Island Multicast: Combining IP Multicast With Overlay Data Distribution

Xing Jin, Kan-Leung Cheng, and S.-H. Gary Chan, *Senior Member, IEEE*

***Abstract*—Traditional overlay protocols use unicast connections to form delivery trees. While it can achieve global multicast across the Internet, it is not as efficient as IP multicast. In this paper, we integrate IP multicast into overlay data distribution to improve delivery efficiency. We investigate island multicast where unicast connections are used to connect multicast domains and IP multicast is used within multicast domains. We first explore a centralized island multicast protocol (termed CIM), which relies on a central server to construct a delivery tree. We then study a distributed protocol (termed DIM), where hosts can distributedly join islands and form a delivery tree. We study the key issues in both protocols. We also discuss how to apply these protocols to media streaming applications.**

We have evaluated both protocols on Internet-like topologies. We have also implemented a prototype for CIM and tested it on PlanetLab. The results show that our approaches can significantly improve network performance as compared to pure overlay protocols. Our study shows that it is important to consider local multicast capability when designing overlay protocols.

***Index Terms*—Application-level multicast, IP multicast, island multicast, overlay data distribution, overlay multicast.**

I. Introduction

APPLICATIONS such as Internet-TV, multiparty conferencing, and software distribution require point-to-multipoint or multipoint-to-multipoint Internet communications. Traditionally, there are two techniques for these applications. The first one is IP multicast, where routers form a spanning tree to replicate and forward packets [1]. Despite of its proposal more than one decade ago, IP multicast has not been globally deployed yet. This is mainly because IP multicast requires multicast-capable routers, which need to maintain per group state for packet replication and are not scalable [2]. The second technique is overlay data distribution (also referred to as application-level multicast or overlay multicast) [3]–[5]. In overlay distribution, end hosts, instead of routers, are responsible for replicating and forwarding packets. Multicast is achieved via piece-wise unicast between hosts. This method does not need multicast-capable routers. On the other side of the coin, overlay distribution is not as efficient as IP multicast.

Manuscript received May 01, 2008; revised March 30, 2009. First published May 02, 2009; current version published July 17, 2009. This work was supported in part by the General Research Fund from the Research Grant Council of the Hong Kong Special Administrative Region, China (611107), and in part by the Hong Kong Innovation Technology Fund (ITS/013/08). The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Sheila S. Hemami.

X. Jin is with the Systems Technology Group, Oracle USA, Inc., Redwood Shores, CA 94065 USA (e-mail: xing.jin@oracle.com).

K.-L. Cheng is with the Department of Computer Science, University of Maryland, College Park, MD 20742 USA (e-mail: klcheng@cs.umd.edu).

S.-H. Gary Chan is with the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong (e-mail: gchan@cse.ust.hk).

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.2009.2021804

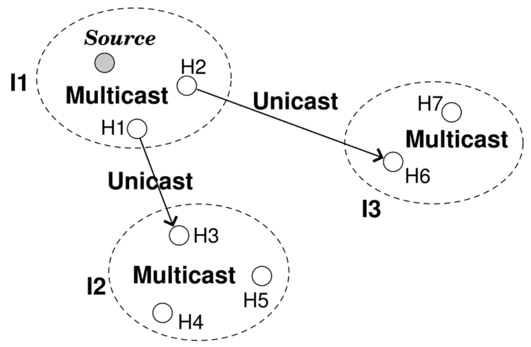


Fig. 1. Example of island multicast.

Recently, we note that IP multicast has achieved significant progress. In theory, multicast protocols with low deployment cost have been studied [6]. In practice, many local networks in today's Internet are already multicast-capable. These local multicast-capable domains, or so-called *islands*, are often interconnected by multicast-incapable or multicast-disabled routers. For example, in the Hong Kong area, Hong Kong Broadband Network Limited has deployed a broadband network to cover more than 2500 buildings. It offers digital television service to subscribers via IP multicast.

Clearly, it would be beneficial if overlay data distribution can make use of local multicast capability. We hence investigate island multicast that integrates IP multicast with overlay distribution. In island multicast, hosts within the same island use IP multicast for data distribution. Islands are then connected by unicast connections. We show an example in Fig. 1. In the figure, seven hosts (labeled as H_1 through H_7) are distributed in three islands, I_1, I_2, and I_3. The source lies in I_1 and all the hosts are receivers. The source transmits packets via IP multicast to receivers H_1 and H_2, which in turn forward the packets via unicast to islands I_2 and I_3, respectively. H_3, upon receiving a packet from H_1, relays it via IP multicast to H_4 and H_5. Similarly, H_6 multicasts its received packets to H_7.

In this example, H_1 and H_3 use a unicast connection between them to connect islands I_1 and I_2. They are called a pair of *bridge-nodes* for the two islands, and the path between them is called a *bridge*. Likewise, H_2 and H_6 are the bridge-nodes for I_1 and I_3. In addition, we call the bridge-node sending packets to the downstream island an *egress bridge-node*, or simply *egress*, e.g., H_1 and H_2. We call the bridge-node receiving data packets from the upstream island an *ingress bridge-node*, or simply *ingress*, e.g., H_3 and H_6.

In this paper, we propose and evaluate two island multicast protocols.

- *Centralized Island Multicast (CIM)*: Applications such as multiparty conferencing are characterized by small session size, many-to-many communication and high bandwidth requirement. We propose CIM for such applications. In CIM, a central server computes and maintains a delivery tree for data distribution. As the tree is computed based on global information, it achieves high delivery efficiency (as indicated by low stress in our simulations).
- *Distributed Island Multicast (DIM)*: Another class of applications such as news broadcasting and software distribution often have a large number of users. They require scalable and distributed solutions for data delivery. We hence propose DIM for them. In DIM, hosts in the same island elect a unique leader. All leaders form an overlay tree. Based on the leader tree, leaders distributedly select bridge-nodes for their islands and construct a delivery overlay.

We have simulated CIM and DIM on Internet-like topologies. We have also implemented CIM on PlanetLab [7]. Our results show that CIM can achieve low stress and is bandwidth-efficient. It also has low control overhead, because the server handles all tree operations and hosts do not need to maintain the tree. On the other hand, DIM can achieve low delay through its bridge-node selection mechanism. Both protocols can significantly reduce end-to-end delay and link stress as compared to pure overlay protocols.

The rest of the paper is organized as follows. In Section II, we briefly discuss related work. In Sections III and IV, we discuss the CIM and DIM protocols, respectively. In Section V, we investigate practical issues when applying CIM and DIM to media streaming applications. In Section VI, we present illustrative simulation results on Internet-like topologies. In Section VII, we present measurement results for CIM on PlanetLab. Finally, we conclude in Section VIII.

II. Related Work

Multicast backbone (MBone) is a virtual network developed for multicast related protocols [8]. Early works on combining IP multicast and unicast focus on setting up tunnels to MBone. For example, Thaler *et al.* use dedicated servers (e.g., gateways and relays) to set up tunnels [9]. But these tunneling mechanisms focus on the connection between a pair of hosts and do not consider data distribution among a set of session hosts. Different from them, CIM and DIM build delivery trees with low end-to-end delay and low bandwidth consumption for all hosts in the session.

Subset multicast (SM) also makes use of local multicast capability [10]. In SM, the source sends a copy of data to each of the multicast islands. The host in an island that receives data from the source then multicasts data within the island. Clearly, each island is connected to the source via unicast. This is not scalable to large sessions with many islands. Our DIM protocol is fully distributed and scalable. Even though CIM requires a central server, its source does not forward data to islands using a star-like topology. They are hence more scalable than SM.

In HMTP, each island has a unique leader (called a designated member) [11]. Designated members form an overlay tree for data distribution. Each designated member also IP multicasts data within its island. While this approach imposes the responsibilities of data receiving, data forwarding and island management on a single leader in each island, a leader has high nodal stress and heavy workload. Different from them, CIM and DIM distribute these responsibilities to different hosts. Each island in CIM or DIM has one ingress and some egress hosts. Ingress receives data from outside of the island and egresses forward data to other islands. The protocols hence achieve more balanced load between hosts. Furthermore, when islands are large (e.g., the whole MBone can be a single island), it is not efficient to represent each island by a single leader, where end-to-end delay depends on leader locations and selection of appropriate leaders is not easy. In CIM and DIM, we can select a pair of close hosts to connect two islands, which is more efficient and practical.

