Title:

PeerCast: Churn-Resilient End System Multicast on Heterogeneous Overlay Networks

Authors:

Jianjun Zhang

Ling Liu

Lakshmish Ramaswamy

Calton Pu

Contact Author:

Lakshmish Ramaswamy

Department of Computer Science, University of Georgia

415, Boyd Graduate Studies Research Center

Athens, GA 30602-7404

Phone: (706) 542-2737

Fax: (706) 542-2966

Email: laks@cs.uga.edu

PeerCast: Churn-Resilient End System Multicast on Heterogeneous Overlay Networks

Jianjun Zhang[♢], Ling Liu[♢], Lakshmish Ramaswamy[♠] and Calton Pu[♢]

[♢]College of Computing
Georgia Institute of Technology
Atlanta, GA 30332
{zhangjj, lingliu, calton}@cc.gatech.edu

[♠] Department of Computer Science
University of Georgia
Athens, GA 30602
laks@cs.uga.edu

Abstract

The lack of wide deployment of IP multicast in the Internet has prompted researchers to propose end system multicast or application-level multicast as an alternate approach. However, end system multicast, by its very nature, suffers from several performance limitations, including, high communication overheads due to duplicate data transfers over same physical links, uneven load distribution caused by widely varying resource availabilities at nodes, and highly failure-prone nature of end hosts. This paper presents a self-configuring, efficient and failure-resilient end-system multicast system called *PeerCast*. Three unique features distinguish PeerCast from existing approaches to application-level multicasting. First, with the aim of exploiting network proximity of end-system nodes for efficient multicast subscription management and fast information dissemination, we propose a novel Internet-landmark signature technique to cluster the end hosts of the overlay network. Second, we propose a capacity aware overlay construction technique to balance the multicast workload among heterogeneous end-system nodes. Third, we develop a dynamic passive replication scheme to provide reliable end system multicast services in an inherently dynamic environment of unreliable peers. We also present a set of experiments showing the feasibility and the effectiveness of the proposed mechanisms and techniques.

1 Introduction

In recent years application-level multicasting or end system multicasting (ESM) has emerged as a practical alternative to IP level multicasting for disseminating information to large sets of receivers (Yeo et al., 2004; Banerjee et al., 2002; Castro et al., 2002; Chawathe, 2000; Chu et al., 2000; Jannotti et al., 2000; Pendarakis et al., 2001; Ratnasamy et al., 2001b). However, supporting ESM in a dynamic Internet-scale environment poses a number of challenges. First, an ESM system usually replicates data on end-hosts and propagates them through multi-hop IP unicast links. A critical challenge for ESM systems is to achieve high efficiency and minimize multicast latency experienced by the end-hosts. Second, end-hosts from a wide-area network tend to vary widely in terms of their computing capacities, their access network bandwidths, and their willingness and ability to share their resources. Such heterogeneity manifests itself as the variations in the amount of workloads the different nodes can handle. Therefore, there is a need for an efficient ESM protocol that can organize end-hosts into efficient multicast overlays, and effectively balance multicast workloads on them. Third, it is widely recognized that large-scale distributed systems like peer-to-peer (P2P) network are confronted with high churn rates (Saroiu et al., 2002) with nodes entering and departing the system at arbitrary points in time. Ensuring high multicast service availabilities in such dynamic systems is crucial for the success of end system multicast.

Research in this area has mostly focused on mitigating the first challenge (Banerjee et al., 2002; Castro et al., 2002; Chawathe, 2000; Chu et al., 2000; Jannotti et al., 2000; Pendarakis et al., 2001; Ratnasamy et al., 2001b). In contrast, the second and third challenges have received very little research attention. We believe that these distinct challenges are in fact closely related. Unfortunately, none, to our best knowledge, has comprehensively addressed these problems. Further, even the schemes proposed to counter the efficiency challenge suffer from significant limitations.

In this paper, we present *PeerCast* – an efficient, self-configuring, and overlay churn resilient end system multicast service, which is built on top of an overlay network of loosely coupled and possibly unreliable end-system nodes. Our ESM service can enable group communication capabilities in generic P2P networks, thereby providing a platform for advanced applications such as audio/video

conferencing, event/content distribution, and multi-user games.

PeerCast uses a structured P2P network protocol to organize end-hosts into an overlay network, and builds ESM applications using the P2P network as communication substrate. While a few existing systems have addressed similar problems (Castro et al., 2002; Ratnasamy et al., 2001b), our approach has three unique features. First, we develop a decentralized mechanism to effectively cluster end-hosts by their physical network proximity in the PeerCast P2P network. Our novel multicast group management protocol utilizes these clusters to build efficient multicast trees so that latencies and overheads of the multicast information dissemination are minimal. Second, we propose a capacity-aware overlay construction technique to balance the multicast load among heterogeneous peers. This scheme can effectively distribute the workload among end-hosts. To the best of our knowledge, PeerCast is the first ESM system that takes end-system heterogeneity into account. Further, our scheme also encourages peers to share more resources by providing better services to the peers contributing more resources. Third, we develop a dynamic passive replication scheme in order to provide reliable end system multicast service in an environment of inherently unreliable peers. This paper reports a set of experiments to evaluate the proposed techniques. The results indicate that the PeerCast system is highly efficient and it exhibits very good load-balancing and reliability characteristics.

2 PeerCast System Overview

PeerCast is an overlay network-based application level multicast system. Its design incorporates several techniques for alleviating the above mentioned performance limitations. This paper presents a detailed description of these techniques and mechanisms. However, for the sake of completeness and self-containment, we first briefly discuss the basic design architecture of the PeerCast system. Detailed description about the architecture can be found in the technical report version of the paper (Zhang et al., 2004).

2.1 System Architecture

The high-level design of the PeerCast is similar to the SCRIBE system (Castro et al., 2002). However, there are some important differences between the two systems, which we highlight at appropriate locations.

The nodes in the PeerCast system interact with one another in a P2P fashion. Henceforth, we use the terms *node* and *peer* interchangeably. Any peer can create a new multicast service of its own interest or subscribe to an existing multicast service. The peers are not required to have global knowledge about all other peers or about all the multicast services that are currently being offered. Further, the peers can enter and exit the system at arbitrary points in time.

Each peer in our system is equipped with a PeerCast middleware, whose design is depicted in Figure 1. The PeerCast middleware is composed of two functional substrates: *ESM Management* and *P2P Network Management*.

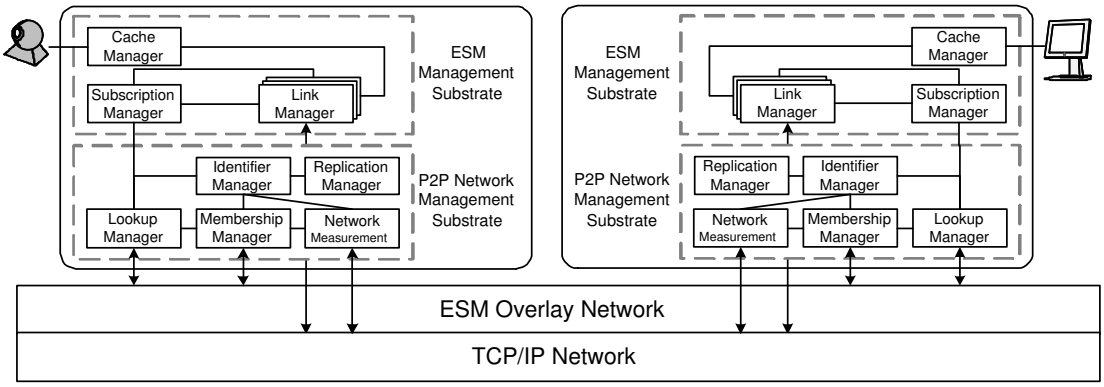

Figure 1: The PeerCast system architecture showing its two functional substrates

The ESM substrate is responsible for ESM event handling, multicast group membership management, multicast information delivery, and cache management. This layer utilizes the services provided by the underlying network management layer and incorporates three protocols, namely *multicast group membership management protocol*, *multicast information dissemination protocol* and *multicast overlay maintenance protocol*. The multicast group membership management protocol provides mechanisms for the peers to create new multicast service, subscribe to an existing multicast group, or exit from an existing multicast group. As discussed later in the paper, this protocol incrementally builds a multicast tree from the subscribers of a multicast service. The multicast group membership management protocol propagates the multicast payload through this tree so that it reaches all the subscribers. The multicast overlay maintenance protocol handles the node exits and failures by appropriately repairing the tree.

The P2P network management substrate is the lower tier of the PeerCast middleware. It provides services such as network membership management, resource lookup, and communication among end-

hosts. A peer invokes the services provided by this layer for entering the network and for communicating with other peers in the network.

The peers in the PeerCast system are organized as a *distributed hash table (DHT)*-based *structured* P2P network (Stoica et al., 2001; Ratnasamy et al., 2001a; Rowstron and Druschel, 2001; Zhao et al., 2002a). The hash function maps multicast tasks to the peers in the system. PeerCast P2P network protocol design differs from the existing DHT–based protocols, such as Chord (Stoica et al., 2001), Pastry (Rowstron and Druschel, 2001), Tapestry (Zhao et al., 2002a), and CAN (Ratnasamy et al., 2001a), in two important ways. First, PeerCast incorporates an efficient mechanism to distribute the multicast tasks to heterogeneous peers according to their capabilities. Second, PeerCast incorporates network proximity information into the topology of P2P network, aiming to bridge the mismatch between P2P network topology and physical network topology.

In the PeerCast system, each multicast service has two unique m-bit identifiers associated with it, namely the *service identifier* and the *group identifier*. The *service identifier*, represented as S_{id}, uniquely identifies the multicast service. It is used to publish summary information about the multicast service. The group identifier g_{id} will be used to identify the group of peers subscribed to the service.

