

A Framework for Architecting Peer-to-Peer Receiver-driven Overlays

Reza Rejaie
Department of Computer Science
University of Oregon
reza@cs.uoregon.edu

Shad Stafford
Department of Computer Science
University of Oregon
staffors@cs.uoregon.edu

ABSTRACT

This paper presents a simple and scalable framework for architecting peer-to-peer overlays called *Peer-to-peer Receiver-driven Overlay* (or *PRO*). *PRO* is designed for non-interactive streaming applications and its primary design goal is to maximize delivered bandwidth (and thus delivered quality) to peers with heterogeneous and asymmetric bandwidth. To achieve this goal, *PRO* adopts a receiver-driven approach where each receiver (or participating peer) (*i*) independently discovers other peers in the overlay through gossiping, and (*ii*) selfishly determines the best subset of parent peers through which to connect to the overlay to maximize its own delivered bandwidth. Participating peers form an unstructured overlay which is inherently robust to high churn rate. Furthermore, each receiver leverages congestion controlled bandwidth from its parents as implicit signal to detect and react to long-term changes in network or overlay condition without any explicit coordination with other participating peers. Independent parent selection by individual peers dynamically converge to an efficient overlay structure.

Categories and Subject Descriptors: C.2.4 [Computer-Communication Networks]: Distributed Systems

General Terms: Design, Measurement

Keywords: Peer-to-Peer Streaming, Congestion Control

1. INTRODUCTION

Limited deployment of IP multicast has motivated a new distribution paradigm over the Internet based on *overlay networks* where a group of participating end-systems (or *peers*) form an overlay structure and actively participate in distribution of content without any special support from the network (*e.g.*, [7]). Since overlay structures are layered over the best-effort Internet, any approach for constructing overlay should address the following fundamental challenges: (*i*) Scalability with the number of participating peers, (*ii*) Robustness to dynamics of peer participation, (*iii*) Adaptation to variations of network bandwidth, and (*iv*) Accommodat-

ing heterogeneity and asymmetry of bandwidth connectivity among participating peers[19]. Coping with bandwidth variations, heterogeneity and asymmetry are particularly important in design of peer-to-peer overlay for streaming applications because delivered quality to each peer is directly determined by its bandwidth connectivity to (other peer(s) on) the overlay.

This paper presents a simple framework for architecting Peer-to-peer Receiver-driven Overlay, called *PRO*. *PRO* can accommodate a spectrum of non-interactive streaming applications ranging from playback to lecture-mode live sessions. The main design philosophy in *PRO* is that each peer should be allowed to independently and selfishly determine the best way to connect to the overlay in order to maximize its own delivered quality. Toward this end, each peer can connect to the overlay topology at multiple points (*i.e.*, receive content through multiple *parent* peers). Therefore, participating peers form an unstructured overlay that can gracefully cope with high churn rate[5]. Furthermore, having multiple parent peers accommodates bandwidth heterogeneity and asymmetry while improves resiliency against dynamics of peer participation.

PRO consists of two key components: (*i*) *Gossip-based Peer Discovery*: Each peer periodically exchanges message (*i.e.*, gossips) with other known peers to progressively *learn* about a subset of participating peers in the overlay that are likely to be good parents. Gossiping provides a *scalable* and *efficient* approach to peer discovery in unstructured peer-to-peer networks that can be customized to guide direction of discovery towards peers with desired properties (*e.g.*, peers with shorter distance or higher bandwidth). (*ii*) *Receiver-driven Parent Selection*: Given the collected information about other participating peers by gossiping mechanism, each peer (or receiver) gradually improves its own delivered quality by dynamically selecting a proper subset of parent peers that collectively maximize provided bandwidth to the receiver. Since the available bandwidth from different participating peers to a receiver (and possible correlation among them) can be measured *only* at that receiver, a receiver-driven approach is the natural solution to maximize available bandwidth to heterogeneous peers. Furthermore, the available bandwidth from parent peers serves as an implicit signal for a receiver to detect and react to changes in network or overlay condition without any explicit coordination with other participating peers. Independent parent selection by individual peers leads to an efficient overlay that maximizes delivered quality to each peer. *PRO* incorporates

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

NOSSDAV'04, June 18–18, 2004, Cork, Ireland.