Also noticing the limitations of HMTP, the authors of HMTP further propose universal multicast (UM) to allow multiple designated members in one island [12]. In the approach, a designated member multicasts its *HeartBeat* messages with a certain time-to-live value so that the messages reach only a subset of the island members. Island members that do not receive *HeartBeat* messages then assume that their designated member has left and automatically elect a new designated member. In this way, an island can have multiple designated members. In our approaches, each island may also have multiple egress nodes. The major differences between UM and our approaches are: 1) In UM, designated members are elected based on their locations in an island. It works as if dividing a large island into multiple small islands and electing a designated member in each small island. But in our approaches, each island selects one or multiple egresses based on the locations of the island and the neighboring islands. Our method can directly reduce inter-island delay. Furthermore, the number of egress nodes in our protocols depends on the number of neighboring islands, instead of the size of its own island as in UM. Our approaches are hence more reasonable and efficient. 2) In UM, if multiple designated members are elected in an island, these designated members need to form a hierarchical structure for data distribution. Otherwise, there might be routing loops and packet duplications at designated members. In our CIM protocol, there are no island leaders. Ingress and egress nodes are all selected by the server. In the DIM protocol, only a single leader is elected in each island. The leader then adaptively identifies ingress and egress nodes. Our approaches hence have simpler management mechanism.

Most recently, we have proposed another protocol called scalable island multicast (SIM) [13], [14]. As a distributed protocol, SIM shares many features with DIM. They both allow distributed host joining and island management. They both use two multicast groups (i.e., DATA and CONTROL groups) for a session. The fundamental difference between them is that DIM uses island leaders but SIM does not. As a result, they have different ways to connect islands. In SIM, hosts first form an overlay tree. Based on the tree, each island identifies its egress(es) and further elects an ingress. But in DIM, each island first elects a leader, which identifies ingress and egress nodes for the island. Hence, a leader is responsible for all island management issues, including ingress/egress selection and maintenance, and island member joining and leaving. DIM's approach has its advantages and limitations. 1) A leader in DIM has more responsibility and heavier load than ordinary island members. It is critical to select and maintain a good leader. As a comparison, SIM does not need island leaders and does not need to specifically select egress nodes. In each island, only a single ingress is selected. SIM is hence simpler with lower control overhead and achieves more balanced load among hosts. 2) DIM achieves higher flexibility, adaptiveness and resilience than SIM. As a leader has full power to manage its island, it can freely select ingress and egress nodes. This makes DIM applicable for different applications with different delivery requirements. And by tuning and refining bridges, it can achieve adaptive improvement on delivery efficiency. If network or host dynamics affects a bridge, a leader can quickly select another new bridge. All these advantages of DIM come from its leader-based island management mechanism. 3) DIM has better system extensibility. In the current SIM and DIM protocols, each island has only one ingress. If the network bandwidth is very limited, we may consider using multiple ingresses for one island. To achieve that, DIM can require leaders to select more bridges to connect two islands. But for SIM, if an island has multiple ingresses, much more efforts have to be taken to avoid routing loops and packet duplications. In summary, DIM is more complicated with more control and extension on the system. A network application may select a proper protocol according to its system requirement.

Preliminary works on CIM and DIM have been partially discussed in [15]–[17]. In [15] and [16], we study CIM and DIM, respectively. In [17], we study the bridge-node selection and loss recovery issues in DIM. In this paper, we combine our previous work and present a complete study on island multicast. We further discuss practical implementation issues when applying our protocols to media streaming, and present comprehensive evaluation results on the protocols.

III. Centralized Island Multicast (CIM)

A. Overview

Applications such as multiparty conferencing often involve a small number of users and consume much network bandwidth. As the session size is not large, we can use a central server to collect all host information. With global information at hand, the server can build a bandwidth-efficient tree. As each user may be the source of data flows, we consider building a shared tree for all users. Therefore, CIM relies on a central server to compute a delivery tree spanning all users. When the tree is computed, the tree structure is distributed to all users. Each tree has a unique version number, which avoids clashing and routing loops with previous trees. Each user can then distribute data along the tree.

B. Host Joining and Leaving

Each session has a unique class-D IP address for IP multicast. A joining host first detects the existence of the island by sending an *Island_Detection* message to the class-D address. An

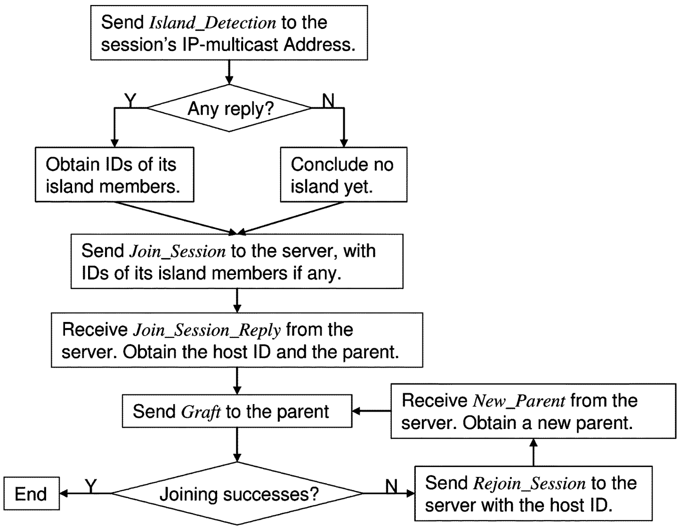

Send Island_Detection to the session's IP-multicast Address.
Any reply?
Y
N
Obtain IDs of its island members.
Conclude no island yet.
Send Join_Session to the server, with IDs of its island members if any.
Receive Join_Session_Reply from the server. Obtain the host ID and the parent.
Send Graft to the parent
Receive New_Parent from the server. Obtain a new parent.
End
Y
Joining successes?
N
Send Rejoin_Session to the server with the host ID.

Fig. 2. Host joining in CIM.

island member receiving the message, if any, replies with an *Island_Detection_Reply* message consisting of its host ID to the same multicast address (using IP-multicast). The joining host then knows members of its island. If no reply is received after a few trials, the joining host concludes that there are no members in its island.

Afterwards, the joining host sends a *Join_Session* message consisting of the IDs of its island members (if any) to the server. The server then replies with a *Join_Session_Reply* message which consists of a unique ID for the joining host and the designated parent. The parent selection mechanism will be discussed in Section III-C.

The joining host then sends a *Graft* message to the designated parent. The parent may reject the request if it is overloaded. If this occurs, the joining host sends a *Rejoin_Session* message consisting of its host ID to the server. Upon receiving the message, the server designates a new parent for it by a *New_Parent* message. We summarize the whole joining process in Fig. 2.

A leaving host sends a *Leave_Session* message to the server, which accordingly fixes tree partition by assigning a new parent to each of the leaving host's tree neighbors. That is, the server sends a *New_Parent* message to each of the host's tree neighbors, which then sends a *Graft* message to the new parent. After reconstructing the tree, the server sends a *Leave_Session_Reply* message to the leaving host. The leaving host then stops forwarding data packets and leaves the session.

C. Tree Computation and Maintenance

1) Tree Mechanisms: Suppose that each host measures round-trip time (RTT) to some other hosts and reports results to the server (details discussed in Section III-D). We consider building a minimum-diameter tree spanning all session members. As hosts have different edge bandwidth and computational power, we impose certain degree bounds on hosts according to their capabilities. This leads to the *minimum-diameter degree-bounded spanning tree* problem, which has been studied by Shi *et al.* [18]. We adopt and modify Shi's approach. In our approach, an inter-island path is assigned a weight equal

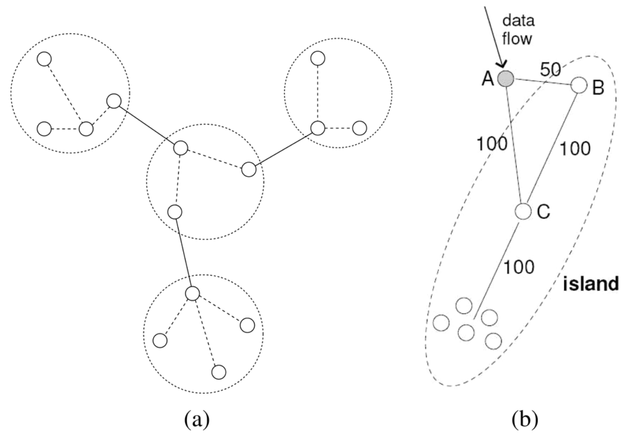


Fig. 3. (a) Example of a CIM tree. Solid lines indicate data forwarding paths, and dashed lines indicate control paths. (b) Example of bad bridge-node selection in CIM.

to its RTT (infinity if unknown), and an intra-island path is assigned weight -1. In this way, hosts within the same island are connected together as a cluster.