The PeerCast system provides a distributed lookup service for peers to lookup, subscribe to, and unsubscribe from a multicast service using its group identifier g_{id} as a handle. It maps each multicast subscription request to a number of peers that will forward the multicast content to the subscriber.

Each peer in the PeerCast system is also assigned a set of m-bit identifiers. The number of identifiers assigned to a peer depends upon the amount of resources it donates. A peer p is described as a tuple of two attributes, denoted by $p : (\{peer\_ids\}, (peer\_props))$. $peer\_ids$ is a set of m-bit identifiers, each of which is denoted by $\frac{m}{d}$ digits, where each digit is a substring of d-bit. $peer\_props$ is a composite attribute that is composed of peer properties like its IP address, port number, and its resources (such as network bandwidth, CPU power and memory). Peers own multiple identifiers for load balancing purposes (see Section 5). However, in the interest of simplicity, let us for now assume that each peer is associated with a single ID. The peer that is associated with ID i is denoted as p_i.

In order to better understand the protocols and mechanisms of the PeerCast system, the peer identifiers, multicast service identifiers and the multicast group identifiers can all be conceptualized as being

points on a logical ring with range from 0 to $(2^m - 1)$. With this conceptual model as the basis, we first define a few terms which will be used to explain the various aspects of the design architecture of the PeerCast system. The distance between two identifiers i, j (denoted as $Dist(i, j)$), is the shortest distance between them on the identifier circle, defined as $Dist(i, j) = min(|i - j|, 2^m - |i - j|)$. Identifier i is considered as being *numerically closest* to the identifier j when there are no other identifiers with a shorter distance to j, i.e., $\forall_{k \neq j} Dist(k, j) \geq Dist(i, j)$. A peer p' with its peer identifier j is said to be an *immediate right neighbor* to a peer p with its peer identifier i (denoted by $p'_j = IRN(p_i)$), if there are no other peers that have identifiers in the clockwise identifier segment from i to j on the identifier circle. The term *immediate left neighbor* is analogously defined. Each peer maintains a list of neighboring peer identifiers and their reference information. Specifically, for an identifier i, the peer p_i keeps track of r successor identifiers on the logical ring and r predecessor identifiers on the logical ring. The peer p_i can communicate directly to any of the peers owning the identifiers that are present in the neighbor list. The peers also maintain a local routing table besides the neighbor lists for each of its identifiers. Details about routing table maintenance and handling of peer dynamics can be found in the associated technical report (Zhang et al., 2004).

The PeerCast P2P network provides a basic lookup service. Given an m-bit identifier k, the PeerCast P2P network maps it to a peer that has the identifier sharing the longest prefix with k. The lookup request can be initiated by any peer in the P2P network. From the initiating peer the lookup request is forwarded towards the target peer progressively. At each step, the request is forwarded to a peer whose identifier shares one additional digit with k than the current peer. Thus, the lookup request is resolved in at most $\frac{m}{d}$ steps.

2.2 Basic Multicasting in PeerCast system

In PeerCast, multicast service establishment and maintenance occurs through three distinct operations.

Publishing the Multicast Service: Multicast sources join the PeerCast P2P overlay as peers. As mentioned earlier, each multicast service is associated two identifiers, namely, service identifier and group identifier. The service identifier will be used to advertise and publish meta-information about the service, whereas the group identifier would be used by peers to subscribe to and unsubscribe from

the multicast service. A peer generates these two identifiers for each of its multicast services. Suppose a peer p_i wants to initiate a new multicast service S, it selects one of its unused peer identifiers and uses it as the group identifier of the multicast service MS. In contrast, the peer generates the service identifier S_{id} by replacing a substring one of its ids by a number it obtains from a certification service.

After generating the two identifiers for the multicast service, the source publishes the summary of the multicast service and its group identifier on another peer in the system. The node which hosts the summary of the multicast service MS is called MS's *rendezvous node*. The rendezvous node is determined by executing the above lookup protocol on MS's S_{id}. The source peer initiates a lookup query on S_{id} which discovers the rendezvous peer in at most $log_2(N)$ hops. The source node publishes the summary and the group identifier of the new multicast service on the discovered rendezvous node. Any other node in the system can now obtain summary and the group identifier of the multicast service by performing a similar lookup on the service's S_{id} and contacting the respective rendezvous node.

Multicast Tree-based Subscription Management: Now let us see how a peer can subscribe to a multicast service that it is interested in. As described above a peer can lookup the rendezvous node of any existing multicast service and obtain its group identifier. Since the group identifier of a multicast service is always chosen from one of the source peer's identifiers, the lookup operation (described in Section 2) always maps the group identifier to the service's source node irrespective of the scale or the dynamics of the network. Hence, an arbitrary node in the system can locate the source node of a multicast service given the service's group identifier.

The newly subscribing node (denoted by p_j) issues a subscription request with the group ID of the multicast service. This subscription request is treated exactly like a lookup request on the group ID, and is forwarded towards the multicast source through a series of intermediate peers whose identifiers satisfy the progressive prefix matching criterion. Eventually, the request reaches the source node, which subscribes p_j to the multicast service and starts sending the multicast data. However, in many cases it is not necessary to forward the request to the source node. If one of the intermediate nodes has already subscribed to the multicast service requested by p_j (i.e., the node is already in the multicast tree), the forwarding of the multicast subscription request is terminated. Instead, the intermediate node that is already in the multicast tree adds p_j as a new leaf to the multicast tree and starts forwarding the

messages of the multicast services to p_j. Thus, we see that each subscription request adds one or more edges into the multicast tree. Conversely, the multicast tree is pruned through analogous operations when nodes *unsubscribe* from the respective multicast service (Zhang et al., 2004).

Dissemination of Multicast Payload: The source of a multicast service uses the corresponding multicast tree for delivering the multicast data to all the subscribers. It injects the data at the root of the multicast tree, which then gets disseminated through the tree and reaches all the subscribers.

2.3 Limitations of the Basic End System Multicast

The above ESM scheme has three limitations that can hinder its performance. These limitations need to be adequately addressed in order to ensure scalability, efficiency, and reliability of the ESM.

The first limitation arises from the mismatch between the P2P overlay network and the underlying physical network, which has been well studied in prior literature (Ratnasamy et al., 2001a; Rowstron and Druschel, 2001; Xu et al., 2003b). In most generic P2P networks (including the basic PeerCast network), the indexing techniques and the routing schemes are completely independent and oblivious to the underlying network structure. Hence, communications in these networks are likely to be very inefficient in terms of the physical network routes traversed by individual messages.

The overlay-underlay mismatch manifests in the following manner. The multicast tree that is constructed at the logical level can be very inefficient in terms of the physical network connections. Consider the multicast tree depicted in Figure 2, wherein six end-hosts located in three states participate in an ESM overlay. Node WA 1 serves as the multicast root and all others are subscribers. Since the multicast tree is constructed without considering the physical network, the multicast messages have to travel from WA 1 (located in Washington) to GA 2 (located in Georgia) and then again traverse the link from GA 2 to WA 2. Thus, the multicast messages have to traverse twice between the east coast and the west coast and three times along the east coast. This not only affects the multicast latency, but also increases the load on the underlying physical network. Instead, if the nodes were to be organized in a tree as shown in Figure 3, the multicast would be significantly more efficient as the multicast messages do not have to traverse to and fro between the nodes located at the two coasts. However, because the basic PeerCast scheme is oblivious to the underlying physical network, it cannot ensure

efficiency of the resultant multicast tree.

Figure 2: Less efficient multicast overlay topology

Figure 3: A more efficient multicast overlay that groups peers by their network-proximity

The second shortcoming of the basic ESM mechanism is the load imbalance between the participating nodes. One of the main causes of load imbalance is the heterogeneity in the resource availabilities at various nodes. Ideally, the load on each peer should be proportional to its resource capabilities. This prevents nodes from becoming bottlenecks, thereby improving the efficiency and dependability of the system. Therefore, it is essential to augment the basic ESM multicast scheme with techniques that distribute the multicast load among the various nodes in the system in accordance with their resource availabilities.

The third limitation of the basic ESM multicast scheme is that its re-subscription-based mechanism to repair damaged multicast trees are often ineffective. This drawback becomes especially pronounced when the P2P network is highly dynamic, which is the case in most P2P systems. Therefore, we need to design mechanisms that can guarantee reliability of the PeerCast system, even when the P2P network exhibits considerable churn.

The remainder of this paper is dedicated to the design of effective and efficient mechanisms to overcome each of the above limitations of the basic PeerCast scheme.

3 Network Aware Multicast Tree Construction

Our discussion in Section 2 highlighted the need and importance of taking the physical network locations of the nodes into consideration while multicast tree. However, developing multicasting mechanisms that are sensitive to the physical network locations of the nodes poses two significant research challenges. The first challenge is to device techniques for accurately estimating the relative locations of the nodes in the physical network. Due to the decentralized and highly dynamic nature of the PeerCast network, it is not possible to maintain a complete global view of the overlay network or the underlying physical network. The second major challenge is to incorporate the network position awareness into

the decentralized framework of the ESM multicast tree construction framework.

Prior works such as Pastry (Rowstron and Druschel, 2001) and CAN (Ratnasamy et al., 2001a) have explored various techniques improve the network awareness of P2P systems. In Pastry (Rowstron and Druschel, 2001), peers probe the network neighbors and carefully select entries to fill their routing tables. A list of network neighbors is setup along with the routing table to accelerate the routing. However, the Pastry approach suffers from two major problems. First, its assumption about the triangular inequality properties in the IP network may not always hold. Second and more importantly, due to *logarithmic routing deterioration* (Xu et al., 2003b), the benefits obtained by this scheme might be extremely limited. The CAN system (Ratnasamy et al., 2001a) tries to map a peer into the CAN identifier space by its network coordinate information. This approach may introduce uneven identifier distribution in CAN hypercube space, thus resulting in some routing paths that are of poor quality.