Copyright 2004 ACM 1-58113-801-6/04/0006 ...\$5.00.

several damping functions to ensure stability of the overlay despite uncoordinated actions by different peers.

PRO is part of a larger architecture that we have developed for peer-to-peer streaming. In our earlier work, we developed a mechanism called *PALS* [18] that enables a receiver to stream layered structured content from a given set of congestion controlled senders. Thus, *PRO* and *PALS* are both receiver-driven but complement each other. More specifically, *PRO* determines a proper subset of parent peers that collectively maximize delivered bandwidth to each receiver whereas *PALS* coordinates “in-time” streaming of different segments of multimedia content from these parents despite unpredictable variations in their available bandwidth. This division of functionality provides a great deal of flexibility because it decouples overlay construction from delivery mechanism. In this paper, we primarily focus on the overlay construction mechanism, or *PRO*.

The rest of this paper is organized as follows: In Section 2, we revisit the problem of overlay construction for peer-to-peer streaming and identify its two key components and explore their design space. We illustrate the differences between *PRO* and previous solutions, and justify our design choices. We present our proposed framework in Section 3. In Sections 4 and 5, the key components of our framework are described in further detail. Finally, Section 6 concludes the paper and presents our future plans.

2. REVISITING THE PROBLEM

Constructing a peer-to-peer overlay for streaming applications should not only accommodate *global* design goals such as scalability and resilience but also satisfy the *local* design goal of maximizing delivered quality to individual peers¹. More specifically, delivered quality of streaming content to each peer should be proportional to its incoming access link bandwidth. Achieving these goals is particularly challenging because participating peers often exhibit heterogeneity and asymmetry in their bandwidth connectivity.

Solutions for constructing peer-to-peer overlays often require two key mechanisms to be implemented at each peer: *Peer Discovery (PD)* and *Parent Selection (PS)*. The PD mechanism enables each peer to learn about other participating peers in the overlay. Information about other peers are used by the PS mechanism at each peer to determine proper parent peers through which it should connect to the overlay. The collective behavior of PD and PS mechanisms at all participating peers leads to an overlay structure that achieves the above design goals. There has been a wealth of previous research that explored design space of the PD and PS mechanisms as follows:

Peer Discovery: In structured peer-to-peer networks, the existing structure enables each peer to find other participating peers in a scalable fashion (*e.g.*, [4]). However, structured peer-to-peer networks may not be robust against high churn rate [5]. In contrast, unstructured peer-to-peer networks can gracefully accommodate high churn rate [5] but require a separate peer discovery mechanism. Mesh-first approaches (*e.g.*, [7, 6]) that require each peer to know about all other participating peers as well as centralized approaches (*e.g.*, [16]) to peer discovery exhibit limited scalability. NICE [2] leverages a hierarchal structure to achieve

¹It is worth clarifying that our design goal is different from common goals in building application-level multicast trees [7] (*i.e.*, minimizing stretch and stress).

scalability but each peer only knows about a group of close-by peers who may not be good parents (*i.e.*, may not provide sufficient bandwidth).

Parent Selection: We examine two key aspects of parent selections:

(i) *Selection Criteria:* There are two main criteria for parent selections: relative delay and available bandwidth between two peers. Relative delay between any two peers can be estimated in a scalable fashion with one of the existing landmark-based solutions such as Global Network Positioning (GNP) [15]. However, estimating available bandwidth between two peers requires end-to-end measurement. Using available bandwidth as criteria for parent selection does not scale for two reasons: First, to cope with dynamics of bandwidth variations, each peer requires to periodically estimate the available bandwidth from all other peers through measurement (*e.g.*, [6]). Second, the probability of interference among different measurements grows with the number of peers in an overlay (similar to joint experiment in RLM [13]).

