Path-Diversity P2P Overlay Retransmission for Reliable IP-Multicast

Wenjun Zeng, *Senior Member, IEEE*, Yingnan Zhu, Haibin Lu, *Member, IEEE*, and Xinhua Zhuang, *Senior Member, IEEE*

***Abstract*—IP-multicast is a bandwidth efficient transmission mechanism for group communications. Reliability in IP-multicast, however, poses a set of significant challenges. To address the reliability and scalability issues in IP-multicast, this paper proposes a novel, highly distributed, and lightweight overlay peer-to-peer retransmission architecture that exploits path-diversity by taking advantages of both IP-multicast and an overlay network. An approach that leverages both disjoint-path-finding and periodic selective probing to take into account peer's recent packet loss probability, retransmission delay and recent retransmission success rate is proposed to effectively construct an efficient and dynamic overlay retransmission network. We show that the proposed path diversity overlay retransmission architecture has the potential to significantly reduce the retransmission delay, improve the reliability, playback quality, and scalability of IP-multicast based multimedia applications. Given a deployed IP-multicast network, the proposed overlay retransmission architecture is practical, scalable, and easy to deploy, requiring no change to the existing network infrastructure.**

***Index Terms*—IP-multicast, overlay, path-diversity, reliability, retransmission.**

I. Introduction

TRANSPORTING audio/video over the best-effort IP networks has been considered a cost-effective way to deploy many multimedia applications. IP-multicast (or native-multicast) is a bandwidth efficient transmission mechanism for applications where there are multiple participants. There have been significant deployments of IP-multicast in exchanges and securities trading companies, enterprises and college campuses, edge networks, and military networks [1]. For example, the Abilene Network in Internet-2 [2], a private network that interconnects most major universities in the U.S. for the education and research purpose, fully supports IP-multicast, and application platforms such as Microsoft Research's ConferenceXP platform [5] have been developed for such networks. Most Telecom IPTV networks have supported or will support multicast/broadcast delivery of TV programs. Despite all these progresses, we are yet to see "the *ubiquitous* deployment of a revenue generating native multicast infrastructure capable of securely and robustly supporting both reliable, TCP-friendly file transfer, all manner of streaming media, and any style of audio/video conferencing (with minimal jitter and end-to-end delay)—all with only minimal additional router complexity, deployment effort, management needs, or cost" [1]. In the meantime, application layer multicast (ALM) [6] that addresses the deployment and complexity issues at the cost of some performance loss (in terms of link stress, relative delay penalty, and instability due to the nodal join and leave activities) has been enjoying popularity. Nevertheless, research efforts on IP-multicast continue [7] and the concept has been revisited in both academia and industry [8]–[10]. Examples are the ongoing automatic multicast tunneling (AMT) work in IETF [9], and the recent initiative in IRTF on scalable adaptive multicast (SAM) [10] that aims to devise multicast framework that addresses limitations of native multicast and leverage both application layer and native techniques.

This paper does not intend to address the ubiquitous deployment issue of IP-multicast. Instead, it will focus on the reliability issue, one of the significant limitations of currently deployed IP-multicast networks. Reliability in IP-multicast scenarios is typically more challenging than a unicast scenario as it suffers from several unique problems, including heterogeneous client bandwidth and capabilities, bandwidth inefficiency and *NACK implosion* in IP-multicast retransmission [7], and potential sender overloading. This paper investigates the retransmission architectures and mechanisms to achieve more effective and reliable packet delivery in video IP-multicast. We propose a novel lightweight overlay peer-to-peer (P2P) retransmission architecture to exploit path diversity to address the above reliability-related issues in IP-multicast. An approach that leverages both disjoint-path-finding and periodic selective probing to take into account peer's recent packet loss probability, retransmission delay and recent retransmission success rate is proposed to effectively construct an efficient and dynamic overlay retransmission network. We evaluate its performance in comparison to the traditional source-based retransmission approach and demonstrate the significant advantages it provides in supporting the quality-of-service (QoS) performance (reliability, delay, scalability) of multimedia applications.

The main contributions of this paper are summarized as follows.

1) A hybrid framework that integrates IP-multicast network and lightweight P2P overlay network is proposed to address the reliability and scalability of both live and on-demand IP-multicast applications. Unlike previous hybrid approaches that mainly serve to "bridge" existing IP-multicast networks using overlay networks [9],

Manuscript received September 30, 2007; revised December 22, 2008. First published May 02, 2009; current version published July 17, 2009. This work was supported in part by a grant from Microsoft Research and in part by the NSF under Grant CNS-0423386. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Ishfaq Ahmad.

The authors are with the Computer Science Department, University of Missouri, Columbia, MO 65211 USA.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2009.2021712

[10], the proposed framework takes advantages of the strength of both in functionality by using IP-multicast (a more stable infrastructure) for bandwidth efficient data delivery and lightweight P2P overlay network for achieving reliability that otherwise has been previously shown to be very difficult to handle in IP-multicast.

2) A novel exploitation of both path diversity and peer-to-peer network is proposed to construct a highly distributed, robust, lightweight overlay retransmission network to significantly improve the *retransmission* performance for IP-multicast-based media distribution.
3) Cross-layer interaction is exploited that leverages existing functions supported by the current IP network to help construct an efficient, underlying network topology aware P2P overlay retransmission network.
4) *Given a deployed IP-multicast network*, the proposed framework is practical, scalable (in terms of multicast session size), and easy to deploy, requiring no change to the existing network infrastructure.
5) The proposed architecture can be further extended to bridge IP-multicast networks and non- IP-multicast networks and provide a unified framework to address both multicast connectivity and QoS performance.

Given that IP-multicast currently has deployment issue in wide area networks, this paper will focus on small to mid size IP-multicast networks in simulation. In addition, system implementation issues will only be briefly discussed. The focus will be on the retransmission architecture and how the proposed framework helps to address the reliability issue and reduce the end-to-end delay, which in turn can serve as a building block in various IP-multicast application systems.

The rest of the paper is organized as follow. Section II briefly discusses the challenges in reliable IP-multicast and some related work, and highlights how our approach differs. Section III presents our proposed path-diversity P2P overlay retransmission architecture for reliable IP-multicast. We present the details of the algorithm developed to construct the lightweight P2P overlay retransmission network in Section IV. Some additional design considerations such as complexity and overhead are also discussed there. Section V presents the simulation results and performance analysis. We conclude in Section VI.

II. Reliability in IP-Multicast and Related Work

To help address data loss in challenging network environments, forward error correction (FEC) [11], [12] and selective retransmission have been proposed to recover lost packets. FEC introduces bit overhead in the transmission, thus should be used judiciously. There is no efficient "one-fits-all" FEC solution in IP-multicast due to heterogeneous client channel conditions. In addition, delay will be introduced when FEC is applied across packets to address packet-loss. On the other hand, selective retransmission is generally considered more bandwidth efficient, at the cost of some delay penalty. There is ongoing work in IETF to support RTP retransmission (with the support of early feedback and more frequent RTCP feedback [13]) which is considered to be an effective packet loss recovery technique for reliable real-time applications with relaxed delay bounds [14], [34]. Selective retransmission can also be combined with FEC to achieve better performance, especially for reliable multicast [7], [15].

Traditionally, to address reliability, the receiver sends the retransmission requests to the original sender who then may choose to retransmit the lost packets if deemed important [7], [14]. A common problem in this approach is that it is very likely that the retransmitted packets will go through roughly the same routes as the original packets, which unfortunately are probably congested as suggested by the loss of the original packets. As a result, the retransmitted packets are also likely to experience congestion and loss, especially in a bursty loss scenario. If not done wisely, the retransmitted packets may even deteriorate the congestion condition [14]. In an IP-multicast scenario, there is also a scalability problem, more specifically, the request and reply message implosion problem [7], [16] in supporting retransmission. Typically the retransmitted packets are also IP-multicast to all participants who have subscribed to the retransmission session, even though only one or few participants might have experienced the loss of that particular packet. This would result in a waste of the bandwidth. The original sender may choose to only retransmit (multicast) packets that are requested by many participants at the cost of additional delay introduced in aggregating the feedbacks from multiple participants, and of some lost packets not being retransmitted. Separate unicast session can also be established by the original sender to convey retransmissions to each of the requesting receivers. This, however, will significantly increase the load of the original sender. Switching between unicast and IP-multicast retransmission based on the number of requests has also been considered [17], [18].

Error recovery for reliable IP-multicast transport with controlled bandwidth overhead and reduced latency through *local* IP-multicast retransmission has been studied in the past [17]–[29]. Most of these previous works, however, require various degrees of static configuration [17], [18], centralized coordination [20], or router/proxy support [21], [26]–[29], which make them difficult/costly to deploy in practice. In [22], a hop-based scope control for IP-multicast retransmission and use of separately addressed local recovery multicast groups are proposed to address local recovery. As discussed in [22], the hop-based scope control approach has performance limitation when compared to global error recovery if the topology of a session is star-shaped, and the group-scoped error recovery has more overhead on the members as well as on the underlying IP-multicast routing. Complexity and practical deployment is again a major obstacle. More significantly, the focus of all the above *local* IP-multicast retransmission approaches is mainly to address the bandwidth overhead issue inherent in IP-multicast retransmission. Path diversity is not exploited in any of the above approaches. As a result, retransmission still suffers from the congestion bottleneck on the IP-multicast tree.

