Task Force, Audio-Video Transport Working Group, Jan. 1996. RFC-1889.

[26] XU, X. R., MYERS, A. C., ZHANG, H., AND YAVATKAR, R. Reliable multicast support for continuous-media applications. In *Proceedings of the Seventh International Workshop on Network and OS Support for Digital Audio and Video* (St. Louis, MO, May 1997), ACM.

[27] YAVATKAR, R., GRIFFIOEN, J., AND SUDAN, M. A reliable dissemination protocol for interactive collaborative applications. In *Proceedings of ACM Multimedia '95* (San Francisco, CA, Nov. 1995), ACM.