Most of the previous solutions adopted the idea of application level multicast and used delay as the main selection criteria. Participating peers cooperatively run a distributed algorithm to organize themselves into a source-rooted tree structure in order to minimize either overall delay across all branches of the tree (*e.g.*, [7]), or delay between source and each receiver peer (*e.g.*, [20]). While these parent selection strategies minimize associated network load, they may not provide sufficient bandwidth to individual peers because delay is often not a good indicator for available bandwidth between two peers [12, 14]. The key issue is that minimizing overall delay (global design goal) and maximizing delivered bandwidth to each peer (local design goal) could easily be in conflict. More specifically, parent peers with longer relative distance may provide higher bandwidth than close-by parents. This suggests that there might exist a tradeoff between maximizing provided bandwidth to each peer and minimizing overall delay across the overlay.

(ii) *Single vs Multiple Parents:* A single tree structure for the overlay (where each peer has a single parent) is inherently unable to accommodate peers with heterogeneous and asymmetric bandwidth. A common approach to accommodating bandwidth heterogeneity is to use layer structured content (either layered or multiple description encodings) and allow each receiver to have multiple parents. This approach could accommodate heterogeneity but it introduces several new challenges. First, parent selection strategy should be determined based on location of a bottleneck. If the bottleneck is at the (outgoing) access links of parent peers², then a receiver should simply look for more parents. However, when the bottleneck is else where in the network, a receiver should select parents with a diverse set of paths (*i.e.*, utilize different network paths). In practice, a combination of these cases might simultaneously exist among participating peers [1]. Second, streaming a single content from multiple senders is challenging for two reasons: 1) This requires tight coordination among senders to determine overall delivered quality (*e.g.*, number of layers) and decide which sender is responsible for delivery of each segment. 2) Delivered segments from different senders should arrive before their playout times despite uncorrelated vari-

²if bottleneck is at the receiver’s access link, then provided bandwidth to the receiver is already maximized.

ations in (congestion controlled) bandwidth from different senders. This also implies that those solutions that build multi-parent overlay structure but do not explicitly ensure in-time delivery of individual segments (e.g., [3, 11]) may not be able to support streaming applications.

One approach to build a multi-parent overlay is to organize participating peers into different trees where each layer of the stream is sent to a separate tree (e.g., [4, 16]). Each peer can maximize its quality by participating in a proper number of trees. This approach raises several issues: 1) the provided bandwidth to peers in each tree is limited by minimum uplink bandwidth among upstream peers on that tree. In the presence of bandwidth asymmetry, this could easily limit delivered bandwidth on each tree below the required bandwidth for a single layer, 2) it is not feasible to build separate trees that are all optimal for a single selection criteria (e.g., overall delay), 3) connections across different trees are likely to compete for available bandwidth on a single bottleneck³. We conclude that a practical solution for peer-to-peer streaming applications should incorporate the following design properties: (i) it should use an unstructured, multi-parent peer-to-peer overlay, (ii) it should provide a scalable peer discovery mechanism that enables each peer to find its good parents efficiently, (iii) it should detect (and possibly avoid) any shared bottleneck among different connections in the overlay, and (iv) it should deploy congestion controlled connections but ensure in-time arrival of delivered segments to each receiver. In the next section, we explain how PRO incorporates all the above design properties.

3. P2P RECEIVER-DRIVEN OVERLAY

Assumptions: We assume that each peer can estimate the relative distance between any two peers using the GNP mechanism [15]. Furthermore, each peer knows the incoming and outgoing bandwidth of its access link. Each peer uses the PALS mechanism to stream content from multiple parent peers. All connections are congestion controlled by senders (e.g., [17]). To accommodate peer bandwidth heterogeneity, we assume that the content has a layered representation. In other words, with proper adjustment, the framework should work with both layered and multiple-description encodings. Participating peers have heterogeneous and asymmetric bandwidth connectivity. Furthermore, peers may join and leave in an arbitrary fashion. **Overview:** In PRO, each peer (or receiver) progressively searches for a subset of parents that collectively maximize delivered bandwidth and minimize overall delay from all parents to the receiver. Such a subset of parents may change over time as some parents join (or leave) the overlay, or available bandwidth from current parents significantly changes. Note that each peer can be both receiver and parent at the same time⁴. Each receiver periodically exchanges messages (i.e., gossips) with other peers in the overlay to learn about those participating peers that are potentially good parents. Potentially good parents for a receiver are identified based on their relative utility for the receiver. The utility of a parent peer p_i for a receiver p_j is a function of their relative network distance (del_{ij}), and the outgoing access link bandwidth of the parent ($outbw_i$), (i.e., $U(p_i, p_j)$

³These multi-tree approaches often do not use congestion control for each connection.