We now explain our approaches to address the above twin challenges. The PeerCast system includes an Internet landmark-based technique for estimating the relative locations of the peers in the network. Conceptually, Internet landmarks (landmarks, for short) (Dabek et al., 2004; Ng2003, ; Tang and Crovella, 2003) are a set of few key Internet hosts that serve as a frame of reference for determining the relative position of any other node on the Internet. An arbitrary node measures the round trip time to each of these landmarks and uses these values to determine its relative location in the physical network.

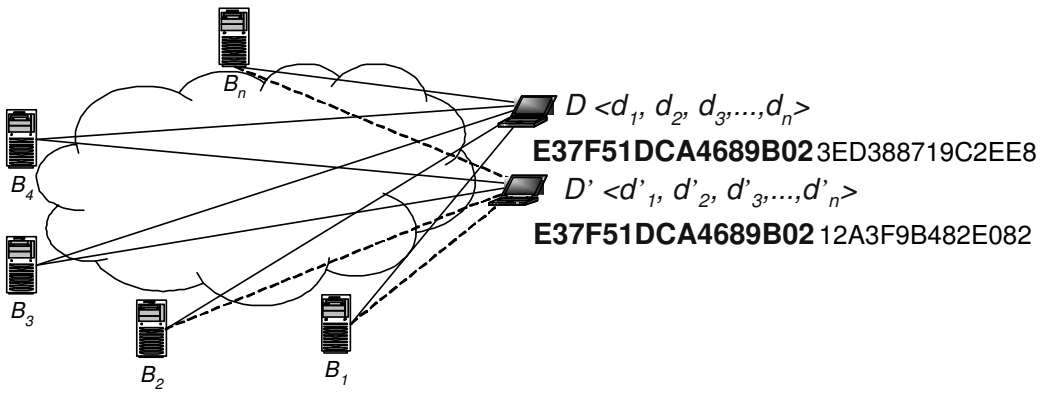

Figure 4: Landmarks-based peer ID generation − The two peers construct their landmark vectors by measuring their distances to each landmark node. The peers IDs are generated based upon the respective landmark vectors

Concretely, let us denote the landmark set as $\{B_1, B_2, B_3, \ldots, B_n\}$. When an end-host joins the network, it obtains the landmarks set through the bootstrapping service. An end-host measures its

distance (round trip time (RTT)) to the given set of landmark points and records the results in a vector $D < d_1, d_2, d_3, \ldots, d_n >$, which we refer to as its *landmark vector*. The intuition behind using landmarks vectors is that the end-hosts that are close to one another will have similar landmark vectors as their probing packets are more likely to traverse through similar network routes to reach each landmark point.

Designing completely decentralized multicast tree construction techniques that seamlessly utilize the above information about relative locations of nodes is a more difficult challenge. Observe that the multicast tree construction technique of the basic PeerCast scheme is fully decentralized. While retaining this core decentralized structure of the basic ESM mechanism, we augment it with two unique techniques so that the resultant multicast tree is sensitive to the relative locations of the nodes thereby optimizing the communication costs.

The first technique is a novel landmarks-signature scheme which embeds network position information of each node into its identifier. Recall that the three operations of the basic PeerCast scheme, namely publishing the multicast service, managing the multicast subscriptions and disseminating the multicast payload, are directly dependent upon the identifiers of the nodes in the P2P network. By embedding network position information into the node identifiers, we attempt to ensure that the generated multicast tree is efficient in terms of the communications costs. The basic idea is to allocate identifiers such that the peers that are close in terms of their locations in the physical network would have numerically closer peer identifiers. Specifically, an arbitrary peer p sorts its landmark vector and creates a list of landmark IDs that is ordered by the landmarks' RTT values to p. We utilize this ordered list of landmark nodes to capture the relative location of the peer p in the physical network. Concretely, the peer p encodes the ordered list of landmark node IDs into a binary string to obtain its *landmark signature*. All the peers in the system generate their landmark signature in an identical fashion. Since peers that are located in close network proximity tend to have similar landmark vectors, their landmark signatures would be similar too. Thus, by comparing the landmarks signature of two nodes one can infer their proximity within the network.

One question that arises is: *why not use the landmark signature of a peer as its identifier?* However, previous studies have shown the importance of preserving enough randomness of the identifier

distribution (Xu et al., 2003b). Hence, we see that the identifiers of nodes have to satisfy two conflicting requirements. On the one hand they should encapsulate the node's position information, but on the other, they should also have enough randomness. We balance these conflicting requirements as follows. Upon entering the network, a new peer generates its identifiers using the normal identifier generation functions such as MD5 and SHA-1. Further, it also measures its distance to various landmarks and generates its landmarks signature. Now, it replaces a substring of the generated identifier with the landmark signature at a certain offset. We call this offset as the *splice offset*. Its value is a system parameter that could be tuned according to the overlay population. The modified identifiers now contain information that can be used to identify the network location of the new peer while preserving randomness to some configurable degree. Figure 4 illustrates the landmark signature scheme for peer identifier generation. The two peers probe the landmark nodes and construct their landmark vectors. They generate their peer identifiers based on their landmark vectors. The part of the peer identifier that is in bold font represents the landmark signature of the peer. Observe that this part is identical for both peers indicating that they are in close network proximity.

The peer identifiers can be used to cluster the peers based on their network proximities. Suppose the splice offset is set to l digits. We can now envision the leading l digits (ld bits) before the splice offset to randomly partition the identifier space into 2^{ld} *buckets*. Hence, the peers from the same network locality are uniformly scattered into these identifier buckets. The Peers that are in close network proximity are still clustered together within each bucket by their landmark signatures. But they will not occupy a large continuous P2P identifier segment. The advantage of this approach is that it reduces the probability of the P2P network getting partitioned by the network domain level failures, which would force a large number of peers to depart at the same time.

The peer clustering improves the multicast performance as follows. In our scheme, the peers that are close to one another have similar identifier prefixes. But due to the splice offset, these peers may be in different buckets. When a subscription request is forwarded among a set of peers sharing the same prefix, there is a high probability that this request is forwarded among physical network neighbors that are in different buckets. Thus, when we build the multicast tree using the lookup method of PeerCast P2P protocol, we seamlessly achieve higher efficiency at the top portion of the multicast tree. This is

because the nodes located at the top of the tree share longer identifier prefixes with the multicast root node.

While our landmark signature technique improves the efficiency of the top portion of the resultant multicast tree, enhancing the efficiency of the bottom portion (portion near the leaves of the tree) requires additional mechanisms. Towards this end, we develop our second technique called *neighbor lookup*. As the name suggests, each peer initiating or forwarding a subscription request will first query its P2P network neighbors to see if they are already in the requested multicast tree. The closest neighbor in the multicast tree is chosen as the *potential parent*. To avoid looping of the subscription request among neighbors, a peer compares its distances to its potential parent and the parent of the potential parent. The subscription request will be forwarded to the closer of the two. The neighbor lookup is essentially a local operation in PeerCast, because P2P neighbors frequently exchange status information to maintain the integrity of the P2P routing tables. Thus, a peer can learn the subscription statuses of its P2P neighbors by checking its local cache.

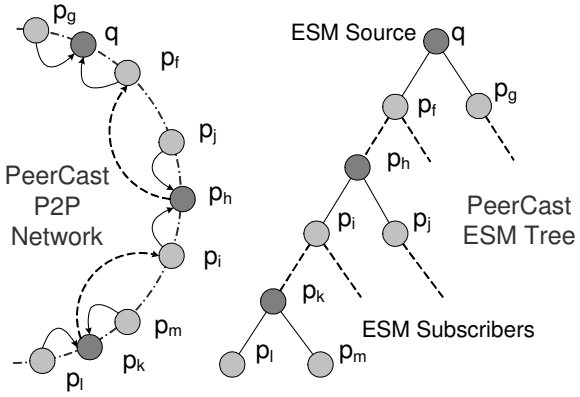

Figure 5: Constructing efficient multicast trees through landmark signature and neighbor lookup techniques

Figure 5 illustrates the neighbor lookup technique. Before forwarding the subscription request to the next hop peer that satisfies the prefix matching, peer p_i first checks if its neighbor has already joined the multicast group. It finds that peer p_h is in the multicast tree and chooses it as its potential parent. Similarly, peer p_l subscribes to its physical network neighbor p_k.

Our landmarks signature scheme ensures that a node present in a peer's neighbor list is also close to the peer in the physical network. Thus, in the multicast tree constructed using neighbor lookup technique, end-hosts close to one another are grouped together. One peer in each group (the darker

nodes in Figure 5) serves as the parent of other peers (the lighter nodes in Figure 5), and forwards the multicast payloads to them. Because the unicast links among network neighbors usually have higher bandwidth and lower latency, we improve the efficiency of the multicast tree and reduce the number of IP packets traversals in the underlying network.

An important challenge for all schemes that use network proximity information for optimizing system performance is the constant transience of the underlying IP network links. The bandwidth of these links change, sometimes drastically. Thus, the landmark signature may no longer accurately reflect the current network conditions. PeerCast system incorporates two mechanisms to counter this challenge. First, the individual peers periodically monitor their bandwidth availabilities. If the bandwidth availability of a peer is considerably less than any of its children in the multicast tree, the peers undergo a local reorganization of the multicast tree, whereby the child node with higher bandwidth is promoted up the multicast tree. We call this mechanism virtual node promotion. Section 5 provides a detailed description of the this mechanism. Second, the peers also periodically probe the landmark nodes and recompute their landmark signatures. If a peer observes a significant change in its landmark signature, it rejoins the PeerCast P2P overlay with its new landmark signature. This would insert the peer into a different identifier segment which will ensure that the node will be placed closer to its physical network neighbors.