A tree edge between two hosts within the same island does not indicate a data delivery path. Data packets are forwarded by IP multicast within islands. These intra-island tree edges hence imply logical neighbor relationship between hosts, and are used for tree maintenance and loss recovery. On the other hand, tree edges across islands do indicate packet forwarding paths.

We show an example of a CIM tree with four islands in Fig. 3(a). The intra-island tree edges, as indicated by the dashed lines, are not used for data delivery. Instead, data are multicast within islands. Between different islands, packets are forwarded along tree edges via unicast, as indicated by the solid lines.

The server periodically computes a new tree (e.g., every 30 s in our PlanetLab experiments). If the total RTT along the new tree is smaller than that of the current tree by a certain threshold, the new tree is adopted. That is, the server informs each host of its new parent via a *New_Parent* message. Clearly, tuning the threshold can trade off tree cost with tree stability. Our preliminary experiments show that a threshold of 95% of the current total RTT can provide reasonable tree stability without sacrificing much tree cost.

Notice that a host may send *Rejoin_Session* message to the server and require a new parent (e.g., when the host's current parent unexpectedly fails). In this case, since the rest of the tree remains unchanged, we do not need to re-compute the whole tree and only need to fix the tree partition. The server will select a new parent with enough residual degree and small RTT for the host. In selection, the server prefers hosts:

- in the same island: The server will select a host in the same island if possible. Selecting a new parent in another island indicates a separate unicast connection. And a host in another island is often farther;
- with high responsiveness to ping messages: A host not responsive to ping messages is often busy or overloaded;
- with high residual degree: A host with high residual degree has enough space to accommodate a new child. Selecting such a host as the new parent can achieve load balancing among hosts.

2) Discussion on Tree Limitations: In CIM tree construction, we do not consider intra-island latency. When islands are large, intra-island latency may be high, and the above mechanism may construct a bad tree. We show an example in Fig. 3(b). In the figure, host A is going to unicast data to a nearby island. It has two choices to select the bridge-node in the island, i.e., hosts B or C. The distances between A, B and C are shown in the figure. According to the CIM tree construction mechanism, A will select B as the bridge-node, as B is closer to A than C.

However, we will see that C is a better choice than B, in terms of average end-to-end delay. Suppose there are five other hosts in the island, whose distances to C are all 100 and distances to B are all 200. For simplicity, suppose end-to-end delay at host A is 0. If B serves as the bridge-node, it will receive data from A and multicast them to other island members. End-to-end delay at B and C is 50 and $(50 + 100) = 150$, respectively. End-to-end delay at the remaining five hosts is all $(50+200) = 250$. The average delay at island members is $(50+150+250 \times 5)/7 = 207.14$. Alternatively, if C serves as the bridge-node, end-to-end delay at B and C is $(100 + 100) = 200$ and 100, respectively. End-to-end delay at the remaining five hosts is all $(100 + 100) = 200$. Accordingly, the average delay at island members is $(200 + 100 + 200 \times 5)/7 = 185.71$. Clearly, C leads to lower average end-to-end delay at island members than B.

The reason that we discard intra-island latency in CIM is that we want to connect all hosts in the same island together as a cluster in the spanning tree. We can then identify inter-island connections from the tree. As shown, this sacrifices some tree efficiency.

D. Neighbor Monitoring

In order to obtain updated RTT between hosts, the server periodically generates a neighbor list for each host. Upon receiving the list, a host pings peers in the list to either obtain the RTT between them or identify some failed hosts. Hosts then report the measurement results to the server. Each host also periodically pings its parent and children. Upon detecting the failure of its parent, a host requests a new parent from the server by a *Rejoin_Session* message.

To reduce overhead, the server limits the length of the neighbor list (e.g., 5 in our implementation). The frequency of sending lists from the server is set inversely proportional to the session size in order to achieve high scalability. When generating the neighbor list, the server prefers those with unknown or old RTT values. The server also removes unresponsive hosts from its tree computation.

IV. Distributed Island Multicast (DIM)

A. Overview

While CIM can quickly build a delivery tree with low overhead, it relies on a central server for tree construction and maintenance. The server becomes the system bottleneck and forms a single point of failure. When the session size is large, the server is easily overloaded. On the other hand, applications such as news broadcasting, software distribution and media streaming often involve more than thousands of end users. We need to develop a scalable protocol for these applications.

DIM is a fully distributed protocol for such purpose. It organizes hosts into a two-level hierarchy. The upper level contains inter-island connections, where a unicast-based overlay tree connects all islands. The lower level contains intra-island connections, where packets are delivered via IP multicast within islands. This two-level architecture guarantees that the whole delivery flow is loop-free.

In order to set up inter-island connections, each island elects a unique leader. A pure overlay protocol runs on top of leaders. Given a pair of neighboring islands (i.e., their leaders are directly connected in the inter-island tree), one host is selected from each island and the two hosts form a pair of bridge-nodes. The connection between bridge-nodes, instead of the connection between leaders, is the actual data delivery path between islands. In this way, each island has only one ingress and may have no or some egresses.

We require two class-D multicast addresses for each DIM session. One is used for multicasting data packets and the other is used for multicasting control messages. We call the groups corresponding to these two IP addresses a DATA group and a CONTROL group, respectively. Each host joins both groups. Packet forwarding rule at a host depends on the host role. The source or an ingress multicasts data packets to its DATA group. An egress forwards packets along its inter-island connection to the downstream island. The other hosts receive packets without forwarding. Note that a host may have multiple roles. For example, the source may also be an egress. In that case, it also forwards packets as an egress.

A leader periodically multicasts *HeartBeat* messages to the CONTROL group. A leader also runs a bridge-node selection algorithm to select ingress and egress. A leader will play the roles of ingress and egress for its island if there is no ingress or egress in the island. For a host not in any multicast island, we consider that it forms an island only consisting of itself. It plays the roles of leader, ingress and egress for the island.

B. Host Joining and Leaving

When a host joins the session, it first joins the DATA and CONTROL groups. If there exists an island, the host will receive the island leader's *HeartBeat* messages from the CONTROL group. The joining process then ends, and the host can receive data from its leader.

If the host does not receive any *HeartBeat* message, it forms an island only consisting of itself and becomes the island leader. The host then needs to further join the inter-island tree formed by leaders. This tree joining process depends on the used overlay protocol, which can be any existing overlay protocol. Afterwards, the host receives packets and forwards them according to the forwarding rules.

Regarding host leaving, if the host is not a leader, it unicasts a *Leave_Session* message to the leader and informs its leaving. The leader then plays the roles of the leaving host, if any, for the time being. On the other hand, if a leader leaves, it multicasts a *Leader_Leave* message to the CONTROL group and triggers the leader election process.

C. Leader Election

If the current leader fails (detected through the absence of its *HeartBeat* messages) or leaves the system (detected through the *Leader_Leave* message), a new leader needs to be elected. The leader election process works as follows. When a host discovers that its leader is absent, it waits for a random time and sends a *Leader_Elect* message with its local timestamp to the CONTROL group. On the other hand, if a host receives *Leader_Elect* messages before sending its own *Leader_Elect* message, the host does not send any message. If a host receives multiple *Leader_Elect* messages, it selects the sender with the smallest timestamp as the leader. In this way, the host sending message with the smallest timestamp finally becomes the new leader. In case of contention, the host with the lexically lowest IP address is selected as the leader. The new leader then advertises itself to the whole group. Note that we do not need to synchronize time at different hosts.