⁴Throughout this paper we use “receiver” and “parent” as short form for “receiver peer” and “parent peer”.

$= f(del_{ij}, outbw_i)$). Using parents’ access link bandwidth instead of available bandwidth has several advantages: (i) outgoing bandwidth is an upper bound for available bandwidth from a parent. Therefore, it enables the receiver to roughly classify different parents. (ii) estimating available bandwidth requires end-to-end measurement and such a solution does not scale with the number of peers, and more importantly, (iii) given a utility function, this approach enables any peer in the overlay to estimate relative utility of any other two peers. Each receiver only maintains information about a fixed (and relatively small) number of promising parent peers in its *local image*. The local image at each receiver is dynamically updated with new gossip messages as other peers join/leave the overlay. Each peer selects a new parent in a demand-driven fashion in order to minimize the number of end-to-end bandwidth measurements, and thus improve scalability. When a receiver needs a new parent, its PS mechanism *randomly* selects a peer from its local image where probability of selecting a peer directly depends on its utility. Then, the actual properties (i.e., available bandwidth and delay) of the selected parent are verified through passive measurement. Toward this end, the selected parent is added to the parent list which triggers PALS to request content from this parent. Figure 1 depicts the interactions between PD and PS mechanisms.

In PRO, each receiver leverages congestion controlled bandwidth from its parents as an implicit signal to detect two events: (i) any measurable shared bottleneck among connections from different parents, and (ii) any change in network or overlay conditions (e.g., departure or arrival of other close-by peers). Figure 2 shows part of an overlay to illustrate this feature. Each receiver continuously monitors available bandwidth from all its parents. Receiver p_0 initially has only p_1 as a parent. When p_0 adds a new parent (p_2), the receiver examines the available bandwidth from p_0 and p_1 and any measurable correlation between them. If the available bandwidth from p_0 decreases after p_1 is added, the receiver can conclude that these two parents are behind the same bottleneck (i.e., link L_0). We note that paths from two parents might have some overlap that does not include any bottleneck. Assume another receiver p_3 selects p_1 as a parent and thus competes with receiver p_0 for available bandwidth on link L_1 . Suppose that L_1 becomes a bottleneck and the connection between p_1 to p_3 obtains a significantly higher share of L_1 ’s bandwidth than connection between p_1 to p_0 . This change in available bandwidth from p_1 serves as a signal for p_0 . Whenever a receiver detects such a drop in bandwidth, it waits for a random period of time (proportional to the available bandwidth) and then drops

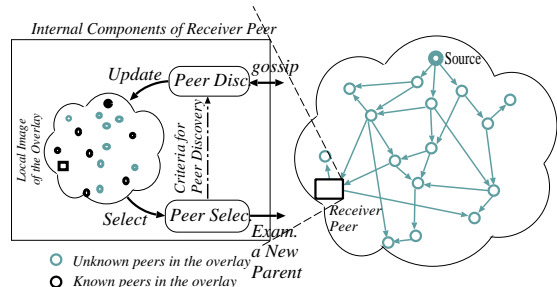


Figure 1: Interactions between PD and PS mechanisms through local image

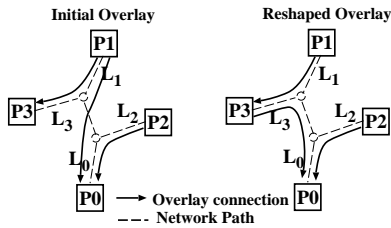


Figure 2: Using congestion controlled bandwidth as signal to reshape the overlay

the corresponding parent if its bandwidth remains low [8]. Therefore, the receiver with a higher bandwidth connectivity (p_3) is more likely to keep p_1 as parent whereas p_0 may examine other parents with higher bandwidth including p_3 . The congestion controlled bandwidth signals the receiver to properly reshape the overlay. We present a summary of key features and limitations of *PRO* in the next two sections. Table 1 summarizes our notation throughout this paper.