Overlay networks have been considered as effective alternative solutions to address some fundamental and challenging networking problems. The concept has been used to build peer-to-peer systems for file sharing, content distribution and streaming, and distributed storage, e.g., in [30]–[34], to address resilient routing [35] and IP security [36], to develop application layer multicast [6], [37] to overcome the deployment issue of IP-multicast, etc. Path-diversity has also been exploited in the construction of some of the overlay networks, e.g., in [30], [35], and [38].

Error recovery for media streaming in application-level multicast is addressed in [39] by adding some extra paths or "links" into the ALM tree and replicating packets along these links with some probability, and in [40] by constructing multiple independent ALM trees to reduce the packet loss correlation between nodes on different ALM trees, at the cost of some additional delay and bandwidth inefficiency. In [40], only delay is used as the metric in choosing the recovery nodes and designing the retransmission schedule which are performed before data delivery or during the ALM tree reconfiguration. This multi-tree delay-centric approach is mainly targeted for recovering packet loss due to node join/leave/failure (a significant issue in ALM), but has limited performance for recovering packet loss due to network congestion, as close recovery neighbors, even though on different ALM trees, tend to suffer from the same network congestion problem. Note that the issues that need to be addressed to achieve reliability in IP-multicast and ALM are quite different. For example, packet loss in ALM is a result of network congestion, node join/leave, or node failure, while packet loss in IP-multicast is mainly due to network congestion, which is the main issue we address in this paper.

Typically, peer selection and maintenance plays an important role in a P2P system and often is the most costly part of the system, as peers are expected to serve the bulk data. In the first generation of P2P network, such as Napster [45], peer selection is done by a central server, which is not scalable. Peer selection can also be facilitated by leveraging different capacities of the nodes. For example, KaZaa [46] selects super nodes from peers which store some peers' information. A popular technique is to find peers using the distributed hash table (DHT), and it assumes that all peers have the same "hash" function and each content has a unique key. Examples are Chord [47] and Pastry [48], which concern about how to find a close peer with the requested content. They do not consider load-balancing and path quality fluctuation. In [49], the authors present a machine learning approach such as decision tree learning for peer selection and Markov decision for peer switching, but the online update of the rating system is a concern and the current system does not use multi-source transmission which is important for a P2P system. In [50], the authors present a topology-aware peer selection scheme and dynamic switching scheme for active sender selection in a multi-sender to one receiver streaming scenario, in which each peer will infer the topology of the network several times throughout the session by using logical topology, available bandwidth and loss rate (measured on a segment basis). After finding the logical topology, each receiver tries to cluster peers with similar path to the *receiver* and select a subset of senders who have path diversity to the *receiver* (so that they have a good aggregated streaming rate and smaller loss rate at the receiver). Because measuring available bandwidth in such application is important but costly, the authors also explore alternative ways to measure the bandwidth with less extra effort by trading off accuracy. Their approach relies on some underlying P2P substrates such as Pastry to provide, with additional lookup complexity, initial peer lists based on which the proposed peer selection is performed.

In our work, an important objective is to build a very lightweight retransmission overlay, as the overlay is not designed for bulk data delivery. We propose a two-step process in building the overlay. The first step is performed once by each receiver when it joins the multicast session and it concerns only about inferring the logical topology. In this step, the receiver finds the path information from the original sender to the receiver and by exchanging the information with other peers, the receiver can preliminarily identify a subset of candidate retransmission peers with disjoint path to the sender. The details of this step will be discussed in Section IV-A. The second step is to dynamically select the "best" retransmission node from the candidate subset on the fly by leveraging measured end-end delay and packet loss ratio through selective probing and taking into account successful retransmission rate, which will be discussed in Section IV-D.

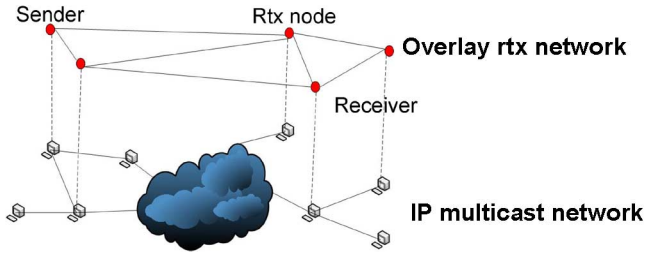

Fig. 1. Overlay retransmission network architecture for IP-multicast, where rtx is the abbreviation for retransmission, the bottom layer is the physical IP-multicast network, Sender is the actual sender in IP-multicast network, and both Rtx node and Receiver are receiving nodes in IP-multicast network.

III. Hybrid Network Architecture for Reliable IP-Multicast

The basic idea of the proposed path diversity overlay retransmission is to build a very simple overlay network among the participants of the IP-multicast session, in which each receiving node identifies a few peer receiving nodes who, upon requests, will be responsible for retransmitting a lost packet to the requesting receiving node through *unicast*. These identified peer nodes are called *retransmission nodes*. The original data packets will be delivered using IP-multicast to all receiving nodes. It is expected that in most cases the retransmission nodes would have received the original packets and thus could perform retransmission upon requests. This hybrid network architecture for reliable IP-multicast applications is illustrated in Fig. 1.

Intuitively, the retransmission nodes for a receiver should be those end hosts that have been recently having little problem receiving packets from the original sender (source), have a good network connection to the receiver, and are typically closer to the receiver than the original sender. Having little problem receiving data from the original sender implies that this candidate node is not sharing the congested link in the IP-multicast tree with the retransmission requesting node. Identifying peer nodes that have a more disjoint path to the sender will help to address this issue, as will be discussed in Section IV. The quality of the network connection between the candidate retransmission nodes and the receiver can be estimated by the receiver, for example, based on past retransmission experience, or by periodic probing in a streaming scenario. The quality of the network connection between the original sender and the candidate retransmission nodes is estimated by the candidate retransmission nodes themselves and then conveyed to the receiver upon periodic probing. When there are multiple sending sources (e.g.,

in video conferencing), each receiver identifies a small set of "good" retransmission nodes for each potential sending source. Note that, unlike in pure P2P applications, finding peers behind NAT/firewall is not a problem in an IP-multicast scenario, as discussed in Section IV-E.

With the identification of a small set of good retransmission nodes, a receiver can send the retransmission request (including the original sequence number of the lost packets) to *one* (presumably the best one) of its retransmission nodes who typically would have received or will receive that packet. The chosen retransmission node then forwards the requested packet, if available, using a separate *unicast* RTP session to the requesting receiver. If the retransmission nodes are chosen intelligently, the retransmitted packet would go through a unicast route that is different from the original congested path along the IP-multicast tree, and thus have a much better chance of getting to the requesting receiver reliably and timely, improving the user experience of the requesting receiver. Furthermore, this overlay retransmission architecture provides load balancing that redistributes the retransmission load of the original sender to other peers, making the system much more scalable. The retransmission nodes are highly distributed without the limitation of a regular topology, which makes the system very robust to network failure. It also addresses the potential bandwidth inefficiency problem of the traditional approaches that use IP-multicast retransmission.

Since the overlay network is mainly used for the retransmission purpose, the construction and maintenance of the overlay network should be very *lightweight*. Therefore, we choose to use a highly distributed *lightweight* overlay network. There is no central controller. Each receiver simply picks a small set of candidate retransmission nodes. Each receiver periodically probes its retransmission nodes. Based on its own load, the retransmission node only accepts a limited number of overlay requests. Therefore, the only extra traffic is generated by periodic probing. To reduce the probing overhead, the probing interval can be chosen to be relatively large, e.g., every tens of seconds. We will show in Section V that with a small cost of periodic probing, the overlay retransmission network can significantly improve the retransmission performance in the case of challenging network environments with dynamic background traffic.

IV. Algorithm Design

In an IP-multicast scenario, data loss is mainly due to network congestion. Typically, multiple nodes will simultaneously face the same congestion problem as they share the same congested link. Other nodes that have a more disjoint path to the sender may not have the identical congestion problem, therefore will have a good chance to help the nodes in trouble to recover the lost data, potentially using some paths different from the congested one. For example, in [38], the AS-level information is used to find alternative path. In our design, we consider how to explore both path diversity information and periodic probing to help dynamically identify the best retransmission candidates to recover the lost data.