4 Reliability in the PeerCast System

In the context of application-level multicast systems, reliability has two related but distinct meanings. First, it can refer to the availability of multicast services in resilience to the volatile nature of end hosts. Alternatively it can also refer to the reliable delivery of multicast content from content provider to receivers via multiple hops of unicast IP links.

In designing PeerCast, our focus has been on ensuring high availability of multicast services by providing resilience to the end hosts dynamics. We believe that the reliable delivery of application content should be addressed by incorporating failure resilience and fault tolerance mechanisms both at the transportation layer and at the application layer. PeerCast is designed to be a middleware that uses the existing transport layer infrastructure for supporting various applications require a content

distribution spanning tree composing of IP unicast links.

In the PeerCast system, a failure is represented by the interruption in the multicast service to the nodes. It is typically caused by the exit of one or more peers in the system. Node exits can be of two types. A node might *gracefully depart* from the system, in which case the departing peer notifies the other peers about its departure. In contrast, a *failure* is a disconnection of a peer without notifying the system. This can happen due to a network problem, computer crash, or improper program termination. Failures are assumed to be detectable (a fail-stop assumption), and are captured by the PeerCast P2P protocols neighbor list polling mechanisms. The nodes in the P2P network periodically poll the peers in their neighbor list. If a peer does not answer to two consecutive polling messages, it is presumed to have failed. The polling periods are carefully adjusted to ensure that a peer does not receive all the poll messages at the same time. Irrespective of how a node exits, the mechanisms to handle its exit are similar.

One possible way to handle node exits would be to require the peers whose multicast service has been disrupted to re-subscribe to their respective multicast service (Castro et al., 2002). This is in fact a fall-back option in PeerCast. However, this approach has a major drawback; each node exit would trigger re-subscriptions of all its children. This imposes significant overheads on the system. Instead, the PeerCast system incorporates a failure resilience mechanism that is based on the principle of service replication.

4.1 Service Replication Scheme

The main idea is to consistently replicate the ESM group information of a peer p_i on a few other peers, so that one of these peers can take over the functionality of sending data to the downstream nodes if the peer p_i were to exit the network. Our service replication scheme involves two phases. The first phase is right after the ESM group information is established on a peer. Replicas of the ESM group information are installed on a selection of peers. After replicas are in place, the second phase keeps those replicas consistent as end-system nodes join or leave the ESM group.

In our scheme, the group information of a peer p_i participating in a multicast group G is replicated on a set of peers denoted as $ReplicationList(G, p_i)$. We refer this set as the replication list of peer

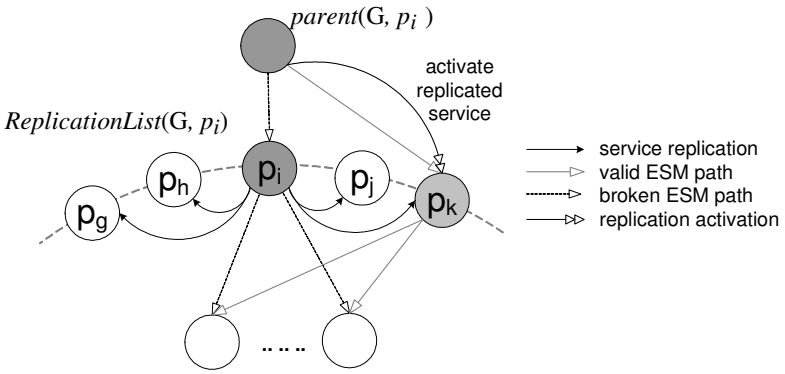

Figure 6: Multicast Service Replication $-$ p_i replicates its group information on four other nodes. The parent node activates p_k when p_i fails

p_i with respect to the group G. The size of the replication list is called the replication factor (r_f), and is a tunable system parameter. To localize the operations on the replication list, we construct the replication list from the neighboring peers of node p, which implies that $r_f \leq 2 \times r$.

For each ESM group G in which a peer p_i is actively participating, p_i forwards the replication list $ReplicationList(G, p_i)$ to its parent peer $parent(G, p_i)$ in group G. Once p_i departs from group G, its parent peer $parent(G, p_i)$ will use $ReplicationList(G, p_i)$ to identify another peer p_k to take over the ESM multicast forwarding functionalities of p_i. p_k uses the group information that p_i installed on it to carry out the ESM forwarding for group g. We say that p_k is *activated* in this scenario. Upon activation p_k uses its neighbor list $NeighborList(q, j)$ to setup the new replication list ($ReplicationList(G, p_k)$), and then forwards it to the parent node. Figure 6 illustrates the multicast service replication scheme when r_f is set to 4.

Observe that the subscription process of a peer can be accelerated by using the combination of the neighbor lookup technique and the service replica information that may have been installed in its neighborhood.

Replica Management: Since the overlay network is dynamic, it is necessary to continuously monitor and manage the replicas in order to maintain the replication factor at the specified level, and to ensure the consistency of the replica. We outline the process of maintaining the replicas when end-systems join or leave the PeerCast system. For the purpose of brevity, we assume that the replication factor r_f is equal to $2r$. However, our scheme can be easily adapted to the scenarios where r_f is less than $2r$.

Through the PeerCast P2P protocol a peer p_i can detect the entry and exit of any other peer in its $NeighborList$. Once such an event happens, an upcall is triggered by the P2P management protocol. The replica management protocol queries the peers in $NeighborList(p_i)$ and updates p_i's replication list appropriately.

The precise set of update operations that occur depends upon the nature of the event that is detected. However, the objective is to ensure the consistency of the replication list and maintain the replication factor at the desired level. For example, when a node p_i detects the exit of node p_j, it removes p_j from its replication list (if p_j is present). It may also find an alternate node to host its replica. The updated replication list is sent to p_i's parent node. Further, p_i remove any replica of p_j that it might be hosting. Analogous operations occur when a new node enters the PeerCast system. The exact protocols are discussed in detail in the technical report (Zhang et al., 2004).

Updating Replicas. As end-system nodes subscribe or unsubscribe from ESM groups, their subscription or un-subscription requests will be propagated through the ESM tree and change the group information on some peers. Once the group information of group G is changed on peer p_i, p_i sends its group information to other peers in its replication list.

Replica Selection Policy: As mentioned earlier, when a node p_i exits the PeerCast network, its parent in the multicast tree activates one of the nodes in p_i's replication list asking it to handle the forwarding functionalities of p_i. The choice of the peer to be activated depends upon two factors, namely *peer load factor* and *replication distance factor*. Intuitively, the peer load factor quantifies the willingness of neighboring peer p_r to accept one more multicast forwarding workload considering its current load. The replication distance factor, on the other hand, is a measure of the network proximity of the peer p_r to the failed peer p_i. A utility function called $ReplicaSuitability$ combines these two factors into a single value (Zhang et al., 2004). The parent of the failed peer p_i activates the peer in p_i's replica list that has the highest $ReplicaSuitability$ value.

5 Load Balancing in PeerCast

We now describe our techniques to address the third major limitation of the basic ESM scheme, namely the load imbalance due to node heterogeneity. Measurement studies (Saroiu et al., 2002) have shown

that end-hosts exhibit noticeable heterogeneity in large-scale P2P networks. For a distributed system in which end-hosts rely on one another to provide services like end-system multicast, balancing workloads among heterogeneous end-hosts is vital for utilizing the full system capacity and for providing efficient services. One way to address the node heterogeneity problem is to place end-hosts of a P2P network into different service layers depending upon their capabilities. A number of systems have adopted this approach to achieve better performance. For example, KaZaA (Kazaa, 2002) and Gnutella v.0.6 (Gnutella, 2003) have the notions of *super node* and *ultra peer* respectively. Similar strategies have also been proposed for structured P2P networks (Xu et al., 2003a; Zhao et al., 2002b).

However, the above approach has a significant drawback: The predetermined hierarchical system architecture of such schemes introduce vulnerabilities into the overlay network. Usually, nodes at higher levels of the hierarchy do not use the lower level nodes to relay traffic to one another. Rather, the supernodes interact directly with one another. When these supernodes drop out of the network, the overlay network is likely to be partitioned into a number of disconnected smaller networks, thereby leading to large-scale service disruptions.

The PeerCast system adopts an alternative approach. Our approach is based on the concept of *virtual nodes*. A virtual node is a conceptual entity that encapsulates all the functionalities of an individual peer. Each virtual node has its own identifier in the PeerCast system. An actual peer in the PeerCast system is assigned one or more virtual nodes based upon the resource availability at the peer. In other words, an actual peer in our system hosts multiple identifiers and is responsible for performing the functionalities of the corresponding virtual nodes. Thus, our scheme implicitly assigns more workloads to powerful nodes.

The resource availability of a peer p_i, represented as $RA(p_i)$, denotes the resources available for the PeerCast application at p_i. In the current design of the PeerCast system, the resource availability of a peer p_i is computed as the weighted sum of three components, namely bandwidth availability, CPU availability, and memory availability. Mathematically, $RA(p_i) = W_{BA} \times BA(p_i) + W_{CA} \times CA(p_i) + W_{MA} \times MA(p_i)$ where $BA(p_i)$, $CA(p_i)$ and $MA(p_i)$ denote the bandwidth availability, CPU availability and the memory availability at p_i respectively and W_{BA}, W_{CA}, and W_{MA} denotes the corresponding weights such that $W_{BA} + W_{CA} + W_{MA} = 1$. The bandwidth, CPU, and memory

availabilities of a peer p_i may be set by the end-user, or they may be limited by other applications running on p_i. In either case, the resource availabilities are estimated through techniques similar to those proposed by Gedik and Liu (Gedik and Liu, 2002). The weight of a particular parameter signifies its importance to the overall performance of the application executing on the PeerCast system. Weight assignments can be based upon the specification of the publisher of the contents, and will be passed down to subscribers when they join the multicast tree. For example, when PeerCast is handling an application demanding more bandwidth, the content publisher will set W_{BA} to a high value.