Main Features: Gossiping provides a scalable approach to peer discovery because each peer does not require global knowledge about all group members, and its generated traffic can be controlled. The PD mechanism actively participates in peer selection by identifying peers for the local image which limits the possible choices of parents for the PS mechanism. *PRO* constructs a multi-parent, unstructured overlay. But *PRO* does not have the same limitations that exist in multi-tree approaches because it allows each receiver to independently micro-manage its parents to maximize its overall bandwidth based on local information. *PRO* conducts passive measurement not only to determine available bandwidth from a parent but also to detect any shared bottleneck between paths from different parents. Furthermore, by selecting a new parent from the local image, *PRO* increases the probability of finding a good parent in each selection, and thus significantly decreases number of required measurements which in turn improves scalability. *PRO* can gracefully accommodate bandwidth heterogeneity and asymmetry among peers since *PALS* is able to manage delivery of content from a group of parents with different bandwidth.

Limitations and Challenges: The main hypothesis in our framework is that the best subset of parents for each receiver are likely to be part of its local image *i.e.*, PD mechanism can find the best parents. Whenever this condition is not satisfied, either a receiver may not be able to maximize its overall bandwidth or resulting overlay may not be efficient.

Table 1: Notation used throughout the paper

Symbol	Definition.
p_i	Peer i
$inbw_i$	Incoming access link BW for p_i
$outbw_i$	Outgoing access link BW for p_i
min_nop_i	Min. No of parents for p_i
max_nop_i	Max. No of parents for p_i
$nop_i(t)$	No of active parents for p_i at time t
img_sz	Size of local image at each peer
sgm	Size of gossip message
del_{ij}	Estimated delay between p_i and p_j

Clearly, properties of the selected utility function as well as accuracy of estimated parameters (in particular using outgoing bandwidth instead of available bandwidth) determine properties of the local image at each peer which in turn affects performance of the framework in some scenarios. In these cases, the utility value may not effectively guide the search process in identifying good parents which increases the average convergence time until each peer finds a good subset of parents. Similar to many other adaptive mechanisms (*e.g.*, [13]), the parent selection mechanism should address the fundamental tradeoff between responsiveness and stability. Finally, the congestion controlled bandwidth from parent peers may not provide a measurable signal to detect a shared bottleneck when level of multiplexing is high at the bottleneck link. However, this is not a major limitation since the negative impact of a shared bottleneck in these cases is minimal. All the above limitations are in part due to the simplicity of our framework and would adversely affect its performance. However, we believe that this is a reasonable design tradeoff since simplicity is one of our key design goals. In the following sections, we describe the two key components of our framework in further details.

4. GOSSIP-BASED PEER DISCOVERY

Peer discovery at each receiver is basically a search among all participating peers in the overlay for a certain number (img_sz) of peers with the highest relative utility. *PRO* adopts a gossip-like [10] approach to peer discovery. Gossiping (or rumor spreading) has been frequently used as a scalable alternative to flooding that gradually spreads information among a group of peers. However, we use gossiping as a search mechanism [9] for finding promising parents since it has two appealing properties (*i*) the volume of exchanged messages can be controlled, and (*ii*) the gossip-based information exchange can be customized to leverage relative utility values to improve search efficiency.

The gossip mechanism works as follow: each peer maintains a local image that contains up to img_sz records where each record represents the following information for a previously discovered peer p_i in the overlay: 1) *IP address*, 2) *GNP coordinates*, 3) *number of received layers*, 4) *timestamp* when the record was last generated by a peer, 5) *outbw_i* and 6) *inbw_i*. To bootstrap the discovery process, a new receiver needs to learn about a handful of other participating peers in the overlay. This information can be obtained from the original server (or a well-known rendezvous point). The server should implement a strategy for selecting the initial peers that are provided to each new receiver. We call this the *initial parent selection* mechanism. Once the initial set of peers are known, each peer p_i periodically invokes a *target selection* mechanism to determine a target peer (p_j) from its local image for gossip. Given a utility function, peer p_i uses a *content selection* strategy to select sgm records (or smaller number when sgm records are not available) from its local image that are most useful for p_j and send those records to p_j . In response, p_j follows the same steps and replies with a gossip message that includes sgm records from its local image that are most useful for p_i , *i.e.*, bidirectional gossip. When a gossip message arrives at each peer, an *image maintenance* scheme integrates new records into the current local image and discards excess records such that certain property of the local image is improved (*e.g.*, increase overall utility of peers in the image) Aggregate performance of