The construction of the overlay network topology is the most important component in our design. Nodes can monitor the status of a small set of other nodes and the path quality through periodic probing to decide how to choose a neighbor in the overlay network. A basic probing approach is to send probing packets between each pair of nodes, as is done in RON [35], resulting in $O(n^2)$ overall probing traffic for an overlay network with a size of n. Although this method can acquire accurate information, the probing overhead may not be acceptable in our application scenario. In our approach, we use a periodic *selective probing* mechanism in which we only probe a small subset of a constant size of the overlay nodes, resulting in $O(n)$ complexity for the entire multicast group, as discussed in Section IV-C.

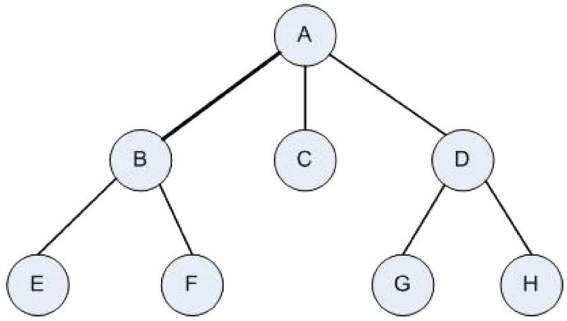

Fig. 2. Simple IP-multicast tree.

As discussed above, another important consideration in the design of the overlay network is the underlying network path diversity for the nodes in the overlay. We choose to use the readily available network utility function "Tracert" to gather the path information to the sender for each overlay node and identify nodes with disjoint path to the sender. Note that "Tracert" is only used to help *initially* identify a subset of candidate peers with path diversity property. Identification of more accurate candidate retransmission nodes is done dynamically later through periodic probing.

We have the following basic assumptions in our design.

- All the routers support the "Tracert" function.[1]
- The exchange of "Tracert" information is lossless.

A. Identify Path-Disjoint Retransmission Candidates

In IP-multicast, packets flow along an IP layer tree rooted at the source. The leaves of the tree are the receivers in the multicast session, the IP-multicast routers are the internal nodes on the tree, and the links form the edges of the tree. A very simple example is illustrated in Fig. 2, where node A is the root; nodes E, F, C, G, and H are the receivers; and B and D are IP-multicast routers.

In this structure, any packet loss along any "internal" link will result in all the downstream nodes/receivers not receiving the packet. For example, if congestion occurs along the link between node A and node B, then both node E and node F will have the same packet loss problem. This congestion, however, will not affect the receiving status of node C, G, and H. Therefore choosing some appropriate receiver node different from the

[1]Even if some routers do not support "Tracert" so that the path information gathered may not be complete, our proposed system with some variation is expected to still work without noticeable performance degradation.

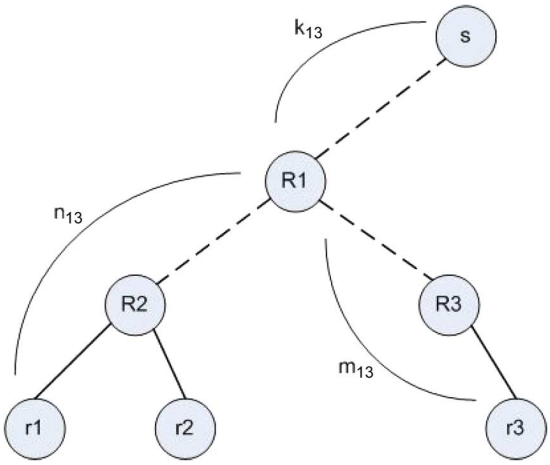

Fig. 3. Finding a retransmission node with the most disjoint path to the sender.

original sender as the retransmission node may help. A good strategy is to choose the receiver nodes with the most disjoint path to the sender. Here we define two sets of nodes, routers and end nodes (including both senders and receivers). We denote the routers with capital letter "R" and denote the end nodes with lower case "r" for receivers and "s" for senders. Assume the root is node s, and the receiver $r \in \{r_i | i = 1, 2 \ldots\ldots n\}$, where n is the total number of receivers in the multicast group, and let $P_{l_{ab}}$ be the loss probability of the link l_{ab} from node a to node b along the IP-multicast tree.

In Fig. 3, each dash line represents potentially more than one links in the tree. For a pair of nodes including a receiver node r_1 and one of its potential retransmission candidates, node r_3, k_{13} denotes the number of links from node s to node R_1; n_{13} is the number of links from node R_1 to node r_1; and m_{13} is the number of links from R_1 to r_3. When r_1 experiences packet loss, r_1 would try to identify a candidate retransmission node that has the highest probability of receiving that packet. The problem can be formulated as finding a "good" node g with the maximum probability $P(g = 1 | r_1 = 0)$, where $r_1 = 0$ means r_1 loses the packet and $g = 1$ means g receives the packet.

Let $\tilde{P}_{AB} = \prod_{1_{ab} \in \{\text{links from } A \to B\}} (1 - P_{1_{ab}})$ be the probability of successful transmission along all the links from a node A to another node B. We know that for r_3

$$P(r_3 = 1 | r_1 = 0) = \tilde{P}_{sR_1} \tilde{P}_{R_1 r_3} [1 - \tilde{P}_{R_1 r_1}] / [1 - \tilde{P}_{sr_1}]. \quad (1)$$

If for each particular receiver i and each of its potential retransmission candidate j, we can determine the values of k_{ij}, m_{ij}, and n_{ij}, we can then identify the retransmission candidates with the best disjoint-path. For this purpose, we use a common tool "Tracert" that is readily available in most, if not all, network systems. At the beginning of the multicast session, the active/joining receivers will send a "Tracert" request to the original sender to collect the information about the *unicast* route to the sender and then send it to others through the *IP-multicast channel*. We assume there is no packet loss during this route information exchange phase, so each node will have the information about the path and hop count to the original sender of all or sufficient number of active participants, and k_{ij}, m_{ij}, and n_{ij} can be determined. Denoted $c_{ij} = k_{ij} + n_{ij}$. Let us assume that $P_{l_{ab}}$ is the same for each link and can be denoted as P. Then we have

$$P(r_j = 1 | r_i = 0) = \left[(1 - P)^{k_{ij} + m_{ij}} - (1 - P)^{m_{ij} + c_{ij}}\right] / \left[1 - (1 - P)^{c_{ij}}\right]. \quad (2)$$

So for r_i, we need to identify $j = \arg\max(P(r_j = 1 | r_i = 0))$. In our simulation, to simplify the problem, we use the number of unshared links normalized by the total number of links on the path from the sender to the receiver to estimate this probability.

Note that "Tracert" collects the information about the *unicast* route to the sender, which may be different from the route on the IP-multicast tree. Nevertheless, this is sufficient for our application as only approximate route information is necessary to determine an initial set of path-disjoint nodes. In addition, periodic probing is also employed to dynamically determine the retransmission node, making the accuracy of route information less critical. Another important consideration of using unicast "Tracert" is that it is supported by all routers, making the proposed system more practical and deployable.

B. Integrated "Tracert" and Periodic Probing

"Tracert" allows a receiver to find good retransmission candidates with a more disjoint path. This will increase the probability of successfully retransmitting the lost packet to the receiver. However, as will be shown in our simulation results, "Tracert" alone is not sufficient to effectively solve the reliability problem. When the network load is more or less static, using retransmission node with the most disjoint path can perform well where a peer that does not share the same congestion problem as the receiver will typically be chosen. However, since the network load will typically change over time, there is also a high probability for a node with a disjoint path to have different congestion problems at the same time. In this case, it is difficult to tell whether the chosen retransmission peer, although with a good disjoint path, has the packet the receiver requests for. The only way for the receiver to judge whether the chosen retransmission node is performing well is to calculate its retransmission success rate. If the chosen retransmission peer is not performing well, the receiver could make a decision to switch to a different retransmission candidate. This adaptive method can help to identify a good retransmission candidate eventually, but it is a passive approach with a cost of delay. For delay sensitive multimedia applications, this may not provide satisfactory performance.

If a receiver is periodically updated about the status (e.g., delay and packet loss information) of other peers through unicast probing, it can identify a good retransmission candidate more quickly and accurately. But periodic probing alone also has some limitations. First, periodic probing only provides the peer's recent past information. If a receiver's close neighbor has no problem receiving data before congestion occurs, the receiver will most likely choose this close neighbor as the retransmission candidate as the retransmission path from this close neighbor is short. However, this close neighbor may share a lot of common

links on the IP-multicast tree with the receiver, therefore experiencing the same congestion problem as the receiver. Second, probing overhead is a concern that may limit the system's scalability. In the next subsection, we will discuss how to reduce the periodic probing overhead in our approach.

From the above discussion, we know that "Tracert" can help to identify a set of retransmission candidates with good disjoint path, but has limited performance when the network load is more dynamic. Periodic probing can perform well when the network is more dynamic, but cannot exclude close neighbors on the IP-multicast tree from being selected as the retransmission node in a timely manner. Therefore, a hybrid approach that integrates both "Tracert" and probing will get the benefits of both. It will help to identify a good set of retransmission candidate nodes by excluding close neighbors on the IP-multicast tree, and adaptively and timely choose a good retransmission node in a dynamic environment. In Section IV-D, we will describe in detail the design of the hybrid approach, and demonstrate its superior performance in Section V.