The PeerCast system incorporates three load-balancing operations:

Generate Virtual Nodes. Each end-host joins the PeerCast ESM overlay with a set of identifiers generated with different random seeds. Each identifier represents a virtual node with one unit of resource. Each virtual node maintains its own overlay state information like routing table and neighbor list.

Subscribe with Virtual Nodes. An end-host subscribes to a multicast group by starting the subscription process at one of its virtual nodes with an identifier numerically closest to the group identifier (g_{id}) of the service. Statistically, a more powerful end-host should have higher probability to own an identifier that is closer to any service identifier.

Virtual Nodes Promotion Our virtual node subscription scheme assigns shorter multicast path to more powerful end-hosts with high probability. However, because our scheme uses randomly generated leading digits to control the distribution of identifiers, there is still a non-negligible probability of a weak end-host owning an identifier that is numerically close to the multicast root. Such a node would be placed closer to the root of the multicast tree. Hence, it would have to serve a large number of subscribers and can become a bottleneck.

In order to mitigate this problem, we design a technique called *virtual node promotion*. In this technique, each node in the multicast tree periodically probes its child nodes. It chooses the child that has the most available resources (child with the maximum value of RA) as its *potential replacement*. Whenever a node detects that its potential replacement has more resources than itself, it informs the potential replacement to subscribe to its parent and informs its children to subscribe to the potential

replacement. On receiving the promotion notification, the potential replacement will inform its children to subscribe to its current parent. Thus, end-hosts contributing more resources will be gradually *promoted* towards the root of the ESM tree, and they obtain better multicast service than other end-hosts. Periodic monitoring of the various resources enables the end-hosts to respond to changes in their resource availabilities by initiating a local reorganization of the multicast tree. Note that the whole process is transparent to the end users.

One of the concerns with the virtual node promotion technique is whether it would adversely impact the physical network locality property of the multicast tree, which would in-turn impact the multicast latency. However, note that our node promotion technique guarantees that the nodes are promoted within a sub-tree of the multicast tree. Our subscription management and multicast tree maintenance protocols ensure that all the nodes in the same sub-tree share the same node identifier prefix with the root node. Since, our landmark-based clustering mechanism provides the property that the nodes sharing the same identifier prefix are more likely to be close to each other in the IP network, the powerful nodes are well contained within certain network proximity, even after being promoted.

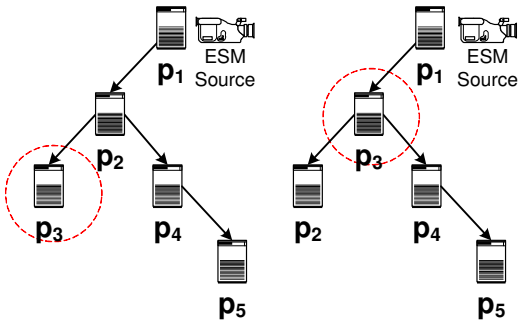

Figure 7: Virtual node promotion technique – Node p_3 is promoted by one level since it has more resources than p_2.

Figure 7 illustrates the virtual node promotion technique. Node p_2 detects that p_3 has more resources than itself, and it initiates the promotion of p_3. p_2 then becomes a child of p_3.

6 Experimental Results

We have developed a simulator that implements the mechanisms presented in this paper. We evaluate the PeerCast system with respect to its three unique features. We used the Transit-Stub model from the GT-ITM package (Zegura et al., 1996) to generate a set of network topologies. Each topology

consists of 5150 routers and has the same latency setup as those used in the experimental study of SCRIBE (Castro et al., 2002). Concretely, the link latency values are uniformly distributed. The distribution ranges vary according to the type of the links − (15ms, 25ms) for intra-transit domain links, (3ms, 7ms) for transit-stub domain links and (1ms, 3ms) for intra-stub domain links. We used the routing weights generated by the GT-ITM package to simulate the IP unicast routing. IP multicast systems are simulated by merging the unicast routes into shortest path trees.

6.1 Clustering by Network Proximity

The first set of experiments examines the precision of the landmark signature technique in clustering the end-hosts by their network proximity. We use two metrics for this purpose. *Clustering precision* is defined as the percentage of peers that have physical network neighbors in their local P2P neighbor lists. *Clustering accuracy* is defined as the average number of physical network neighbors in each peer's P2P neighbor list. We simulate the P2P networks with 1×10^4 to 9×10^4 peers, and set the neighbor list size parameter r to be 8, 12, and 16. The splice offset is set to 1.

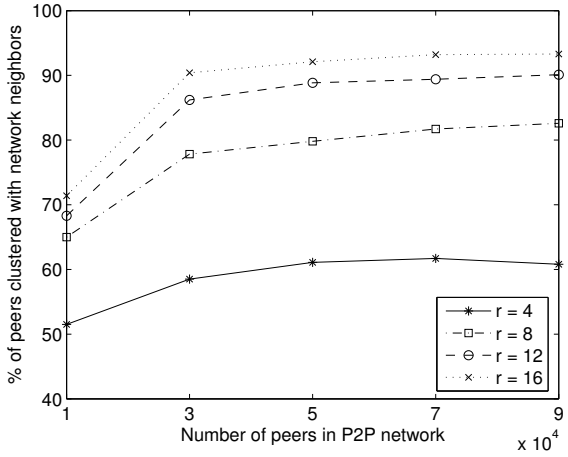

Figure 8: Clustering precision the landmark signature technique when *splice offset* = 1

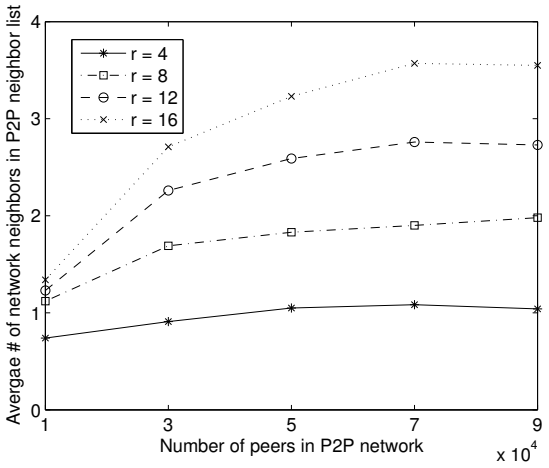

Figure 9: Clustering accuracy of the landmark signature technique when *splice offset* = 1

Figure 8 and Figure 9 plot the experiment results. The results reveal a few interesting facts. First, increasing the value of r increases the probability that peers find their physical network neighbors in their local P2P neighbor list. Second, the landmark signature scheme can capture the network proximity information with satisfactory precision. Around 95% of the peers have one or more network neighbors in their local neighbor list, when r is set to 16. Third, larger peer populations increase the precision of clustering because more peers within the same network region join the PeerCast ESM

overlay.

6.2 Efficiency Multicasting in PeerCast ESM Overlay

In the next set of experiments, we evaluate the PeerCast system with respect to multicast efficiency. We study three different flavors of the PeerCast ESM system: one without the landmark signature technique, one with only the landmark signature technique, and one with both the landmark signature technique and the neighbor lookup technique. We simulate a P2P network with 5×10^4 peers. The number of peers joining the multicast group varies from 1×10^4 to 4×10^4. We set the value of neighbor list parameter r to 8 and use 16 landmark points. The splice offset is set to 0, 1, and 2.

The number of landmarks impacts the accuracy of the PeerCast system. In general, the more the landmarks, the better the accuracy. However, as several researchers have reported, when the number of landmarks increases beyond a certain value, the corresponding improvements in accuracy are minimal (Ng2003, ; Ramaswamy et al., 2006). Our decision to use 16 landmarks for the experiments is supported by our previous work as well as the research results reported by other researchers. In the context of our work on cooperative cache group formation for web content delivery, we have shown that the performance of network proximity-based overlay networks stabilize when the number of landmarks is around 15, and further increasing the landmarks set would result in minor improvements of accuracy (Ramaswamy et al., 2006). Ng and Zhang (Ng2003,) report 15 landmarks are sufficient for their GNP algorithms to predict the network distances among hosts with very high accuracy. Further, we also note that the overheads of having 16 landmarks are not very high; each node in the system needs to send out 16 probe messages and sort a 16-elements vector, which will not consume too much system resources.