D. Bridge-Node Selection

Bridge-node selection is a periodical and distributed process for tree improvement. We consider and compare two classes of selection methods as follows.

Individual Bridge-Node Selection: In individual selection, a bridge-node is selected independent of the other bridge-node in its neighboring island. An island leader periodically multicasts the list of current bridge-nodes to its island members through *HeartBeat* messages. Upon receiving the message, if a host finds that itself is a better bridge-node (based on the metrics discussed below) for some neighboring island, it sends a *Candidate* message to the CONTROL group after a random delay. The *Candidate* message contains a list of numerical values, each representing the cost (e.g., delay) of connecting to one neighboring island.

Based on the received *HeartBeat* and *Candidate* messages, a host in the island can maintain a list of the best bridge-nodes to the neighboring islands. A host suppresses its *Candidate* message if it cannot improve any of the costs. Whenever a better bridge-node is found, the leader informs the corresponding neighboring island's leader about the new bridge-node.

We propose the following two metrics for individual selection.

— **Closest to Neighbor's Centroid (CNC)**: Suppose that hosts can obtain their network coordinates using tools like GNP [19] or Vivaldi [20]. In CNC method, an island selects, for each neighboring island, a bridge-node that is the closest to the centroid of the neighboring island.
In detail, each host reports its network coordinates to the leader when joining the island. The leader can then compute the island centroid and periodically advertise it to the neighboring leaders. The hosts closest to the centroids of the neighboring islands are selected as bridge-nodes. For example, in Fig. 4(a), host N_1 is the closest to the centroid of the neighboring island I_1. It is selected as the bridge-node to I_1.

— **Closest to Neighbor's Leader (CNL)**: CNC requires a leader to periodically compute the island centroid. With frequent joining or leaving of island members, the computational overhead may be high. We hence propose CNL,

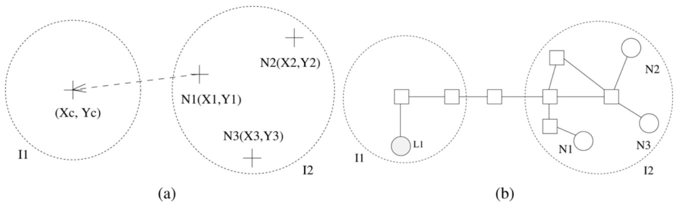


Fig. 4. Two metrics for individual bridge-node selection in DIM. (a) CNC: N_1, N_2 and N_3 are three hosts in island I_2. (X_c, Y_c) is the centroid of island I_1. In I_2, N_1 is selected as the bridge-node to I_1 because it is the closest to the centroid of I_1. (b) CNL: N_1, N_2 and N_3 are three hosts in island I_2. L_1 is the leader of island I_1. Squares in the figure are routers. In island I_2, N_1 is selected as the bridge-node to island I_1 because N_1 is the closest to L_1.

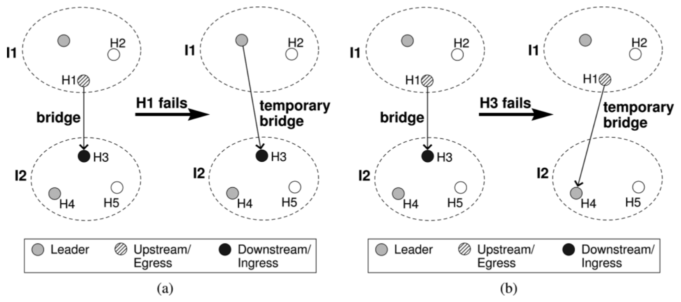


Fig. 5. Recovery for bridge-node failure in DIM. (a) Recovery for egress failure. (b) Recovery for ingress failure.

where hosts closest to the leaders of neighboring islands are selected as bridge-nodes. Here distance can be computed based on RTT or network coordinates. We show an example in Fig. 4(b), where host N_1 is selected as the bridge-node to I_1 since it is the host closest to L_1, the leader of I_1, from island I_2.

Pair-Wise Bridge-Node Selection: In pair-wise bridge-node selection, two neighboring islands cooperatively select a pair of bridge-nodes. A pair of close hosts are often selected, as this can reduce delivery delay. It can also be extended to take residual bandwidth or loss rate into consideration. We compare the following two selection methods.

— **Closest among All Pairs (CAP)**: If ping measurements are used to obtain inter-host distance, each leader needs to exchange the list of all its island members with the neighboring leaders. It also periodically multicasts to its members the list of hosts in its neighboring islands. A host then measures RTT to the hosts in the neighboring islands. If it finds a path with smaller RTT than the current bridge path, the host informs its leader, which in turn informs the corresponding neighboring leader about the new bridge. Clearly, this method requires $O(N_1 \times N_2)$ ping measurements across islands, given a pair of neighboring islands with N_1 and N_2 members, respectively. This generates heavy traffic if islands are large.
Alternatively, if network coordinates are used, leaders only need to exchange host coordinates and multicast them within islands. A host can then compute the distance to any other host based on their coordinates. Systems like GNP take only $O(N)$ pings to estimate the coordinates of N hosts. The measurement overhead is then reduced.

— **Closest among Random Pairs (CRP)**: CAP either incurs high measurement overhead or requires additional coordinate estimation service. We hence propose CRP to further reduce measurement overhead. In CRP, each island randomly selects a constant number of island members to perform ping measurements (denoting the number as C). It works as follows.

1) The leader polls its island members by periodical *HeartBeat* messages.
2) Upon receiving a *HeartBeat* message, a non-bridge-node replies with a certain probability ρ and a bridge-node always replies, all by unicast. The absence of bridge-node replies for a certain period of time would trigger bridge-node selection. A leader can dynamically adjust ρ to ensure the total number of replies is roughly equal to C.
3) During bridge-node selection, a list of C members in an island is sent to the neighboring islands.
4) Upon receiving the list of C hosts, a leader randomly pairs each of the hosts in the list with a member from its own island. These host pairs then perform ping measurements with each other.
5) A leader selects the pair with the minimum RTT as the bridge-nodes.

Using CRP, there are $O(C)$ pings between a pair of neighboring islands. If we set C to a constant, this overhead is lower than $O(N)$ or $O(N^2)$ in CAP. In our simulations, we set C to 10. The results show that it can achieve similar performance as CAP.

E. Failure Recovery

There are two types of host failure that require recovery: leader failure and bridge-node failure. When a leader fails, the leader election process is triggered (as described in Section IV-C).

If a bridge-node fails, the leader temporarily takes the role of the bridge-node, as shown in Fig. 5. The leader then selects a new bridge-node as described above. In detail, an ingress is monitored by its leader, and an egress is monitored by its downstream bridge-node. An ingress or egress is required to periodically report its status to its monitoring host. Its failure or leaving can be detected through the missing of its reports. A special case is that an ingress happens to be a leader. In this case, the failed ingress is directly replaced by a new leader.

Clearly, the robustness of DIM relies on the robustness of leaders. As long as an island has a qualified leader, all island management issues can be well addressed. So the major issue is to keep and maintain robust leaders. Firstly, leaders had better be stable with a long lifetime. Secondly, the overlay protocol spanning leaders should be highly robust. For example, the Delaunay triangulation (DT) overlay protocol is a robust protocol [21]. It maintains a mesh among hosts, which provides backup connections between hosts in case of tree partition.

V. Application Case: Media Streaming

Our island multicast protocols may be used in various applications such as media streaming, video conferencing and file distribution. In this section, we select media streaming as an example. Media streaming has become a popular application over the Internet. For example, a peer-to-peer streaming software PPLive reported more than 400 000 concurrent peers for its over 300 program channels [22]. When applying CIM or DIM for media streaming, there are some practical implementation issues. We explore them as follows.

A. *Traversing Network Address Translators (NATs)*

In the real Internet, some hosts are behind NATs and have only limited connectivity. These hosts are called restricted hosts. A host that is not behind any NAT is said to be public. A restricted host can communicate with only public hosts or other restricted hosts behind the same NAT, while a public host can communicate with either a restricted host (in this case, the restricted host has to set up the connection) or a public one.