C. Reducing the Probing Overhead—Selective Probing

In an overlay network with n nodes, a full scale unicast probing will gather more accurate information. However, the probing overhead will be $O(n^2)$. There are ways to reduce the probing overhead in overlay networks. For example, in structured peer-to-peer systems [41], a node maintains connections to $O(\log n)$ neighbors. Thus the overall probing overhead is reduced to $O(n \log n)$. In our proposed overlay network, the main objective is to help retransmit the lost data for the requesting node. So the construction/maintenance process of the overlay should be very lightweight. Each node could periodically probe only a small subset of the nodes in the overlay network. For example, it can randomly select C nodes as the probing targets, and as a result, the probing overhead is reduced to $O(n)$. In our proposed solution, instead of randomly selecting the probing targets, the node selects the targets by considering the frequency that the target node appeared in its candidate retransmission nodes list. More specifically, a node always chooses C best (based on the previous performance) candidate retransmission nodes to probe. Other candidates will be probed much less frequently. As a result, the best candidates are updated with a unicast probing overhead of $O(n)$ for the *entire* overlay network.

D. Design of the Hybrid Scheme for Retransmission Node Selection

In this section, we describe in more detail the distributed algorithm we developed to build the lightweight overlay retransmission network. The process of building the overlay network mainly concerns the selection of the best retransmission nodes for each receiver. Once the retransmission nodes are determined, RTP unicast channels can be established directly between each receiver and its retransmission nodes. The algorithm is run at each receiver so that each receiver can dynamically determine its own retransmission nodes.

Based on our previous discussion, the best retransmission candidate should satisfy the following conditions:

1) uses less shared IP-multicast path with the retransmission requesting node;
2) not too "far away" from the retransmission requesting node, which means the retransmission path in the overlay should satisfy the delay constraint of the multimedia application;
3) has better receiving status than the retransmission requesting node, so that it has a high probability of having the packets that need to be retransmitted.

We could use the information obtained through "Tracert" to determine condition 1 and use the round-trip time (RTT) between the receiver and the original sender and the RTT between the receiver and the retransmission candidate to determine condition 2. Finally, condition 3 could be enforced by considering the packet loss ratio (PLR).

In each RTP IP-multicast session, each receiver in the multicast group can easily measure the RTT between itself and a particular original sender, as well as the packet loss ratio of the data it receives from that sender. Only the very recent RTT and packet loss ratio that reflect the current network condition are of relevance. Let us define RTT_i^j as the RTT between sender j and receiver i as measured by receiver i, and PLR_i^j as the packet loss ratio observed by receiver i for the data sent by sender j.

As discussed previously, at the beginning of each multicast session, peers will exchange their "Tracert" information. After gathering the "Tracert" information, each receiver k could calculate, for a particular sender j, the probability $P(i = 1|k = 0)$ denoted as $\mathrm{PT}_{\mathrm{ki}}^{\mathrm{j}}$ according to (1), where i is the potential retransmission candidate. Based on this information, a subset of peer nodes can be identified to serve as the initial candidate retransmission nodes.

In addition, each receiver will periodically probe through unicast a subset of other receivers about their receiving statistics with respect to the original sender. Once a node receives a probing packet, it will send its own measured RTT and packet loss ratio with respect to the original sender back to the probing node. For example, if node 1 receives a probing packet from node 2 for sender 3, node 1 will send RTT_1^3 and PLR_1^3 to node 2. Node 2 will also measure the unicast delay between node 1 and 2 in the probing process. For the retransmission purpose, the receiver should be more concerned about the delay between the retransmission nodes and the receiver as opposed to the delay from the original sender to the retransmission node.

The best retransmission node is chosen using the following strategy.

For node k, the retransmission node for sender j is

$$RTX_k^j = \underset{i=1,2,3\ldots \quad i\neq k}{\arg\min} \left\{ A * \left(\alpha \frac{RTT_k^i}{RTT_k^j} + \beta \frac{PLR_i^j}{PLR_k^j} \right) + B * F\left(PT_{ki}^j\right) \right\} T\left(1/G_k^i\right) \quad (3)$$

and $A + B = 1$, $\alpha + \beta = 1$, $(RTT_k^i/RTT_k^j) \leq 1$, and $PLR_i^j/PLR_k^j \leq 1$, where function F is to calculate the normalized $\mathrm{PT}_{\mathrm{ki}}^{\mathrm{j}}$, and is defined as $\mathrm{F(x)} = (1-\mathrm{x})^{\gamma}$, where γ is set to 1 in our experiments, RTT_k^i/RTT_k^j and PLR_i^j/PLR_k^j are the normalized RTT and PLR, respectively, for node i with

respect to sender j, which are not greater than 1 as nodes with larger RTT or PLR are not considered as good candidates, A, B, α, and β are the weighting factors, and G_k^i is the success rate of retransmitting packets from node i to node k, which is initially set to 1. After node i is chosen by node k as the retransmission node, G_k^i will reflect the quality of the retransmission path between node i and node k and also reflect how well node i receives data from the original sender. The choice of the function T will be determined by the system design and it could be a polynomial function. We use the linear function as the T function in our simulations. If G_k^i is low, it means that either node i often does not have the data that node k requests for, i.e., it probably also has a packet loss problem, or the path between node i and k is bad such that the retransmitted data from node i cannot reach node k. In fact, G_k^i is more important in the decision process than the normalized RTT and PLR, as in some cases, two nodes that are physically very close to each other and are in the low level of the IP-multicast tree are likely to face the same congestion problem. We need to be able to avoid these nodes being retransmission node of each other. We can avoid this situation by considering both the degree of path disjoint and the retransmission success rate. Because those physically close nodes will have similar packet loss pattern, when congestion occurs, although the normalized RTT and PLR collected in the last probing could be good enough, the retransmission success rate will decrease a lot because that node most likely does not have the packets requested to be retransmitted. Using (3), this problem can be successfully addressed very quickly. Because the decision will be frequently adjusted based on the performance of the selected retransmission node at a finer scale and on the periodic probing feedback from others at a coarser scale, it can address any potential faulty decision and switch to better retransmission nodes quickly.

E. Other Design Considerations

Although the focus of this paper is not on system implementation, it is worthwhile to discuss a few additional design considerations here.

Retransmission Session and Synchronization: We choose to use a separate *unicast* session to send the retransmission packets for each receiver, a good strategy that has been discussed sufficiently in [14]. As a matter of fact, we can exactly follow the recommendations made in [14] on the retransmission payload format, association/synchronization of a retransmission stream to its original stream, use of the retransmission payload format with the extended RTP profile for RTCP-based feedback (including retransmission requests) [13], congestion control, and other considerations, keeping in mind that the retransmission session is between the receiver and the retransmission node, as opposed to the original sender.

Dynamic Join and Leave: When a new node joins the multicast session, it will send a "Tracert" request to the original sender. As soon as it receives the reply, it will send the results through the IP-multicast session. Other nodes who receive this message will reply with their own "Tracert" results through *unicast*. In a multicast session with size of n, this is an overhead of $O(n)$ unicast operations plus *one* IP-multicast operation per joining node, and is done only *once* when the node joins. If overhead is still a concern for large n, then two strategies can be employed to reduce the unicast operations per joining node. One is random reply, with a probability depending on n resulting in a constant number of total replies, from existing nodes to the joining node. The other is to only let those existing nodes who deem, based on the information they have (e.g., path diversity or delay), that they might be a good node to reply to the joining node. When a node leaves the session, those who have selected this node as a retransmission candidate will detect it in the periodic probing process and will replace it with another good node in their candidate set.

A sudden increase of the number of users may cause the flash crowd phenomenon which often occurs in a traditional client-server network, and may still present a problem in a pure P2P broadcast network, although less severe. In our scenario, a new user only needs to send "Tracert" once to the sender. The bulk data is delivered via more stable and efficient IP-multicast. So potential bandwidth bottleneck caused by the "Tracert" packets is not a significant issue here. In a normal P2P situation, according to coolstreaming [42], the peak joining rate is about 150/min (or 2.5 nodes per second) in a system with 15 000 users. So the "Tracert" packet processing complexity and communication overhead is not an issue.

Peer Finding: Typically, NAT/firewall is a significant problem for a P2P streaming system. However, unlike pure P2P system, the availability of IP-multicast channel makes peer finding an easy job, as each new joining node will declare its existence by sending a message, e.g., its "Tracert" information, to the IP-multicast group. It will also learn the existences of other peers through their feedback message (e.g., in "Tracert" exchange). The limited uplink bandwidth in an asymmetric network does not pose a significant issue here either as the bandwidth requirement for retransmission is significantly smaller than regular data packet delivery.