6.2.1 Delay Penalty

We first compare the message delivery latency of IP multicast and PeerCast. Generally, ESM systems have higher multicast message delivery latencies than the equivalent IP multicast systems. This is because of multi-hop message replication and unicast message forwarding. We use the *relative delay penalty* parameter to compare the delivery latencies of the different ESM schemes. The relative delay penalty of an ESM scheme is defined as the ratio of the mean delay incurred by the ESM scheme to

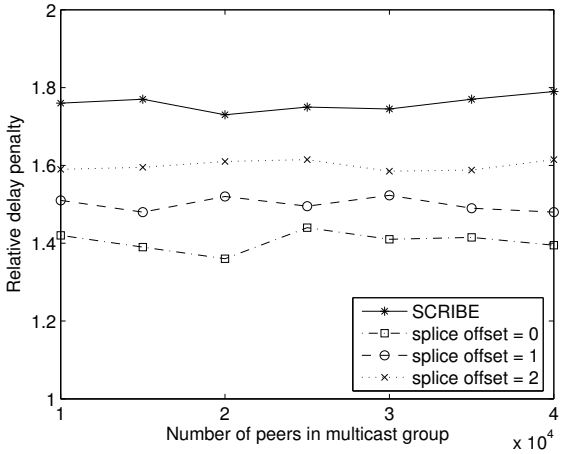

Figure 10: Relative delay penalty when only the landmark signature technique is employed

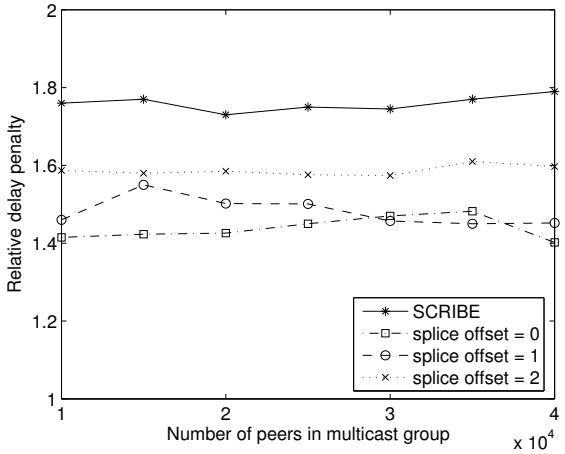

Figure 11: Relative delay penalty when both landmark signature and neighbor lookup techniques are employed

the corresponding mean IP multicast delay.

In the first experiment of this set (Figure 10), we exclusively study the effects of landmark signature technique on the relative delay penalty. Hence, the neighbor lookup mechanism is disabled. The results show that our landmarks signature technique provides significant benefits with respect to the relative delay penalties. The landmark signature technique combined with our ESM management protocol constructs multicast trees such that the nodes that are closer to the root of a tree are likely to be its neighbors in the underlying network. Thus, our scheme incurs significantly less delay penalty. However, when the splice offset is set to higher values, it increases the randomness of the identifier distribution in PeerCast system, which in-turn impacts the effectiveness of the landmarks signature technique.

Our neighbor lookup scheme requires the peers participating in the subscription forwarding process to first contact their neighbors before routing the request according to the progressive matching criterion. The next experiment (Figure 11) examines whether this has any serious impact on the relative delay penalties of the PeerCast system. The results show that neighbor lookup technique has little effect on the relative delay penalty. Because of the landmark signature technique, the nodes that are grouped by the neighbor lookup mechanism are also neighbors in the underlying physical network. Hence, the communications among them are faster and add little to the overall multicast delay.

6.2.2 Link Stress

ESM systems always generate more IP messages than the corresponding IP multicast systems. *link stress* is the ratio between the number of IP messages generated by an ESM multicast tree and the number of IP messages generated by the equivalent IP multicast tree. Smaller the link stress values, better is the ESM scheme.

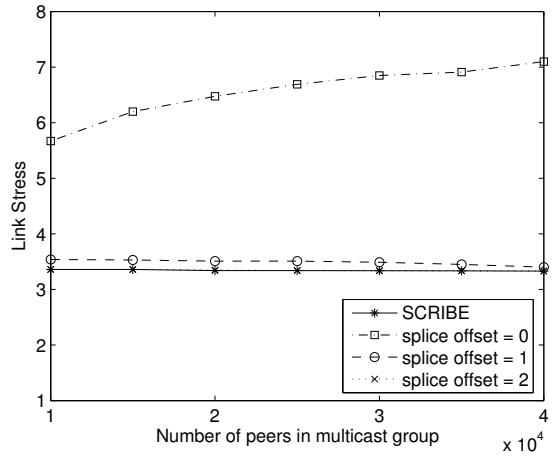

Figure 12: Link stress when only the landmark signature technique is employed

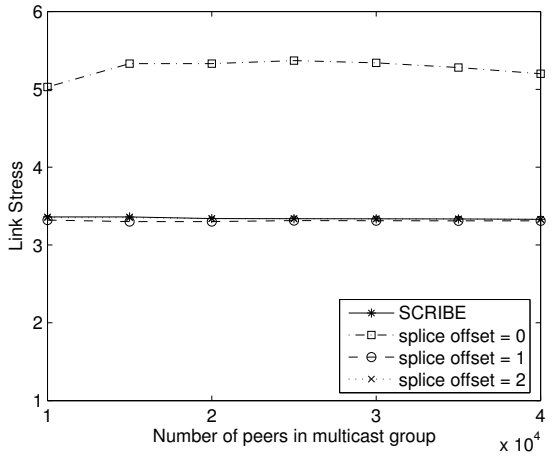

Figure 13: Link stress when both landmark signature and neighbor lookup techniques are employed

Figure 12 and Figure 13 indicate the node stress values of the PeerCast system when the neighbor lookup is disabled and enabled respectively. In both cases the performance of the PeerCast system suffers when the splice offset is set to zero. This is because, when the splice offset is zero, the landmark signature is inserted at the very beginning of each peer identifier, thereby eliminating the randomness in the peer identifier distributions. Thus, peers that are not neighbors in the underlying physical network seldom share the same identifier prefix. In this scenario, the latency of the first hop of the lookup request is very high if the request initiating peer is not a physical network neighbor of the multicast source. When we introduce more randomness into the identifier distribution by increasing the splice offset value, the peers from the same network region get distributed to different segments on the P2P identifier circle. Thus, the link stress drops close to the level of the SCRIBE system (and the basic PeerCast system) when the neighbor lookup optimization is disabled. Enabling the neighbor lookup functionality increases multicast forwarding among physical network neighbors, thereby resulting in lower link stress values.

6.2.3 Node Stress

In the next set of experiments, we study the loads placed on the end-hosts by the ESM schemes. The nodes in a multicast tree perform a variety of functions, including replication and forwarding of multicast messages, maintaining multicast group information and handling failures of children nodes. The cumulative load that a peer encounters due to these functionalities varies directly with the numbers of children it has in each of the multicast trees in which it is participating. Accordingly, we quantify the multicast loads on peers using the *node stress* parameter, which is defined as the average number of children that non-leaf end-hosts posses in the multicast trees in which they participate.

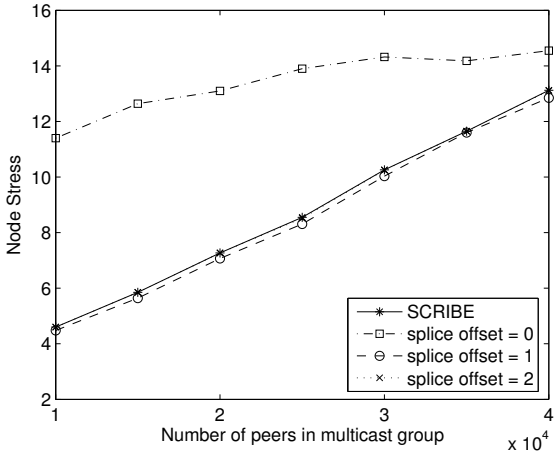

Figure 14: Node stress when only the landmark signature technique is employed

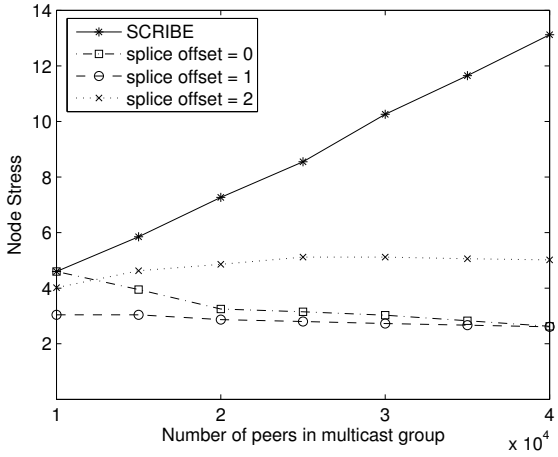

Figure 15: Node stress when both landmark signature and neighbor lookup techniques are employed

Figure 14 indicates the node stress values when the neighbor lookup functionality is disabled. The results can be explained as follows. When we splice the landmark signature at the very beginning of each identifier (i.e., the splice offset is 0), it reduces reduce the number of peers that share the same identifier prefix with the multicast source. In this scenario, subscription requests are more likely to be forwarded through fewer peers in the P2P network, which incur higher node stress values. When the splice offset is increased, the randomness of the peer identifier distribution increases as well. Thus, at splice offset values of 1 or 2, the node stress values of the PeerCast system are very close to those of the SCRIBE system.

The neighbor lookup technique considerably reduces the node stress of the PeerCast system (see Figure 15). This is because, the neighbor lookup technique allows the peers to leverage upon their physical network neighbors for the multicast services they need.

6.3 Load Balancing among Heterogeneous End-Hosts

Next, we evaluate the effectiveness of our virtual node-based load balancing technique. For this purpose, we assign different capacities to different sets of end-hosts. Specifically, we assume that 20% end-hosts possess 8 units of capacity, 30% end-hosts have 4 units of capacity, and the rest 50% end-hosts own only single unit capacity. We measure the average loads on these different sets of end-hosts. Further, we are also interested in the quality of multicast services received by end-hosts donating different amount of resources. We use the relative delay penalty metric for quantifying the quality of multicast service.

We have performed two experiments for evaluating the virtual node technique. In the first experiment, we study the performance of the scheme when the total number of peers in the system varies from 50 to 50,000. This experiment assumes that all the peers participate in a single multicast service. In the second experiment, we consider a scenario wherein there are multiple multicast services on top of an overlay containing $50,000$ peers. We vary the size of each multicast group from 50 peers to $50,000$ peers.