The major challenge for NAT traversal in island multicast is on unicast connection. For the IP multicast part, if a host can successfully join an island through message exchange, it can participate in subsequent island operations. Otherwise, it is regarded as a host not within any island (or equivalently, an island only consisting of itself), which will connect to other hosts via unicast connections.

First of all, a new host needs to identify whether itself is public or restricted before joining the system. This can be achieved by the STUN protocol [23]. Then we discuss the unicast connection setup mechanisms in CIM and DIM separately.

1) CIM: CIM tree construction is based on inter-host RTT. If the RTT value between two hosts is obtainable, the two hosts clearly can set up a connection between each other. An unmeasured path with unknown RTT is assumed to have infinity RTT and will not be used in the tree. Therefore, we only need to consider how to measure RTT between hosts, in other words, how to generate a neighbor list for a host.

If the host is restricted, the server generates a neighbor list where most hosts in the list are public. We still leave some space for restricted hosts in the list. This is for detecting whether the selected restricted hosts and the current host are behind the same NAT. If they are, they can communicate with each other.

If the host is public, the server randomly generates a neighbor list. However, the case is a bit complicated if there is a restricted host in the neighbor list. Note that a public host cannot initiate a connection with a restricted host. After the server sends the neighbor list, the server notifies the selected restricted hosts to set up a connection with the public host. Here the server keeps connections to every host in the system.

2) DIM: In DIM, leaders had better be public hosts, because a leader has to frequently communicate with other leaders and its island members. We can add this constraint to the leader election criteria. However, this is not a hard requirement. Some islands may contain no public hosts. So we still need to consider the case of restricted leaders.

We now analyze messages related to leaders. Most messages sent or received by leaders are distributed via IP multicast, e.g., leader's periodical *HeartBeat* messages, leader election messages, and most bridge-node election messages. Unicast messages sent or received by leaders include: messages exchanged with other leaders, *Leave_Session* messages sent by a leaving host, and messages in CRP bridge-node selection. We analyze the impact of restricted leaders to these messages as follows.

- *Messages exchanged with other leaders*: There have been many solutions for overlay construction in the presence of NATs [24], [25]. By using these protocols, leaders can form a connected overlay, even in the presence of restricted leaders.
- *Leave_Session messages*: This type of message is sent by a non-leader island member to its leader when leaving the session. Given a restricted leader, this message can be multicast in the CONTROL group instead of unicasting to the leader.
- *CRP bridge-node selection messages*: In CRP, island members reply to *HeartBeat* messages by unicast. When receiving a list of hosts from a neighboring island, the leader needs to pair (via unicast) each host with one of its island members. Clearly, all these messages can be multicast in the CONTROL group instead of unicast, at the cost of consuming more bandwidth.

Besides the restricted leader issue, there is another issue due to the presence of NATs. In pair-wise bridge-node selection, a pair of hosts selected by a leader may not be able to communicate with each other. If there is at least one restricted host in the selected pair, the leader needs to notify the restricted host to start the communication. If both hosts in the pair are restricted, the leader may randomly notify one. Notification can be achieved by multicasting in the CONTROL group. That is, if the selected restricted host is in the leader's own island, the leader can multicast a message in its CONTROL group to notify the restricted host. If the selected restricted host is in another island, the leader can first notify the restricted host's leader, which in turn notifies the restricted host by multicast.

B. *Fault Tolerance and Loss Recovery*

During media streaming, a host may suffer packet loss due to various reasons. A path may be congested or fail. The parent of a host may unexpectedly leave the system. Furthermore, there is application level loss. That is, a streaming application usually has a playback deadline by which data delivery and loss recovery have to be accomplished. Data packets received after the deadline are useless and regarded as loss.

1) Packet Loss Recovery: SIM has adopted a modified lateral error recovery scheme for loss recovery [14]. We can use similar idea in our CIM and DIM protocols. Basically, each host identifies a few other hosts as its recovery neighbors. A host also estimates recovery latency from each of its recovery neighbors. Whenever a loss occurs, the host sends a retransmission request to the recovery neighbor with the smallest recovery latency. If the retransmission fails again, the host turns to the recovery neighbor with the second smallest recovery latency, and so on.

The selection of recovery neighbors is not trivial. We first consider the case of CIM. For simplicity, we assume that there is a single source, which provides complete video content to the system. Hosts hence form a tree rooted at the source. The server selects recovery neighbors for a host according to the following requirements: 1) It is not in the host's subtree. 2) It is not the ancestor of the host. 3) It is not in the same island as the host.

These constraints can reduce loss correlation between a host and its recovery neighbor.

In DIM, we can select recovery neighbors based on the inter-leader tree. Similarly, we regard the streaming source as the root of the inter-leader tree. If the source is in an island and not in the tree, we regard its corresponding leader as the tree root. Then, we can select recovery neighbors for a host as follows: 1) In the inter-leader tree, a recovery neighbor's leader is not in the subtree of the host's leader. 2) In the inter-leader tree, a recovery neighbor's leader is not the ancestor of the host's leader. 3) A recovery neighbor is not in the same island as the host. In order to verify the constraints, a leader needs to maintain information about its path to the root in the inter-leader tree.

2) Selectively Discarding Packets: A host in the system receives data from its parent as well as recovery neighbors (if needed). However, if the edge bandwidth of the host is limited, its receiving rate will be accordingly limited and multiple-path delivery does not help much. In this case, the host should accept only the most important packets and skip less important ones.

A video stream encoded by currently popular video standards (e.g., H.264) usually consists of I-frames and P-frames. An I-frame is a standalone frame which can be played back by itself, and a P-frame is predicted from its immediately previous frame. As a result, if the immediately previous frame of a P-frame is lost, the P-frame will be useless. The group of frames leading by an I-frame and ending by P-frames is called a group of pictures (GoP). Clearly, the importance of frames in a GoP sequence decreases from the first I-frame to the last frame in the GoP. Therefore, a host with a limited receiving rate can skip a certain number of trailing P-frames in each GoP until the remaining frames can be transmitted at the current rate. It then requires only the remaining frames from its parent or recovery neighbors.

A more sophisticated design may use advanced coding techniques such as layered coding or multiple description coding. For example, in layered coding, there is a base layer which contains the data representing the most important features of the video [26]. Additional layers, called enhancement layers, contain data that progressively refine the reconstructed video quality. If a host does not have enough edge bandwidth, it can selectively discard enhancement layers.

VI. Illustrative Simulation Results

A. Simulation Environment and Metrics

We generate ten Transit-Stub topologies with Georgia Tech's topology generator [27]. Each topology is a two-layer hierarchy of transit domains and stub domains. The transit domains form a backbone and all stub domains are connected to the backbone. Each topology has four transit domains and 32 stub domains. On average, a transit domain contains eight routers, and a stub domain contains 32 routers. The delay of physical links is uniformly distributed between [0.1, 3) units. In our simulations, a host is randomly attached to a stub router. A certain percentage of stub domains are set as multicast-capable, where all routers in the domain are multicast-capable. Unless otherwise indicated, we set 75% stub domains as multicast-capable and set the session size to 200.

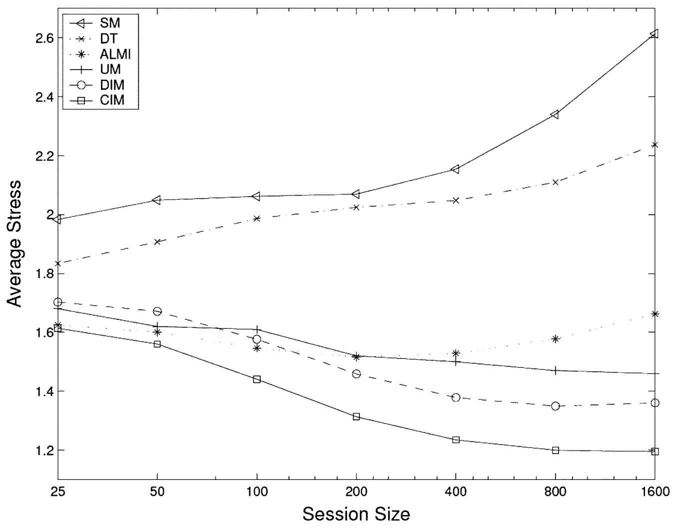


Fig. 6. Average link stress versus session size.