V. Simulation Results

In this section, we evaluate the performance of the proposed path-diversity overlay retransmission mechanism. First, we will compare the performance of our proposed architecture based on "Tracert" only, periodic probing only, and the hybrid of them, against the traditional approach. Then in a large dynamic topology with random burst background traffic, we study the relationship between the probing interval and the performance and overhead introduced by periodic probing so that a good probing interval can be determined. We then show that selective probing will save significant probing overhead without degrading the system performance noticeably.

In all our simulations, the delay constraint for the *retransmission* (referred to as the retransmission deadline, which starts from when a packet loss is detected) is set to 400 ms to reflect a moderate real-time requirement, and retransmitted packets that do not arrive within the retransmission deadline will be treated as lost packets. We only consider the case of performing only *one* retransmission per lost packet. This is to focus our experiments on the effectiveness of the proposed retransmission mechanisms. Based on our tests using random

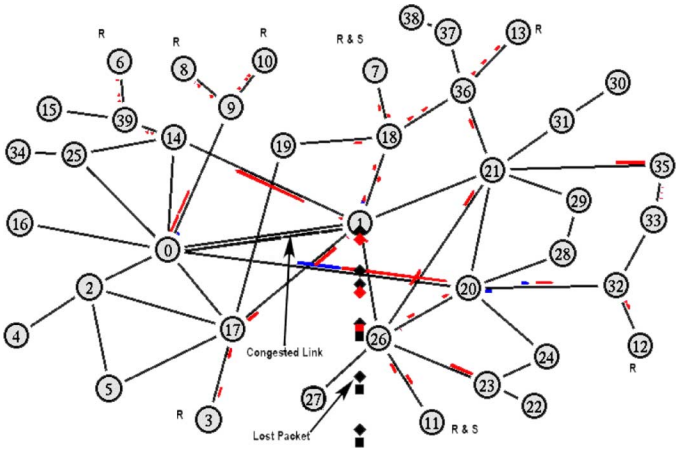

Fig. 4. One of the test topologies with 40 nodes. "R" and "S" represent receiver and sender, respectively. "Falling dots" below node 1 represent packet dropping at node 1 (for both RTP packets and background traffic packets). Other small dots on some of the links are multicast RTP data packets.

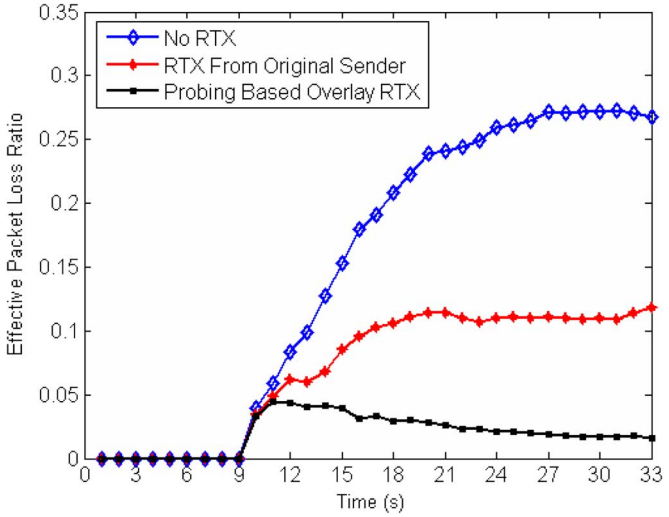

Fig. 5. Effective packet loss ratio of node 8 for the conventional retransmission and the periodic probing based overlay retransmission.

transit-stub (TS) graphs of size up to a few hundred nodes, we empirically identified a good set of parameters in (3), i.e., $A = 0.75$, $B = 0.25$, $\alpha = 0.3$, and $\beta = 0.7$, which performs better than any other choices we tested. When the deviation of α and β is 0.1 and the deviation of A and B is around 0.25, the deviation of the effective packet loss ratio is observed to be within 5% of the lowest effective packet loss ratio observed in the tests, which suggests that the performance is not very sensitive to the choice of the parameters. Unless otherwise specified, the periodic probing interval is 10 s.

A. Performance of "Tracert" and Periodic Probing

We use the *network simulator2* (ns2) [43] with some add-on codes to implement our protocol. The test topology is a transit-stub (TS) graph created by Georgia Tech GT-ITM network topology generator [44]. Fig. 4 shows one of the test topologies which we use here for ease of demonstration purpose. Results for a larger network with more dynamic background traffic will be presented in Section V-B. There are 40 nodes including the routers. Most of the link capacity is 1 Mbps and some of them are 2 Mbps. The propagation delays of the links range from 5 to 50 ms. The queue in the router can buffer 50 packets by default. The packet drop policy used in the routers is Droptail. In our tests, there are eight nodes joining the multicast group and two of them are data senders. In particular, nodes 11 and 7 both send data to the multicast group, and therefore act as both a sender and a receiver. Other active nodes, i.e., nodes 3, 6, 8, 10, 12, 13, serve only as receivers. Each of the two senders, nodes 11 and 7, sends a CBR (constant bit rate) video to the multicast group. The bit rate is 400 kbps. From the 9th second, there is a CBR (about 1.8 Mbps) background traffic on the link between node 1 and node 0 that has a bandwidth of 2 Mbps, which is a main path in both of the two sender-based IP-multicast trees. For node 8, the path from sender 11 in the IP-multicast tree is $11 \rightarrow 26 \rightarrow 1 \rightarrow 0 \rightarrow 9 \rightarrow 8$, which includes the congested link. For node 10, the path is the same except the final link. But for node 3, the IP-multicast path from node 11 is $11 \rightarrow 26 \rightarrow 1 \rightarrow 17 \rightarrow 3$, which does not include the congested link.

1) Performance of Periodic-Probing-Only Based Overlay Retransmission: We first compare the proposed periodic probing based overlay retransmission approach to the conventional approach that performs unicast retransmission from the original sender.

As shown in Fig. 5, the congestion occurs at about the 9th second when the background traffic starts to take effect. With the conventional retransmission approach, the effective packet loss ratio decreases to some extent, but is still as high as 10%. We can see that the periodic probing based overlay retransmission performs much better than retransmission from the original sender. In addition, if a physically close neighbor is selected to be a retransmission candidate because of its previous good receiving state, the measured retransmission success rate can help to address this problem by allowing another good candidate to be chosen [see (3)]. In Fig. 5, we observe a small peak for the periodic probing based overlay retransmission approach at the beginning of the congestion. This is because initially node 8 took node 10 as a good retransmission candidate because prior to congestion, node 10 was in an excellent receiving condition and is also close to node 8. Then within a short time, node 8 was able to detect that node 10 had problem, too, thus changing the retransmission node to node 3 who was really in a good receiving condition. This demonstrates the adaptability of our algorithm in choosing the best retransmission node.

2) Performance of "Tracert" Only Based Overlay Retransmission: It is observed that in most cases, physically close neighbors have little chance to be a good retransmission node of each other. Probing only based approach can avoid choosing such neighbors as retransmission candidates as discussed above, but it may take a couple of seconds for the receiver to detect the problem and correct it. We evaluate the second strategy, i.e., to choose the nodes with the most disjoint path to the sender to be the retransmission node. By taking "Tracert" results into account, the receiver can initially identify a least

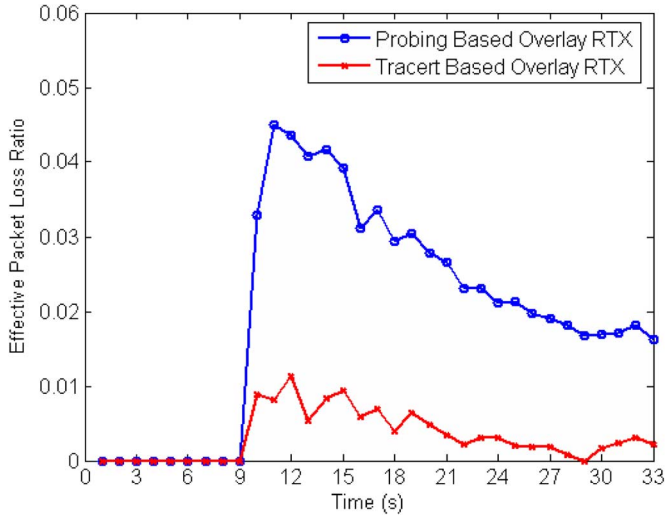

Fig. 6. Effective packet loss ratio of node 8 for the periodic probing based and the "Tracert" based overlay retransmission.