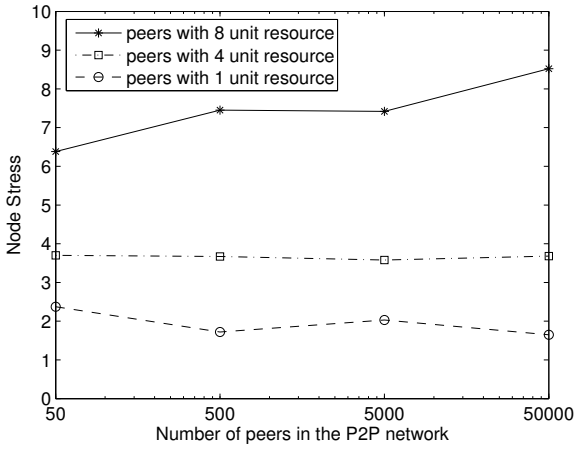

Figure 16: Variations of average node stress with respect to the total number of nodes in the system

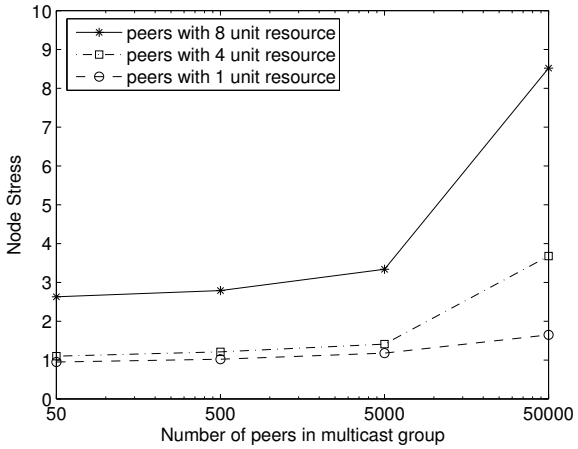

Figure 17: Variations of average node stress with respect to the size of the multicast groups

Figure 16 and Figure 17 show the results of the two experiments on the average node stress values of the three peer categories. The results show that the average node stress values of the three peer categories are commensurate with their respective capacities. In Figure 17, the effects of the virtual node-based load balancing scheme are not very pronounced when the number of peers in each multicast group is below 5000. This is because under these conditions, the system itself is lightly loaded.

However, when the P2P overlay is heavily loaded, the virtual node technique plays a vital role in balancing the multicast workloads.

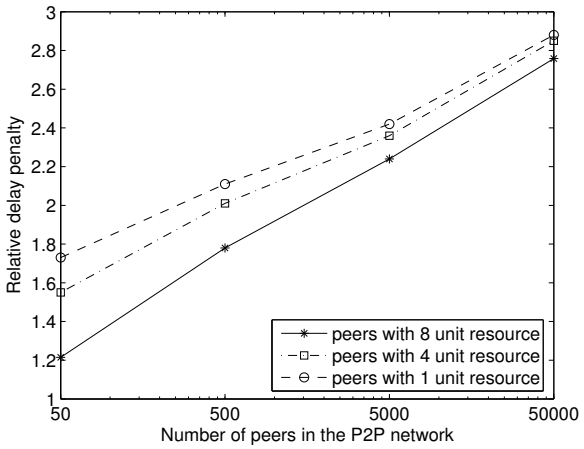

Figure 18: Variations of relative delay penalty with respect to the total number of nodes in the system

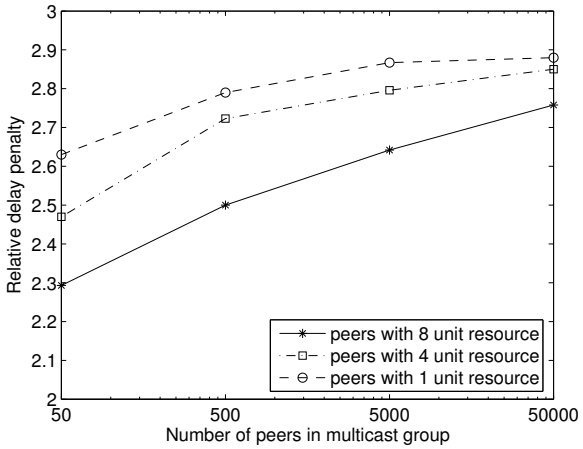

Figure 19: Variations of relative delay penalty with respect to the size of the multicast groups

One of the unique features of PeerCast is that the end-hosts that contribute more resources will be placed closer to the multicast root peer. Thus, peers contributing more resources would be *rewarded* with lower multicast latencies. Figure 18 and Figure 19 indicate the relative delay penalties of the three peer categories. The results show that the peers contributing more resources experience lower latencies.

6.4 Failure Resilience

The next set of experiments evaluates our failure resilience mechanism. In these experiments, we assume that the overlay is dynamic with peers entering and exiting system at arbitrary points in time. Concretely, we assume that the duration for which an arbitrary peer stays in the system (henceforth called service time) and the duration for which the peer is out of the network (henceforth called the recovery time) are both exponentially distributed. In these experiments, we vary the mean service time (st), the mean recovery time (Δt_r) and the replication factor (r_f), and measure the number of *un-recoverable failures* (these are the failures that cannot be re-covered by replica-activation and require re-subscriptions from down-stream nodes). Recall that the objective of our failure resilience technique is to minimize un-recoverable failures, as they cause service disruptions and impose heavy message overheads.

The graphs in Figures 20, 21, and 22 indicate the total number of unrecoverable failures that

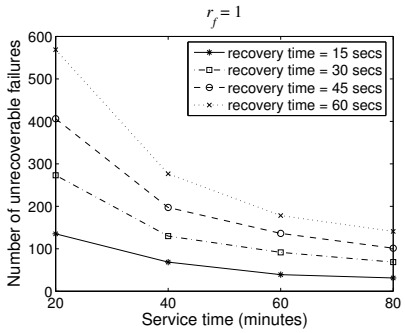

Figure 20: Number of unrecoverable failures when replication factor is set to 1

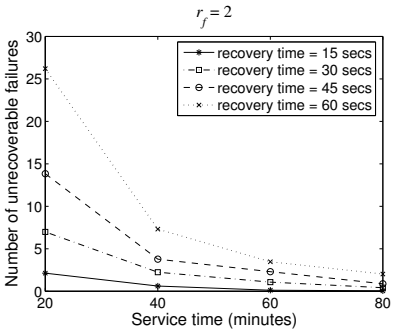

Figure 21: Number of unrecoverable failures when replication factor is set to 2

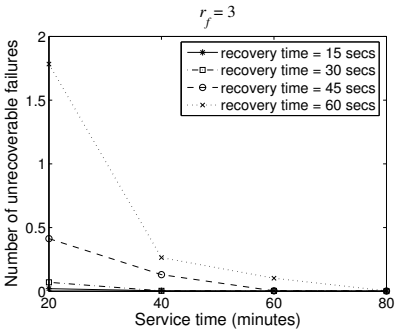

Figure 22: Number of unrecoverable failures when replication factor is set to 3

occurred during the simulation when st, Δt_r, and r_f are set to various values. These graphs show that the number of unrecoverable failures is smaller when the replication factor is larger. Further, the number of unrecoverable failures varies directly with the mean service time and inversely with the mean recovery time. The number of unrecoverable failure becomes negligible when r_f is set to 3. These experiments demonstrate that our dynamic replication technique is able to achieve high reliability even at moderate replication factor values.

6.4.1 Service Recovery Overhead

In the final set of experiments, we study the benefits of the dynamic replication scheme in terms of the overheads involved in restoring multicast services that are interrupted because of node failures. We quantify the service recovery overheads by the total number of messages exchanged during the service restoration process. An unrecoverable failure of a peer causes its downstream peers to re-subscribe (i.e., the downstream nodes try to restore the interrupted multicast services by themselves). On the other hand, if a peer's failure is recoverable, two messages are sufficient to restore the service (one message for reporting the failure and another for activating the service replica).

Figure 23 shows the total number of service recovery messages circulated when the r_f is set to 1 and st is varied from 20 minutes to 80 minutes. The reported message numbers include the messages generated for replica activation (when the failure is recoverable) and the messages for service re-subscription (when the failure is unrecoverable). We observe that the results in Figure 23 show similar trends to those in Figure 20. This is because each unrecoverable failure causes several re-subscription messages from the downstream peers. Figure 24 shows the contribution of replica activation messages

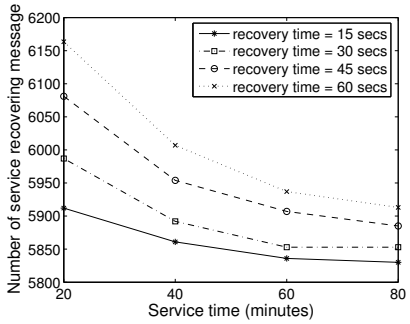

Figure 23: Number of service recovery messages when the dynamic replication is employed ($r_f = 1$)

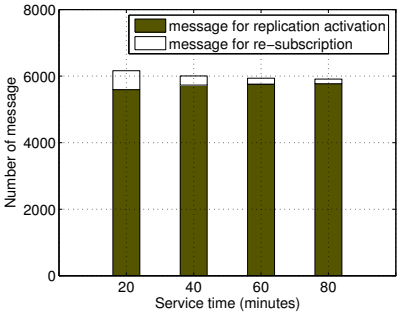

Figure 24: Composition of service recovery traffic ($r_f = 1$, $\Delta t_r = 60$ sec.)

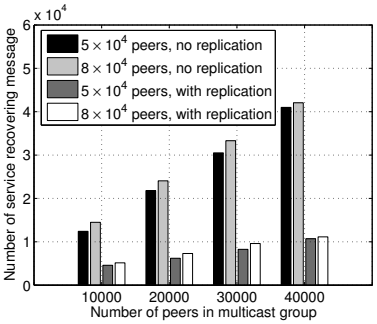

Figure 25: Benefits of the dynamic replication scheme with respect to service recovery overheads($\Delta t_r = 15$ secs, $st = 20$ mins, r_f = 1 for the replication scheme)

and re-subscription messages towards the total message load. Since most of the interrupted services are restored by replica activation, this component dominates the total message load.