a gossip mechanism can be presented by two average metrics and their distribution among peers: (i) *Average Convergence Time*: average number of gossip messages until all peers in an overlay reach their final images, and (ii) *Average Efficiency Ratio*: average ratio of unique records to the total number of received records by each peer.

We have been exploring the design space of four key components of the gossip mechanism. Frequency and size of gossip messages determine average freshness of local images. Currently, the server randomly selects the initial parents from its local image for each new peer.

Target Selection: Target selection randomly picks a peer from the current image to evenly obtain information from different areas of the overlay and speed up discovery.

Content Selection: peer p_k determines relative utility of all the peers (p_j) in its local image for target peer p_i , and then randomly selects sgm peers to prepare a gossip message for p_i . However, probability of selecting a peer directly depends on its utility. This approach is biased towards peers with higher utility but its randomness tend to reduce number of duplicate records in different gossip message from one peer (*i.e.*, improves efficiency). A potential drawback of this approach is the increase in convergence time. We plan to examine more efficient information sharing schemes such as bloom filters [3] in our future work. *PRO* uses joint-ranking [15] to determine relative utility of a parent for a receiver. Given a collection of peers in a local image of p_k , the joint-ranking scheme ranks all the peers once based on their outgoing bandwidth, and then based on their estimated delay from a target peer p_i . The utility of peer p_j ($U(p_j, p_i)$) is inversely proportional to the sum of p_j 's ranks in both rankings. Values for each property (*i.e.*, bandwidth and delay) of various peers are divided into multiple ranges (*i.e.*, bins) where all peers within each range are assumed to have the same value for that property. This "binning" scheme minimizes the sensitivity to minor differences in delay or bandwidth among different peers.

Image maintenance: Image maintenance mechanism evicts extra records (beyond img_sz) that satisfy one of the following conditions: (i) represent peers with the lower utility, (ii) represent peers that were already dropped by the PS mechanism due to poor performance and (iii) have a timestamp older than a threshold. This approach attempts to balance image quality (in terms of overall utility of existing peers) and its freshness.

Note that the gossip mechanism can discover any peer in the overlay as long as reachability is provided through overlap among local images at different peers. The higher the amount of overlap, the higher the efficiency of discovery, and the higher the robustness of the overlay to dynamics of peer participations. The amount of overlap among images depends on both the size and shape of the local images at each peer. The shape of the local image is a function of the deployed utility function. Joint-ranking utility gives the same weight to delay and bandwidth. Delay tends to bias selection towards near-by peers whereas outgoing bandwidth introduces some degree of randomness in location of selected peers. Therefore, the resulting local images should exhibit a sufficient degree of overlap.

5. PARENT SELECTION

The PS mechanism at each peer is essentially a progressive search within the local image for a subset of parent

peers such that the following design goals are achieved: (i) maximizing delivered bandwidth⁵, (ii) minimizing the total delay from all parents to the receiver, and (iii) maximizing diversity of paths from parents (whenever it is feasible). Whenever these goals are in conflict, a receiver optimizes the goal with the highest priority. Currently, our framework does not directly consider diversity of paths from different parents as a criteria for parent selection. However, the indirect effect of shared path among parents is addressed because of its potential impact on available bandwidth from a parent when two or more parents are behind the same bottleneck.

The number of active parents ($nopi(t)$) for each receiver should be within a configured range [min_nop , max_nop]. Each receiver tries to maximize its delivered bandwidth with the minimum number of parents. If this goal can not be achieved after evaluation of a certain number of new parents, the receiver will gradually increase its number of parents. This flexibility is important in order to utilize available bandwidth from low bandwidth parents, *i.e.*, cope with bandwidth heterogeneity. min_nop determines minimum degree of resilience to parent departure, and minimum level of path diversity (whenever diverse paths are available). The number of children for each peer should not be limited. Instead, each peer only limits maximum outgoing bandwidth that it is able (or willing) to provide to its children. This allows child peers to compete for congestion controlled bandwidth from a parent which motivates child peers with poor bandwidth connectivity to look for other parents (*i.e.*, properly reshape the overlay).