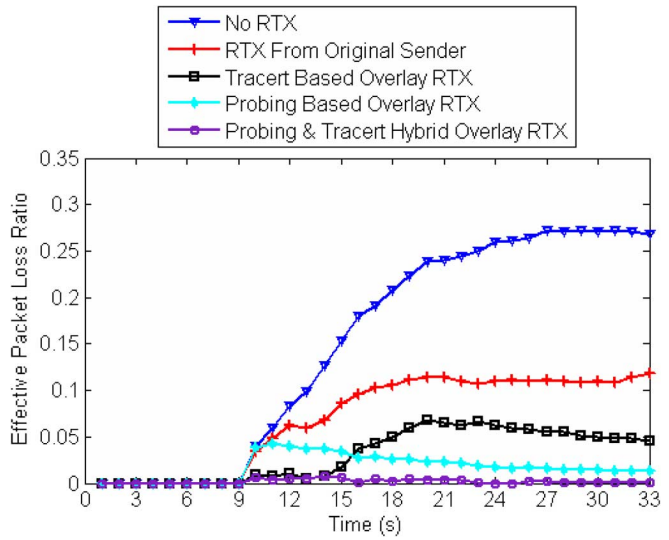

Fig. 7. Effective packet loss ratio of node 8 for different approaches when background traffic is more dynamic.

correlated node in terms of packet loss to avoid the close neighbor selection problem in the first place.

Fig. 6 shows that, with "Tracert", node 8 will choose the path-disjoint node 12 as the retransmission node. From the figure, we can see that the initial peak diminishes.

3) Performance of Hybrid Overlay Retransmission: We evaluate the impact of dynamic traffic on the performance of different approaches. To simulate dynamic traffic, another traffic is injected at the 15th second on link 20 → 32, which will greatly affect the receiving state of node 12. Apparently, node 12 is not a good candidate any more. If only "Tracert" is used, node 8 will still treat node 12 as a good candidate until it detects the problem based on the retransmission success rate. If only periodic probing is used, it cannot overcome the close neighbor selection problem. If we integrate probing information with "Tracert", it can dynamically determine an overall best node as the candidate retransmission node, since

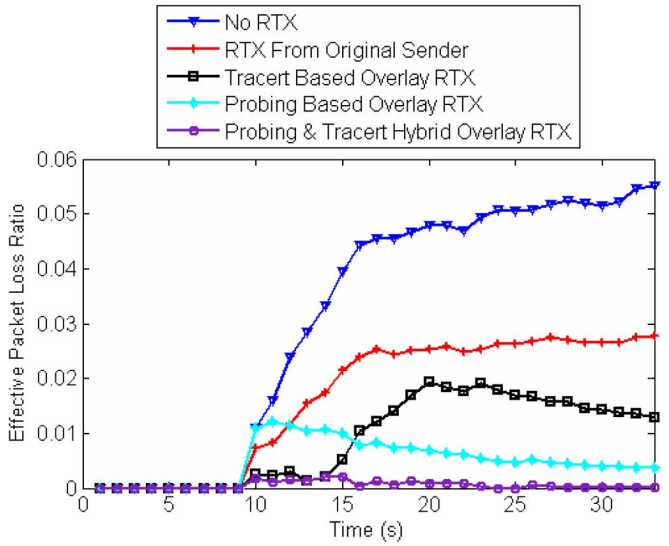

Fig. 8. Average effective packet loss ratio for all nodes for different approaches when background traffic is more dynamic.

it will avoid the close neighbor selection problem based on the "Tracert" information and select a node with disjoint path that has good receiving state. For this particular example, node 8 will choose node 3 as the best candidate. Fig. 7 shows the effective packet loss rate of node 8 for different approaches. It can be seen that the hybrid approach is very effective in finding a good retransmission node. The effective packet loss rate is reduced to almost zero. The "Tracert" only based approach, on the other hand, is not able to adapt to the dynamics of the network sufficiently. When the second background traffic takes effect at the 15th second, the performance suffers. Although the receiver did detect the problem based on the retransmission success rate and switch to other candidates, it has difficulty in finding a good retransmission node in a timely manner.

Fig. 8 shows the average performance for all nodes in the multicast group. We can see again that "Tracert" only based approach cannot adapt to the network dynamics well. The periodic-probing-only based approach suffers from the close neighbor selection problem discussed before. In fact, it may even perform worse than the conventional approach occasionally as shown around the 10th second. On the average, the hybrid approach performs the best as it takes both path diversity and dynamic receiving status into account.

We also observe that when the retransmission is from the original sender, the average delay introduced by retransmission is 332 ms (only received retransmission packets are considered here). With the hybrid overlay retransmission, the average delay introduced by retransmission is reduced to 145 ms. In summary, our proposed retransmission architecture can significantly decrease the effective packet loss ratio, at the same time significantly reduce the retransmission delay.

B. Scalability and Probing Overhead

In order to better understand the performance of our proposed architecture, we do several tests to show 1) the proposed architecture works well for large scale topologies with more dy-

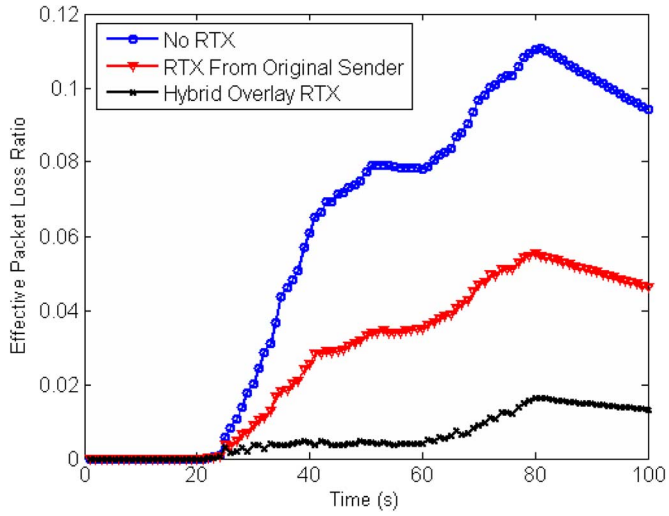

Fig. 9. Average "effective" packet loss ratio for all nodes for different approaches in a large topology.

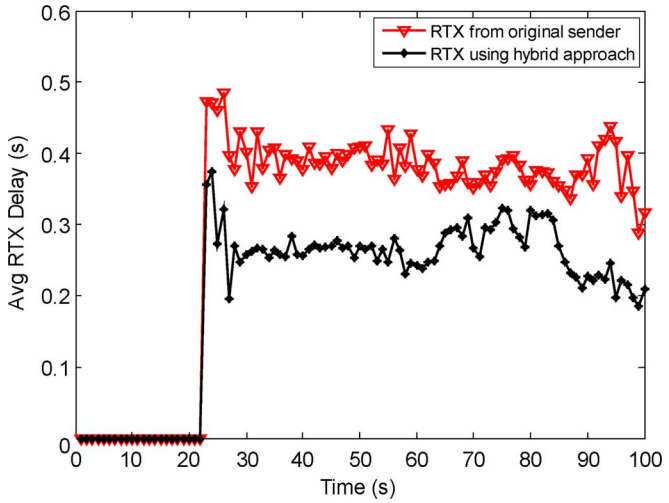

Fig. 10. Average retransmission (only received retransmitted packets are considered here) delay for all nodes for different approaches in a large topology.

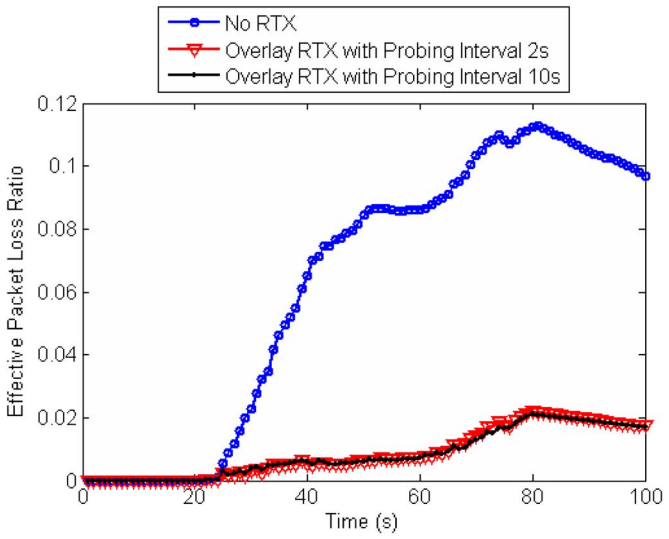

Fig. 11. Performance comparison between different probing intervals in hybrid overlay retransmission.

namic traffic; 2) appropriate probing interval can be determined to achieve a good performance without excess probing overhead; and 3) selective probing can be used to save the periodic probing overhead without sacrificing the performance.

In the following tests, we use GT-ITM to generate a 500-node TS topology. The average degree of each node is 1.9. In these tests, there are 40 nodes joining the multicast group and two of them are sending data to the group. The simulation runs for 100 s. The links' delay is distributed uniformly from 1 to 20 ms, the backbone link bandwidth is 10 Mbps, and the bandwidth of edge/last links is distributed from 1 to 2 Mbps. We use distance vector for IP-multicast routing and shortest path for unicast routing. To simulate the dynamics of the network, we randomly generate 400 exponential burst background traffics by NS-2, each traverses two randomly chosen nodes and has an average bit rate distributed from 500 kbps to 5 Mbps. The random background traffic is injected starting from the 5th second, and its duration is Gaussian distributed from 1 to 100 s.