In DIM, we use the DT overlay protocol to build the inter-leader tree [21]. We use GNP to estimate host coordinates, where we select ten landmarks based on the N-cluster-median criterion as in [19]. We use the CNL bridge-node selection method (using RTT) by default.

We simulate several other protocols for comparison. We select two pure overlay protocols, namely, DT [21] and ALMI [28]. We also select two protocols using IP multicast, namely, SM [10] and UM (with multiple designated members and tree-based inter-island routing) [12].

We use the following metrics to evaluate the protocols:

- *Physical link stress*: defined as the number of copies of a packet transmitted over a certain physical link [3];
- *Relative delay penalty*: defined as the ratio of the overlay delay from the source to a given host to the delay along the shortest unicast path between them [3];
- *Out-degree*: defined as the number of identical packets a host sends. It indicates the forwarding load on a host. Note that sending a packet to a multicast address counts as one out-degree.

B. Performance Comparison Among Different Protocols

Fig. 6 shows the average link stress versus the session size. The stresses of CIM and DIM are significantly lower than those of other protocols. In most other protocols, the stress increases with the session size. But in CIM, DIM, and another IP multicast based protocol UM, the stress decreases with the session size. This is because when the session size increases, there are more hosts in multicast islands. Hence, the number of unicast paths (connecting islands and hosts not in any islands) does not increase as fast as the number of IP multicast paths. The average stress then decreases. Although SM uses IP multicast, it has the highest stress. This is because it forms a star-like overlay network. Note that UM achieves slightly worse performance than DIM. As discussed, its designated member selection mechanism does not consider neighboring islands. The resulting designated members may not be efficient for data distribution.

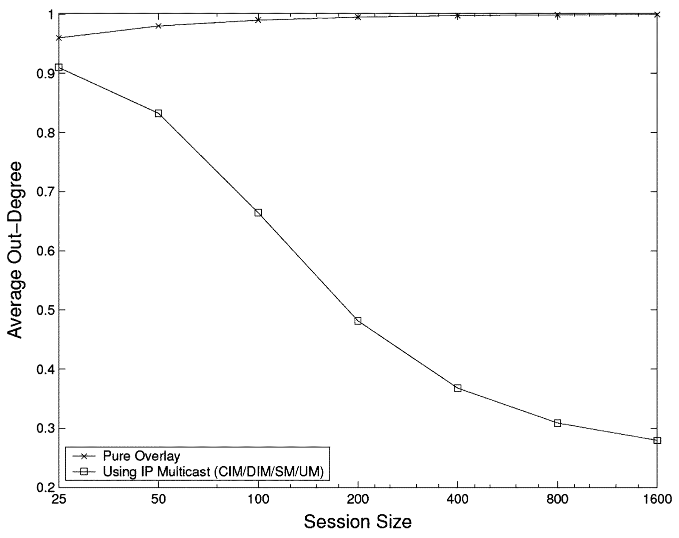


Fig. 7. Average out-degree versus session size.

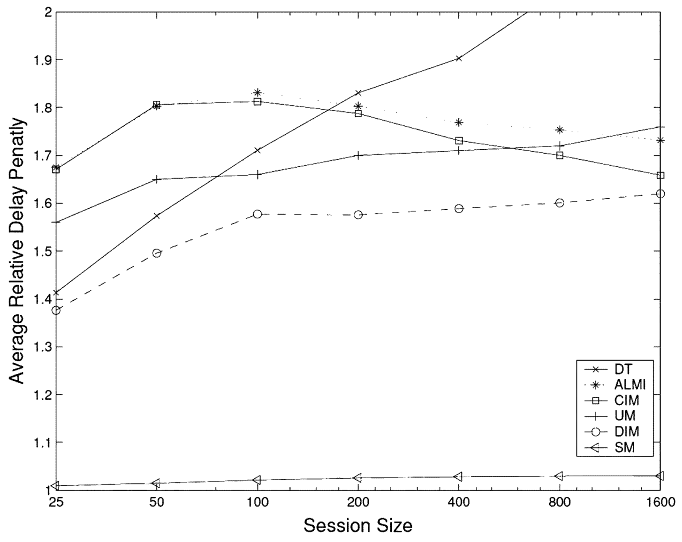


Fig. 8. Average relative delay penalty versus session size.

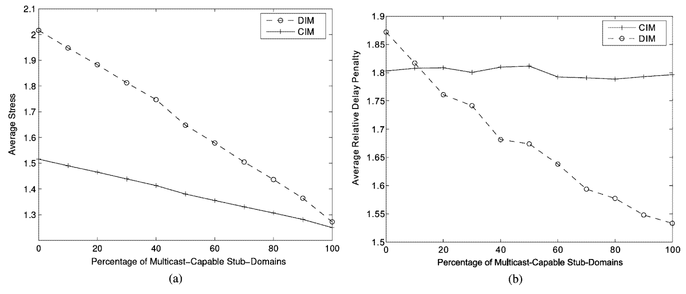


Fig. 9. Performance of CIM and DIM with different percentages of multicast-capable domains. (a) Average link stress. (b) Average relative delay penalty.

Fig. 7 shows the average out-degree achieved by the protocols. We classify the simulated protocols into two categories. One contains pure overlay protocols such as DT and ALMI. The other contains protocols using IP multicast, including CIM, DIM, SM, and UM. In the first category, hosts form a unicast based overlay tree for data distribution. Given n hosts in the tree, the average out-degree is $(n-1)/n$. In the second category, all the four protocols achieve exactly the same average out-degree given the same physical network and the same host locations. We explain it as follows. The total out-degree of hosts for these protocols consists of two parts: 1) Unicast connections: Suppose there are k multicast islands in the network and t hosts not within any islands. They require in total $(k+t-1)$ unicast connections, accounting for out-degree $(k+t-1)$. 2) IP multicast within islands: Each island has one ingress, which has an out-degree 1. So k islands account for in total out-degree k. Therefore, the total out-degree is $(2k+t-1)$. When the physical network and host locations are the same, CIM, DIM, SM, and UM have the same total out-degree, and accordingly the same average out-degree.

From Fig. 7, the average out-degree of protocols using IP multicast decreases as the session size increases. This is because an ingress forwards only one copy of packets within its island, regardless of the number of island members. Therefore, when the number of hosts in islands increases, the average out-degree decreases.

Fig. 8 shows the average relative delay penalty versus the session size. Since SM is based on a star-like overlay, it has the lowest end-to-end delay. Pure overlay protocols like DT and ALMI do not perform well for large sessions. As more hosts join the session, the overlay tree gets deeper and hosts have to experience higher delay. Among the other three protocols using IP multicast, DIM achieves the lowest relative delay penalty. Its bridge-node selection mechanism is highly adaptive and efficient, even when network condition significantly changes. CIM achieves similar relative delay penalty as ALMI. They both build a minimum spanning tree for data distribution. UM performs better than CIM but worse than DIM. As discussed, CIM may not perform well when islands are large, but DIM and UM are both adaptive to large islands. From the figure, DIM is better than UM. It bridge-node selection mechanism directly reduces inter-island delay.

Fig. 9 compares the performance of CIM and DIM in terms of stress and relative delay penalty. Here we do not show the result of out-degree. As discussed above, CIM and DIM achieve the same average out-degree when network and host condition are the same.

In the figure, zero percentage corresponds to the pure overlay case. Fig. 9(a) shows link stress versus percentage of multicast-capable domains. For both protocols, stress decreases as the percentage of multicast-capable domains increases. As known, IP multicast always achieves an average stress of 1. The more IP multicast paths in the delivery tree, the lower average stress. Note that the average stress at 100% multicast-capable domain is not 1 because the transit domains are still multicast-incapable. CIM has much lower stress than DIM, at the cost of higher relative delay penalty. As shown in Fig. 9(b), the relative delay penalty of CIM does not sensitively depend on the percentage while DIM does. A possible reason is that DIM has carefully selected short bridge paths to connect islands, which can significantly reduce relative delay penalty when island number is high.