Design of a PS mechanism should address three main questions as follows:

1) When should a new parent be selected?

There is a fundamental tradeoff between responsiveness of a receiver to changes in network conditions (or convergence time after a change) and stability of the overlay. *PRO* adopts a conservative approach where each peer selects a new parent in a *demand-driven* fashion. This should significantly reduce number of new parent selections, which in turn improves scalability (by minimizing the interference caused by new connections) and stability of the overlay structure. A new parent is selected in the following scenarios: (i) *Initial Phase*: when a new peer joins the overlay, it periodically adds a new parent until it has min_nop parents. (ii) *Replacing a Poorly-Performing Parent*: when available bandwidth from an existing parent is significantly reduced for a long time or a parent leaves the session, the receiver can select another peer after a random delay. Each receiver selects a random delay proportional to its available bandwidth from the parent peer [8]. This approach dampens potential oscillation in the overlay while increasing the chance for receivers with higher bandwidth connectivity to keep a parent (*i.e.*, properly reshapes the overlay). (iii) *Improvement in Performance*: when it is likely that a new parent would significantly improve a non-optimized aspect of performance (increase the bandwidth or decrease the delay). This strategy allows gradual improvement of the parent subset as new peers are discovered (or joined) the overlay. The available information for each peer in the image is used as a heuristic to predict performance of a new peer. Such an improvement should be examined infrequently. A hysteresis mechanism

⁵The target bandwidth is the lower value between maximum stream bandwidth and receiver's incoming bandwidth.

is implemented in scenario (ii) and (iii) to dampen any potential oscillation in the overlay.

2) Which peer should be selected as a new parent?

At any point of time, peers in the local image are the best known candidate peers to serve as parent. In *PRO*, each receiver randomly selects a parent from its current image where the probability of selecting a parent is proportional to its utility. Deploying this selection strategy by all peers lead to proportional utilization of outgoing bandwidth of all peers without making the selection heavily biased towards high bandwidth peers. This approach (similar to [5]) leverages heterogeneity among peers since number of children for each peer is proportional to its outgoing bandwidth.

3) How should a new parent be examined?

Each receiver continuously monitors available bandwidth from all parents and potential correlation between bandwidth of two or more connections as signal for shared bottleneck. The degree of such correlation also reveals the level of multiplexing at the bottleneck link, and could serve as an indicator for separating remote bottlenecks from a local one. Such a monitoring should use average bandwidth of each flow over a relatively long time scale (*e.g.*, hundreds of RTT) to filter out any transient variations in bandwidth. To avoid selecting a poorly-performing parent in the near future, the receiver associates a timer to each parent and exponentially backs off the timer after each failed experience [13].

After the initial phase, each receiver maintains a fixed number of parents at any point of time ($nop_i(t)$). Thus, a new parent should replace one of the active parents. However, to ensure monotonic improvement in overall performance of active parents, a new parent is always added before one of the existing parents is dropped (*i.e.*, a receiver can temporarily have one extra parent). Given the available bandwidth from all parents (including the new one) and possible correlation among them, a receiver can use one of the following criteria to drop a parent: (i) to maximize the bandwidth, the receiver can drop the parent that contributes minimum bandwidth, (ii) to maximize path diversity among connections from parents, the receiver should drop the parent that is located behind the same bottleneck with the largest number of active parents and contributes minimum bandwidth among them. Finally, if the aggregate bandwidth from all parents remains below the required bandwidth after examining certain number of new parents (and $nop_i(t) < max_nop$), the receiver can increase the total number of parents by one.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we presented a simple receiver-driven framework for architecting peer-to-peer overlay structures called *PRO*. *PRO* allows each peer to selfishly and independently determine the best way to connect to the overlay to maximize its performance. Therefore, *PRO* should be able to maximize delivered quality to peers with heterogeneous and asymmetric bandwidth connectivity. Both peer discovery and peer selection in this framework are scalable. Furthermore, *PRO* uses congestion controlled bandwidth as an implicit signal to detect shared bottleneck among existing parents as well as changes in network or overlay conditions to properly reshape the structure. We described the basic framework and its key components, and sketched our straw-man solutions.