1) Performance in a Large Topology With More Dynamic Traffic: Figs. 9 and 10 show the average performance of the proposed hybrid approach for the large topology compared to the traditional approach that performs retransmission from the original sender. We can see that the hybrid approach again significantly outperforms the traditional approach, and can recover about 85%–90% lost packets on average with only one retransmission attempt. This result demonstrates that our proposed architecture can scale well in large networks. We also observe from Fig. 10 that the average delay introduced by retransmission (only received retransmitted packets are considered here) is reduced from 361 ms for the conventional approach to 238 ms for the hybrid overlay retransmission.

2) Determine the Appropriate Probing Interval: In general, more frequent probing will gather more accurate information about the network dynamics, but at the same time, it will incur more overhead which will impose more stress to the links and potentially may worsen the system performance. Based on the above simulation setting, we set the probing interval to be 2 and 10 s, respectively, to study the retransmission performance.

From Fig. 11, we can see that the smaller probing interval has similar performance as the 10 s probing interval. However, the probing overhead is about five times as large. Our experiments show that a probing interval of 10 s is a reasonably good choice to achieve good retransmission performance. The overhead introduced is negligible compared to the multimedia data traffic. More results on the study of probing interval can be found in [4] where probing intervals of 1, 10, 25, and 100 s were tested for the topology shown in Fig. 4 and similar conclusion was reached.

3) Selective Probing: As discussed in Section IV-C, with a constant number of C probing targets, the receiver can randomly select C nodes in the multicast group to probe. This random probing may not be efficient as it may probe the bad candidates therefore missing the chance to probe more good candidates.

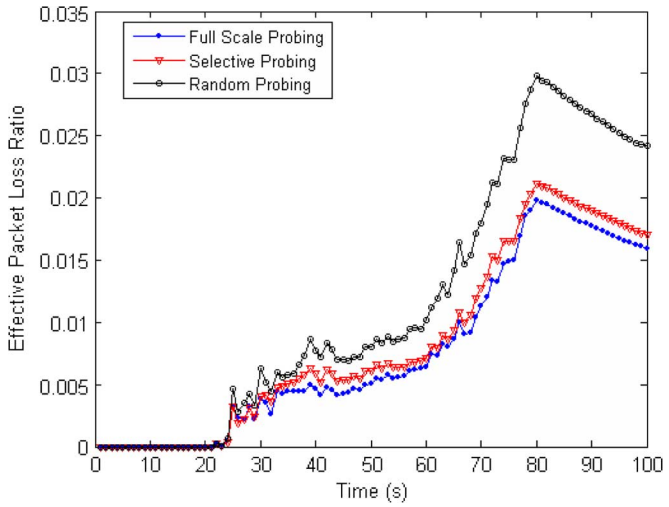

Fig. 12. Performance comparison of selective probing, random probing, and full scale probing.

The basic idea in selective probing is to give the current best candidates more weights when selecting the probing targets. Each receiver will maintain a sorted retransmission candidates list and exclude those nodes whose delay does not satisfy the delay constraint [see (3)]. In each probing period, the receiver selects several (six in the experiments) best candidates from the list to probe. For the other candidates, to get their updated information, the probing interval is three times as long as the probing interval of the best candidates.

Fig. 12 shows that selective probing, with much less probing overhead (i.e., only 6/40 of that of full scale probing), can achieve similar performance as the full scale probing. In addition, in scenarios with more dynamic random background traffic such as in this example, selective probing outperforms random probing significantly with the same probing overhead. In all three cases, the proposed hybrid scheme is used, with the only difference being the probing method.

VI. Conclusion

We introduce a novel path-diversity overlay retransmission architecture for IP-multicast based multimedia applications. The readily available network utility function "Tracert" is used to help identify the path disjoint retransmission nodes, and periodic probing is employed to adaptively and more accurately identify an overall good retransmission node. A hybrid approach that exploits both has been shown to achieve the best retransmission performance. Furthermore, to save the probing overhead, the receiver can use selective probing to probe only a subset of the candidate retransmission nodes who have the highest probability to be a good candidate. Simulation results show that it can significantly improve the reliability, delay, and scalability of IP-multicast based multimedia applications. Future work will include more quantitative evaluation of the impact of the proposed framework on the perceived quality of the multimedia applications in both simulation and real networks.

References

[1] K. Almeroth, "Multicast help wanted: From where and how much?," in *Keynote Speech, Workshop on Peer-to-Peer Multicasting, 2007 Consumer Communications and Networking Conf.*, Jan. 2007..
[2] Internet-2 homepage. [Online]. Available: http://www.internet2.edu/.
[3] W. Zeng, Y. Zhu, H. Lu, and H. Jiang, "A novel path-diversity overlay retransmission architecture for reliable multicast," in *Proc. IEEE Int. Conf. Multimedia and Expo*, Jul. 2006.
[4] Y. Zhu, W. Zeng, and H. Lu, "Exploiting overlay path-diversity for scalable reliable multicast," in *Proc. IEEE Int. Conf. Multimedia and Expo*, Jul. 2007.
[5] J. Beavers *et al.*, The Learning Experience Project: Enabling Collaborative Learning With ConferenceXP, Microsoft Research, Tech. Rep. MSR-TR-2004-42, Apr. 2004.
[6] Y. Chu, S. Rao, and H. Zhang, "A case for end system multicast," in *Proc. ACM SIGMETRICS Conf.*, 2000.
[7] B. Adamson, C. Bormann, M. Handley, and J. Macker, NACK-Oriented Reliable Multicast (NORM) Protocol, IETF Internet-Draft work in progress, Mar. 2007.
[8] S. Ratnasamy, A. Ermolinskiy, and S. Shenker, "Revisiting IP-multicast," in *Proc. ACM Sigcomm*, Pisa, Italy, Sep. 2006.
[9] D. Thaler, M. Talwar, A. Aggarwal, L. Vicisano, and T. Pusateri, Automatic IP-Multicast Without Explicit Tunnels (AMT), IETF Internet-Draft, work in progress, Nov. 2006. [Online]. Available: http://www.tools.ietf.org/html/draft-ietf-mboned-auto-multicast-07.
[10] Internet Research Task Force (IRTF), Scalable Adaptive Multicast (SAM) Research Group. [Online]. Available: http://www.samrg.org.
[11] W.-T. Tan and A. Zakhor, "Video multicast using layered FEC and scalable compression," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 3, pp. 373–386, Mar. 2001.
[12] T. Lestayo, M. Fernández, and C. López, "Adaptive approach for FEC reliable multicast," *Electron. Lett.*, vol. 37, no. 22, pp. 1333–1335, Oct. 2001.
[13] J. Ott *et al.*, Extended RTP Profile for RTCP-Based Feedback, IETF RFC 4585, Jul. 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4585.txt.
[14] J. Rey *et al.*, RTP Retransmission Payload Format, IETF RFC 4588, Jul. 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4588.txt.
[15] J. Nonnenmacher, E. W. Biersack, and D. Towsley, "Parity-based loss recovery for reliable multicast transmission," *IEEE/ACM Trans. Netw.*, vol. 6, no. 4, pp. 349–361, Aug. 1998.
[16] S. Pingali, D. Towsley, and J. Kurose, "A comparison of sender initiated and receiver-initiated reliable multicast protocols," in *Proc. ACM SIGMETRICS*, 1994.
[17] H. W. Holbrook, S. K. Singhal, and D. R. Cheriton, "Log-based receiver reliable multicast for distributed interactive simulation," in *Proc. ACM SIGCOMM*, Aug. 1995.
[18] J. C. Lin and S. Paul, "RMTP: A reliable multicast transport protocol," in *Proc. IEEE INFOCOM*, Apr. 1996, pp. 1414–1424.
[19] M. S. Lacher, J. Nonnenmacher, and E. W. Biersack, "Performance comparison of centralized versus distributed error recovery for reliable multicast," *IEEE/ACM Trans. Netw.*, vol. 8, no. 2, pp. 224–238, Apr. 2000.
[20] R. Yavatkar, J. Griffioen, and M. Sudan, "A reliable dissemination protocol for interactive collaborative applications," in *Proc. ACM Multimedia*, 1995.
[21] C. Papadopoulos, G. Parulkar, and G. Varghese, "An error scheme for large-scale multicast application," in *Proc. IEEE INFOCOM*, Mar. 1998.
[22] C.-G. Liu, D. Estrin, S. Shenker, and L. Zhang, "Local error recovery in SRM: Comparison of two approaches," *IEEE/ACM Trans. Netw.*, vol. 6, no. 6, pp. 686–692, Dec. 1998.
[23] B. N. Levine and J. J. Garcia-Luna-Aceves, "A comparison of known classes of reliable multicast protocols," in *Proc. Int. Conf. Network Protocols*, Oct. 1996.
[24] A. Mankin, A. Romanow, S. Bradner, and V. Paxson, IETF Criteria for Evaluating Reliable Multicast Transport and Application Protocols, RFC2357, 1998.
[25] K. Obraczka, "Multicast transport mechanisms: A survey and taxonomy," *IEEE Commun. Mag.*, vol. 36, no. 1, pp. 94–102, Jan. 1998.