To evaluate the effect of the replication scheme on reducing the service recovery overhead, we compared the number of messages incurred by the replication scheme to the number of messages generated when there is no service replication. We create multicast groups with $1 * 10^4 \sim 4 * 10^4$ peers on top of P2P networks containing $5 * 10^4$ and $8 * 10^4$ peers. The replication scheme is setup with $r_f = 1$ and the peer service times follow the exponential distribution with mean of 20 minutes. The results are indicated in Figure 25. The results show that the service replication scheme provides significant benefits in terms of the message overheads.

7 Related Work

End-system multicast has received considerable research attention in the past few years. Yeo et al. (Yeo et al., 2004) classified existing application level multicast systems into four categories: *(1)* tree-based (e.g. Overcast system), *(2)* mesh and tree-based (e.g. Narada, Scattercast, Yoid), *(3)* embedded structure-based, and *(4)* topology-agnostic (e.g. CAN-multicast). In the following discussion, we compare our system to representative systems from all four categories. Chawathe (Chawathe, 2000) and Chu et al (Chu et al., 2000) propose a two-step protocol to build multicast trees. For example, the Narada (Chu et al., 2000) first generates a mesh network among all members and then uses the distance vectors to generate the multicast tree. However, because of their high maintenance overhead,

such schemes are only suitable for small networks. The Overcast protocol (Jannotti et al., 2000) presents a distributed strategy to organize a set of proxies (Overcast nodes) into a distribution tree, with the multicast source as the root. It uses end-to-end measurements to optimize bandwidth among the root and group members. This scheme is in some ways similar to the Yoid scheme (Francis, 1999). The PeerCast system differs from these schemes in that we design completely distributed protocols to construct and maintain the ESM overlay. The efficiency of the PeerCast ESM overlay is achieved by leveraging the network proximity information provided by the landmarks signature scheme.

PeerCast is similar to ESM protocols like NICE (Banerjee et al., 2002), CAN-multicast (Ratnasamy et al., 2001b), and SCRIBE (Castro et al., 2002), which use implicit approaches to create ESM overlays. In those systems, a control topology is built by using either on-the-shelf P2P protocols (Ratnasamy et al., 2001a; Rowstron and Druschel, 2001; Zhao et al., 2002a) or specially designed protocol (Banerjee et al., 2002).

The Content Addressable Network (CAN) (Ratnasamy et al., 2001a) organizes nodes in a d-dimensional Cartesian space, and it uses landmark technique to partition the Cartesian space into various sized bins. However, as Xu et al. (Xu et al., 2003b) demonstrate, the overlay structure of CAN is constrained by the underlying network topology (Ratnasamy et al., 2002). Further, the technique used by CAN can destroy the randomness of the nodes distribution, thereby inducing high maintenance costs. The Pastry (Rowstron and Druschel, 2001) system exploits topology information in overlay routing using geographic layout, proximity routing, and proximity neighbor selection. It assumes triangle inequality in the network topology and using expanding-ring search to choose the physically closest node at node joining. SCRIBE (Castro et al., 2002) is an ESM protocol built on top of Pastry. Our experiments show that the basic PeerCast system yields similar performance as that of SCRIBE. However, due to *logarithmical deterioration of routing* (Xu et al., 2003b), the stretch of SCRIBE's multicast forwarding link increases exponentially from the subscriber to the multicast source. The randomness of node identifier distribution of Pastry precludes the usage of optimization techniques such as neighbor lookup of PeerCast, which helps PeerCast to achieve better utilization of network resources.

In short, the approach taken by PeerCast differs from the existing systems in two aspects. First,

the efficiency of PeerCast ESM overlay is the result of the synergy of optimization techniques at different system levels. We use the landmark signature technique to improve the routing efficiency at the P2P network level, whereas the neighbor lookup technique optimizes the multicast overlay at the ESM overlay level. Second, PeerCast presents a scalable solution for end-system multicast in a heterogeneous environment by using the virtual node scheme.

8 Conclusion

Designing and implementing an efficient and failure-resilient end system multicast service on top of highly dynamic overlay networks poses several research challenges. In this paper, we have presented an efficient and self-configurable end system multicast system, called PeerCast. Our approach has three unique features compared to existing approaches to application-level multicast systems.

First, we use the landmark signature technique to cluster end-hosts in the PeerCast ESM overlay network, aiming at exploiting the network proximity of end-hosts for efficient multicast information dissemination across wide area networks. Second, we propose a capacity-aware overlay construction technique based on the concept of virtual node to balance the multicast workload among heterogeneous end-hosts. Third, we develop a dynamic passive replication scheme to provide reliable ESM services in an environment of inherently unreliable peers. An analytical model is presented to discuss its fault tolerance properties.

We evaluate PeerCast using simulations of large scale networks. The experimental results indicate that PeerCast can provide efficient multicast services over large-scale network of heterogeneous end-system nodes, with reasonable link stress and good load distribution.

References

Ng, E. GNP Software, http://www-2.cs.cmu.edu/ eugeneng/research/gnp/software.html; 2003.

Banerjee, S., Bhattacharjee, B., and Kommareddy, C. Scalable Application Layer Multicast. In *Proceedings of ACM SIGCOMM*; 2002.

Castro, M., Druschel, P., Kermarrec, A., and Rowstron, A. SCRIBE: A large-scale and decentralized application-level multicast infrastructure. *IEEE Journal on Selected Areas in communications (JSAC)*; 2002.

Chawathe, Y. Scattercast: An Architecture for Internet Broadcast Distribution as an Infrastructure Service. *Ph.D. thesis*, University of California, Berkeley; 2000.

Chu, Y.-H., Rao, S. G., and Zhang, H. A case for end system multicast. In *Proceedings of ACM SIGMETRICS*; 2000.

Dabek, F., Cox, R., Kaashoek, F., and Morris, R. Vivaldi: A Decentralized Network Coordinate System. In *Proceedings of ACM SIGCOMM*; 2004.
Francis, P. Yoid: Extending the Multicast Internet Architecture. White paper. http://www.aciri.org/yoid; 1999.
Gedik, B. and Liu, L. PeerCQ: A Scalable and Self-configurable Peer-to-Peer Information Monitoring System. *Technical Report GIT-CC-02-32, Georgia Institute of Technology; 2002.*
Gnutella, *http://rfc-gnutella.sourceforge.net/*
Jannotti, J., Gifford, D. K., Johnson, K. L., Kaashoek, M. F., and O'Toole, Jr., J. W. Overcast: Reliable Multicasting with an Overlay Network. In *Proceedings of OSDI*; 2000.
KaZaA Media Desktop, *http://www.kazaa.com*
Pendarakis, D., Shi, S., Verma, D., and Waldvogel, M. ALMI: An Application Level Multicast. In *Proceedings of the USENIX Symposium on Internet Technologies and Systems (USITS)*; 2001.
Ramaswamy, L., Liu, L., and Zhang, J. Efficient Formation of Edge Cache Groups for Dynamic Content Delivery. in *Proceedings of ICDCS*; 2006.
Ratnasamy, S., Francis, P., Handley, M., Karp, R., and Shenker, S. A Scalable Content Addressable Network. In *Proceedings of ACM SIGCOMM*; 2001
Ratnasamy, S., Handley, M., Karp, R., and Shenker, S. Application-Level Multicast Using Content-Addressable Networks. *Lecture Notes in Computer Science* 2233; 2001.
Ratnasamy, S., Handley, M., Karp, R., and Shenker, S. Topologically-aware overlay construction and server selection. In *Proceedings of IEEE INFOCOM*; 2002.
Rowstron, A. and Druschel, P. Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems. *Lecture Notes in Computer Science* 2218; 2001.
Saroiu, S., Gummadi, P. K., and Gribble, S. D. A Measurement Study of Peer-to-Peer File Sharing Systems. In *Proceedings of MMCN*; 2002.
Stoica, I., Morris, R., Karger, D., Kaashoek, F., and Balakrishnan, H. Chord: A Scalable Peer-To-Peer Lookup Service for Internet Applications. In *Proceedings of ACM SIGCOMM Conference*; 2001.
Tang, L. and Crovella, M. Virtual landmarks for the Internet. In *Proceedings of the Internet Measurement Conference*; 2003.
Xu, Z., Mahalingam, M., and Karlsson, M. Turning Heterogeneity into an Advantage in Overlay Routing. In *Proceedings of INFOCOM*; 2003.
Xu, Z., Tang, C., and Zhang, Z. Building topology-aware overlays using global soft-state. In *Proceedings of ICDCS*; 2003.
Yeo, C., Lee, B., and Er, M. H. A survey of application level multicast technique *Computer Communication* 27(15); 2004.
Zegura, E. W., Calvert, K. L., and Bhattacharjee, S. How to Model an Internetwork. In *Proceedings of IEEE INFOCOM*; 1996.
Zhang, J., Liu, L., Pu, C., and Ammar, M. Reliable End System Multicasting with a Heterogeneous Overlay Network, *Technical Report GIT-CERCS-04-19, CERCS, Georgia Institute of Technology*; 2004.
Zhao, B., Kubiatowicz, J., and Joseph, A. Tapestry: An infrastructure for fault-tolerant wide-area location and routing, *Technical report, Univ. of California, Berkeley*; 2002.
Zhao, B. Y., Duan, Y., Huang, L., Joseph, A. D., and Kubiatowicz, J. D. Brocade: Landmark Routing on Overlay Networks. In *Proceedings of IPTPS*; 2002.