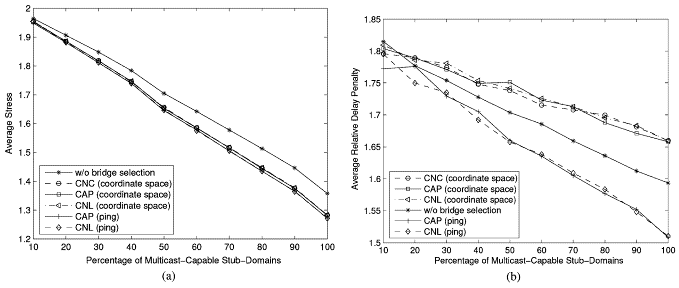


Fig. 10. Performance of different bridge-node selection methods. (a) Average link stress. (b) Average relative delay penalty.

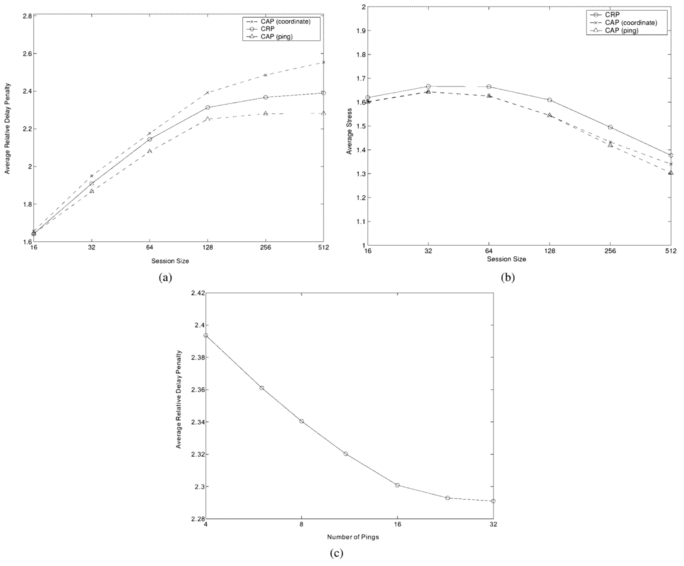


Fig. 11. Performance comparison between CAP and CRP. (a) Average relative delay penalty. (b) Average link stress. (c) Tunning C for CRP.

The scalability of CIM and DIM is not quantitatively evaluated in our simulations. This is because the scalability of CIM depends on the capability of the server (e.g., computational power and bandwidth). On the other hand, because CIM is centralized and DIM is fully distributed, DIM achieves much better scalability than CIM.

C. Performance Tunning on DIM

We compare different bridge-node selection methods CNC, CNL, and CAP in Fig. 10. As shown in Fig. 10(a), proper bridge-node selection can reduce link stress. For stress, there is little difference between using network coordinates and using ping measurement. But for relative delay penalty, these two measurement mechanisms lead to significantly different results [as shown in Fig. 10(b)]. This is because estimation error of path length based on network coordinates is relatively high for short paths. While we prefer short paths for data delivery in order to achieve low delay, the estimation error will seriously affect the tree performance. In our simulations, the protocols using network coordinates have even higher relative delay penalty than those with no bridge-node selection method. As a comparison, using ping measurement can achieve much lower relative delay penalty. From the figure, CAP by ping and CNL by ping achieve similar relative delay penalty. But CAP incurs much heavier measurement traffic. It is hence not as efficient as CNL.

We compare the performance of CRP (using ping) and CAP (using either GNP or ping) in Fig. 11. We set $C = 10$ for CRP. Fig. 11(a) shows that CRP achieves lower relative delay penalty than GNP-based CAP but higher relative delay penalty than ping-based CAP. Again, the results show that GNP is not as accurate as direct ping measurement. From Fig. 11(a) and (b), CRP achieves comparable performance as CAP. As CRP has much lower measurement overhead, it is more cost efficient.

Fig. 11(c) shows the performance of CRP with different C values. As expected, relative delay penalty decreases when C increases. By pinging more paths, the system can discover closer pairs of bridge-nodes. Furthermore, relative delay penalty becomes stable when C is larger than 16. It shows that we do not need to select a large C value beyond that in order to achieve low relative delay penalty.

VII. Experimental Results on PlanetLab

We have implemented a CIM prototype by C++. The library codes are publicly available at http://www.mwnet.cse.ust.hk/im. We have also deployed it on the PlanetLab testbed [7]. In this section, we present our measurement results on PlanetLab.

We do not test DIM on PlanetLab. We observe that most multicast-capable domains on PlanetLab contain two or three hosts. This is due to the construction nature of the PlanetLab, where each university or organization contributes two or three machines in its local domain as PlanetLab nodes. Within these small multicast domains, the bridge-node selection methods of DIM do not have much tuning space. DIM will then work like CIM, and the major difference between them is on the overlay schemes adopted in the two protocols. Therefore, we present only the results of CIM here.

A. Experimental Environment

Our experiments consist of three phases: joining phase, stabilization phase, and leaving phase. In the joining phase, a set of hosts randomly join the session. Then, hosts form a stable overlay tree during the stabilization phase. After that, we measure delivery efficiency of the tree. Finally, in the leaving phase, all hosts leave the session after a random time. In our experiments, PlanetLab domains are randomly selected from America, Europe, and Asia. All hosts in the selected domains are used. We have found that most domains are multicast-capable and contain two or three hosts. We use the following metrics to evaluate the protocol.

- *Joining and leaving latencies*: These metrics measure how long the joining and leaving operations take. Joining latency is the time between sending the first *Island_Detection* message and receiving the *Join_Session_Reply* message from the server. Leaving latency is the time between sending a *Leave_Session* message and receiving the *Leave_Session_Reply* message from the server.
- *Relative delay penalty*: as defined in Section VI-A. We use application-layer ping (i.e., sending a packet to the destination and waiting for the reply) instead of ICMP ping to compute the unicast delay between two hosts. This is

TABLE I
SIZES OF CONTROL MESSAGES

Message Type	*Size (bytes)*
Create Session	44
Create Session Reply	10
Close Session	10
Close Session Reply	8
Join Session	36
Join Session Reply	88
Rejoin Session	40
New Parent	40
Leave Session	8
Leave Session Reply	8
Graft	40
Graft Reply	10
Island Detection	10
Island Detection Reply	10
Ping	10
Ping Reply	8
Peer List	336

because some nodes may be behind firewalls which block ICMP packets.

- *Out-degree*: as defined in Section VI-A.
- *Control overhead*: defined as the bandwidth for transmitting control messages at a host. The sizes of control messages are listed in Table I.

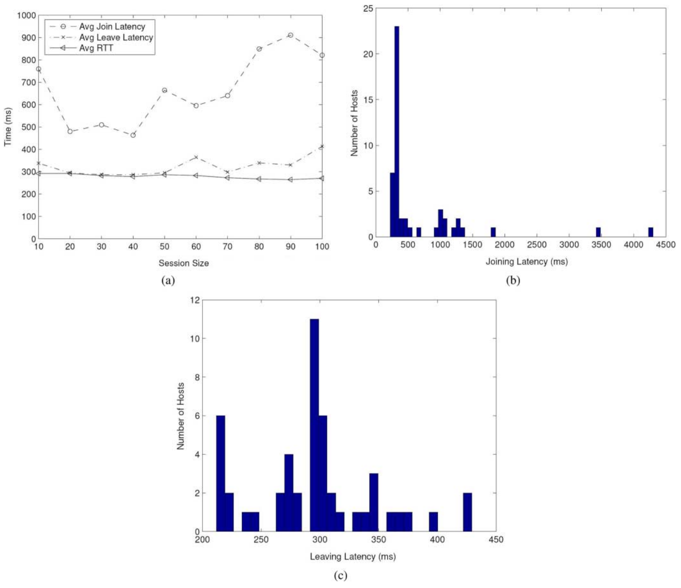


Fig. 12. Joining and leaving latencies (on PlanetLab). (a) Average RTT, joining, and leaving latencies versus session size. (b) Joining latency distribution (session size 50). (c) Leaving latency distribution (session size 50).