[26] J. Gemmell, T. Montgomery, T. Speakman, and J. Crowcroft, "The PGM reliable multicast protocol," *IEEE Netw.*, vol. 17, no. 1, pp. 16–22, Jan./Feb. 2003.
[27] M. Calderon, M. Sedano, A. Azcorra, and C. Alonso, "Active network support for multicast applications," *IEEE Netw.*, vol. 12, no. 3, pp. 46–52, May/Jun. 1998.
[28] S. K. Kasera, S. Bhattacharyya, M. Keaton, K. Kiwior, S. Zabele, J. Kurose, and D. Towsley, "Scalable fair reliable multicast using active services," *IEEE Netw.*, vol. 14, no. 1, pp. 48–57, Jan./Feb. 2000.
[29] E. Kim, S. G. Kang, and J. Choe, "A router-assisted session tree configuration mechanism for reliable multicast," *IEEE Commun. Lett.*, vol. 6, no. 9, pp. 464–466, Sep. 2002.
[30] V. N. Padmanabhan, H. J. Wang, P. A. Chou, and K. Sripanidkulchai, "Distributing streaming media content using cooperative networking," in *Proc. ACM NOSSDAV*, Miami Beach, FL, May 2002.
[31] J. Li, PeerStreaming: A Practical Receiver-Driven Peer-to-Peer Media Streaming System, Microsoft Research, Tech. Rep. MSR-TR-2004-101, 2004.
[32] Z. Xiang, Q. Zhang, W. Zhu, Z. Zhang, and Y.-Q. Zhang, "Peer-to-peer based multimedia distribution service," *IEEE Trans. Multimedia*, vol. 6, no. 2, pp. 343–355, Apr. 2004.
[33] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum, "DONet/CoolStreaming: A data-driven overlay network for live media streaming," in *Proc. IEEE INFOCOM*, Miami, FL, Mar. 2005, vol. 3, pp. 2102–2111.
[34] J.-C. Wu, K.-J. Peng, M.-T. Lu, C.-K. Lin, Y.-H. Cheng, P. Huang, J. Yao, and H. H. Chen, "HotStreaming: Enabling scalable and quality IPTV services," in *Proc. IPTV Workshop in Conjunction With the 15th Int. World Wide Web Conf.*, Edinburgh, U.K., May 2006.
[35] D. G. Andersen, H. Balakrishnan, M. F. Kaashoek, and R. Morris, "Resilient overlay networks," in *Proc. ACM Symp. Operating Systems Principles (SOSP)*, 2001.
[36] A. D. Keromytis, V. Misra, and D. Rubenstein, "SOS: Secure overlay services," in *Proc. ACM SIGCOMM*, Aug. 2002, pp. 61–72.
[37] D. A. Tran, K. A. Hua, and T. T. Do, "A peer-to-peer architecture for media streaming (Zigzag)," *IEEE J. Select. Areas Commun.*, vol. 22, no. 1, pp. 121–133, Jan. 2004.
[38] T. Fei, S. Tao, L. Gao, and R. Guérin, "How to select a good alternate path in large peer-to-peer system?," in *Proc. IEEE INFOCOM*, 2006.
[39] S. Banerjee, S. Lee, B. Bhattacharjee, and A. Srinivasan, "Resilient multicast using overlays," in *Proc. ACM SIGMETRICS*, San Diego, CA, Jun. 2003.
[40] W.-P. K. Yiu *et al.*, "Lateral error recovery for media streaming in application-level multicast," *IEEE Trans. Multimedia*, vol. 8, no. 2, pp. 219–232, Apr. 2006.
[41] A. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems," in *Proc. IFIP/ACM Middleware 2001*, Heidelberg, Germany, Nov. 2001.
[42] X. Zhang, J. Liu, and B. Li, "On large-scale peer-to-peer live video distribution: CoolStreaming and its preliminary experimental results," in *Proc. IEEE Multimedia Signal Processing Workshop*, Shanghai, China, Oct. 30, 2005.
[43] The Network Simulator ns-2. [Online]. Available: http://www.isi.edu/nsnam/ns/.
[44] E. W. Zegura, GT-ITM: Georgia Tech Internetwork Topology Models. [Online]. Available: http://www.cc.gatech.edu/fac/Ellen.Zegura/gtitm/gtitm.tar.gz. 1996
[45] [Online]. Available: http://www.napster.com.
[46] N. Leibowitz, M. Ripeanu, and A. Wierzbicki, "Deconstructing the kazaa network," in *Proc. 3rd IEEE Workshop Internet Applications*, Santa Clara, CA, Jun. 2003.
[47] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for Internet applications," in *Proc. ACM SIGCOMM*, 2001.
[48] A. Rowstron and P. Druschel, "Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems," in *Proc. IFIP/ACM Int. Conf. Distributed Systems Platforms (Middleware)*, Heidelberg, Germany, 2001, pp. 329–350.
[49] D. S. Bernstein *et al.*, "Adaptive peer selection," in *Proc. 2nd Int. Workshop Peer-to-Peer Systems*, 2003.
[50] M. Hefeeda, A. Habib, B. Botev, D. Xu, and B. Bhargava, "PROMISE: Peer-to-peer media streaming using CollectCast," in *Proc. ACM Multimedia*, 2003.

Wenjun Zeng (S'94–M'97–SM'03) received the B.E. degree from Tsinghua University, Beijing, China, in 1990, the M.S. degree from the University of Notre Dame, Notre Dame, IN, in 1993, and the Ph.D. degree from Princeton University, Princeton, NJ, in 1997, all in electrical engineering.

He has been an Associate Professor with the Computer Science Department of the University of Missouri-Columbia since August 2003. Prior to that, he had worked for PacketVideo Corporation, Sharp Labs of America, Bell Laboratories, and Matsushita Information Technology Lab of Panasonic Technologies, Inc. He has also consulted with Microsoft Research, Huawei Technologies, and a couple of start-up companies. From 1998 to 2002, he was an active contributor to the MPEG4 Intellectual Property Management & Protection (IPMP) standard and the JPEG 2000 image coding standard, where four of his proposals were adopted. His current research interests include multimedia communications and networking, content and network security, wireless multimedia, and distributed source and channel coding.

Dr. Zeng has served as an Organizing Committee Member and Technical Program Committee Chair/Member for a large number of IEEE international conferences. He is an Associate Editor of the IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY and IEEE TRANSACTIONS ON MULTIMEDIA and is on the Editorial Board of IEEE MULTIMEDIA MAGAZINE. He co-guest edited the PROCEEDINGS OF THE IEEE's Special Issue on Recent Advances in Distributed Multimedia Communications published in January 2008 and was the Lead Guest Editor of IEEE TRANSACTIONS ON MULTIMEDIA 's Special Issue on Streaming Media published in April 2004. He is serving as the TPC Vice Chair for 2009 *IEEE International Conference on Multimedia & Expo*. In the recent past, he has served as the TPC Chair for the 2007 *IEEE Consumer Communications and Networking Conference* (CCNC), the TPC vice-Chair for CCNC 2006, the TPC Co-Chair of the *Multimedia Communications and Home Networking Symposium* of 2005 *IEEE International Conference on Communication*.

Yingnan Zhu received the B.E. degree in electronic engineering from Tsinghua University, Beijing, China, in 2001, the M.S. degree in electrical engineering from Chinese Academy of Sciences, Beijing, in 2004 and the Ph.D. degree in computer science from the University of Missouri-Columbia in 2009.

He is currently with Samsung Digital Media Solutions Lab in Irvive, CA. He was a summer intern at the Thomson Corporate Research, Princeton, NJ, from 2006 to 2008. His research interests include multimedia communication, P2P networks, IPTV, and wireless mesh networks.

Haibin Lu (M'04) received the B.E. and M.E. degrees in electronic engineering from Tsinghua University, Beijing, China, in 1997 and 1999, respectively, and the Ph.D. degree in computer engineering from the University of Florida, Gainesville, in 2003.

He joined the faculty of the Department of Computer Science, University of Missouri-Columbia, as an Assistant Professor in 2003. His primary research focus lies in algorithmic aspects of computer network and multimedia communication.

Xinhua Zhuang (SM'92) is C.W. LaPierre Professor of Computer Science at University of Missouri-Columbia. He has contributed chapters to eight books and had over 250 publications. His research work in morphological image processing, wavelet image coding, and video over IP is particularly notable.

Prof. Zhuang has served as General Co-Chair of 2004 *IEEE ICME Conference*, General Chair of 2005 *IEEE MMSP* Workshop, and General Co-Chair of 2007 *IEEE ICME Conference*.