This is a starting point for our work on *PRO*. We are currently evaluating various aspects of this framework via simulation, and exploring the design space of key components. We are also prototyping this framework to conduct real-world experiments on the Planet-Lab in a near future.

7. REFERENCES

- [1] A. Akella, S. Seshan, and A. Shaikh. An empirical evaluation of wide-area internet bottlenecks. In *Internet Measurement Conference*, 2003.
- [2] S. Banerjee, B. Bhattacharjee, and C. Kommareddy. Scalable application layer multicast. In *ACM SIGCOMM*, 2002.
- [3] J. Byers, J. Considine, M. Mitzenmacher, and S. Rost. Informed Content Delivery Across Adaptive Overlay Networks. In *ACM SIGCOMM*, 2002.
- [4] M. Castro, P. Druschel, A.-M. Kermarrec, A. R. A. Nandi, and A. Singh. SplitStream: High-bandwidth content distribution in a cooperative environment. In *ACM SOSP*, 2003.
- [5] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and S. Shenker. Making gnutella-like p2p systems scalable. In *ACM SIGCOMM*, 2003.
- [6] Y. Chu, S. G. Rao, S. Seshan, and H. Zhang. Enabling conferencing applications on the internet using an overlay multicast architecture. In *ACM SIGCOMM*, 2001.
- [7] Y. Chu, S. G. Rao, and H. Zhang. A case for end-system multicast. In *ACM SIGMETRICS*, 2000.
- [8] S. Floyd, V. Jacobson, C. Liu, S. McCanne, and L. Zhang. Reliable Multicast Framework for Light-Weight Sessions and Application Level Framing. *ACM/IEEE Transactions on Networking*, 1997.
- [9] M. Harchol-Balter, F. T. Leighton, and D. Lewin. Resource discovery in distributed networks. In *Symposium on Principles of Distributed Computing*, pages 229–237, 1999.
- [10] S. Hedetniemi, S. Hedetniemi, and A. Liestman. A Survey of Gossiping and Broadcasting in Communication Networks. In *Networks*, 1988.
- [11] D. Kotic, A. Rodriguez, J. Albrecht, and A. Vahdat. Bullet: High bandwidth data dissemination using an overlay mesh. In *SOSP*, 2003.
- [12] K. Lakshminarayanan and V. N. Padmanabhan. Some findings on the network performance of broadband hosts. In *Internet Measurement Conference*, 2003.
- [13] S. McCanne, V. Jacobson, and M. Vettereli. Receiver-driven layered multicast. In *ACM SIGCOMM*, 1996.
- [14] T. S. E. Ng, Y. Chu, S. G. Rao, K. Sripanidkulchai, and H. Zhang. Measurement-based optimization techniques for bandwidth-demanding peer-to-peer systems. In *IEEE INFOCOM*, 2003.
- [15] T. S. E. Ng and H. Zhang. Predicting internet network distance with coordinates-based approaches. In *IEEE INFOCOM*, 2002.
- [16] V. N. Padmanabhan, H. J. Wang, and P. A. Chou. Resilient peer-to-peer streaming. In *IEEE ICNP*, 2003.
- [17] R. Rejaie, M. Handley, and D. Estrin. RAP: An end-to-end rate-based congestion control mechanism for realtime streams in the internet. In *IEEE INFOCOM*, 1999.
- [18] R. Rejaie and A. Ortega. PALS: Peer-to-Peer Adaptive Layered Streaming. In *NOSSDAV*, 2003.
- [19] S. Saroiu, P. K. Gummadi, and S. D. Gribble. Measurement study of peer-to-peer file system sharing. In *SPIE MMCN*, 2002.
- [20] D. A. Tran, K. A. Hua, and T. Do. Zigzag: An efficient peer-to-peer scheme for media streaming. In *IEEE INFOCOM*, 2003.