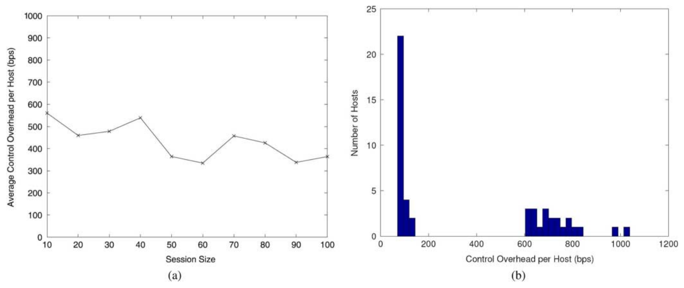


Fig. 13. Control overhead (on PlanetLab). (a) Average control overhead versus session size. (b) Control overhead distribution (session size 50).

B. Measurement Results

Fig. 12(a) shows the average joining and leaving latencies. We also show the average RTT between the server and hosts. The joining latency is higher than the leaving latency because hosts have to detect island before contacting the server. The latencies do not regularly increase as the session size increases. This is because joining and leaving randomly happen during the experiments. If many hosts join or leave at the same time, the server response time will increase and so will the latencies.

Distribution of latencies is shown in Fig. 12(b) and (c). The session size is set to 50. Regarding joining, most hosts experience latencies less than the average (around 660 ms), while a few suffer from high latencies. This may be due to island detection (timeout) and random overload condition at the server. As a comparison, the leaving latencies of hosts are more evenly distributed. A host only needs to contact the server when leaving.

Fig. 13 shows the control overhead of CIM. From Fig. 13(a), the average bandwidth consumption for control messages at a host is less than 600 bps. Such overhead is negligible. CIM's overhead remains low when the session size increases. In CIM, the frequency of RTT measurements decreases with the session size.

Distribution of overhead is shown in Fig. 13(b). The lower overhead in the figure (around 100 bps) corresponds to leaf hosts in the delivery tree, and the reminder corresponds to internal hosts. In CIM, each host other than the source periodically pings its parent to ensure that the parent is alive and that the tree is connected. As a result, the more children a host has, the more ping messages it receives.

In Fig. 14, we compare the average out-degree of CIM with pure overlay protocols. As mentioned, the average outdegree for an n-host overlay tree is $(n-1)/n$. CIM has lower average out-degree than overlay protocols. The out-degree distribution contains little useful information and is not shown here. In a snapshot sample, for a session of 50 hosts, we observe that 34 hosts have out-degree 0 (i.e., leaf hosts), ten hosts have out-degree 1 (e.g., bridge-nodes), and six hosts have out-degree 2. The average out-degree is low because most hosts are leaves.

Fig. 15 shows the relative delay penalty of CIM, which is relatively low as it uses underlying IP multicast. Regarding relative delay penalty distribution, though most relative delay penalty values are around the average, some are very high (corresponding to leaf hosts). Furthermore, a few hosts have relative delay penalty less than 1. This is because the Internet has the route inefficiency problem where an indirect path

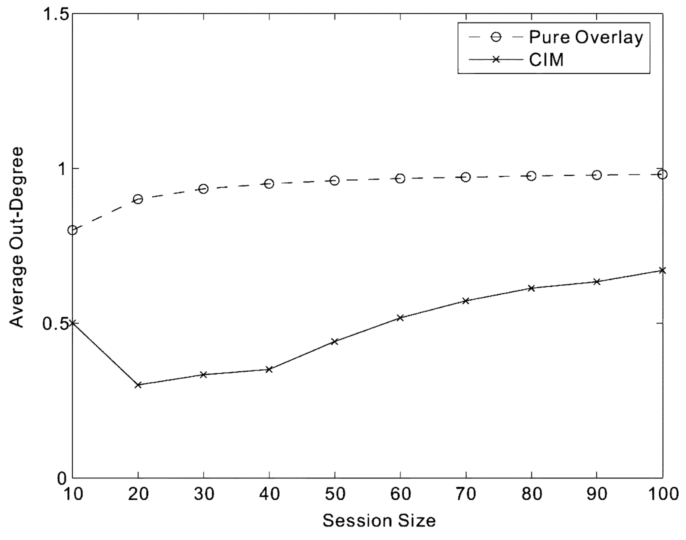


Fig. 14. Average out-degree versus session size (on PlanetLab).

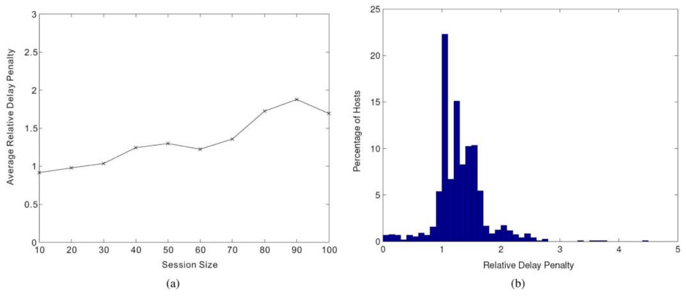


Fig. 15. Relative delay penalty (on PlanetLab). (a) Average relative delay penalty versus session size. (b) Relative delay penalty distribution (session size 50).

between two hosts via third-party hosts may be shorter than the direct path between them [29].

VIII. Conclusion

Today's Internet contains many small multicast-capable islands. In order to achieve global multicast, these local multicast islands have to be connected via unicast connections. This is so-called island multicast. In this paper, we present two protocols for island multicast. The first protocol, termed CIM, relies on a central server to compute and maintain a delivery tree. The second one, termed DIM, allows distributed host joining and tree construction. We study the key components in these two protocols.

We have conducted extensive simulations to evaluate our protocols. We have also implemented and tested CIM on the PlanetLab testbed. Our results show that both protocols are efficient in terms of link stress, relative delay penalty, and control overhead. Our study shows that it is useful to consider local multicast capability when designing overlay protocols.

References

[1] S. E. Deering, "Multicast routing in internetworks and extended LANs," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 18, no. 4, pp. 55–64, Aug. 1988.

[2] C. Diot, B. N. Levine, B. Lyles, H. Kassem, and D. Balensiefen, "Deployment issues for the IP multicast service and architecture," *IEEE Network*, vol. 14, no. 1, pp. 78–88, Jan. 2000.

[3] Y. H. Chu, S. Rao, S. Seshan, and H. Zhang, "A case for end system multicast," *IEEE J. Select. Areas Commun.*, vol. 20, no. 8, pp. 1456–1471, Oct. 2002.

[4] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast," in *Proc. ACM SIGCOMM'02*, Aug. 2002, pp. 205–217.

[5] N. Magharei, R. Rejaie, and Y. Guo, "Mesh or multiple-tree: A comparative study of live P2P streaming approaches," in *Proc. IEEE INFOCOM'07*, May 2007.

[6] S. Ratnasamy, A. Ermolinskiy, and S. Shenker, "Revisiting IP multicast," in *Proc. SIGCOMM'06*, Sep. 2006, pp. 15–26.

[7] PlanetLab. [Online]. Available: http://www.planet-lab.org.

[8] H. Eriksson, "MBONE: The multicast backbone," *Commun. ACM*, vol. 37, no. 8, pp. 54–60, Aug. 1994.

[9] D. Thaler, M. Talwar, L. Vicisano, and D. Ooms, "IPv4 automatic multicast without explicit tunnels (AMT)," *Internet Draft, IETF*, Feb. 2004.

[10] J. Park, S. J. Koh, S. G. Kang, and D. Y. Kim, "Multicast delivery based on unicast and subnet multicast," *IEEE Commun. Lett.*, vol. 5, no. 4, pp. 1489–1499, Apr. 2001.

[11] B. Zhang, S. Jamin, and L. Zhang, "Host multicast: A framework for delivering multicast to end users," in *Proc. IEEE INFOCOM'02*, Jun. 2002, pp. 1366–1375.

[12] B. Zhang, W. Wang, S. Jamin, D. Massey, and L. Zhang, "Universal IP multicast delivery," *Comput. Netw.*, vol. 50, no. 6, pp. 781–806, 2006.

[13] X. Jin, K.-L. Cheng, and S.-H. G. Chan, "SIM: Scalable island multicast for peer-to-peer media streaming," in *Proc. IEEE Int. Conf. Multimedia & Expo (ICME'06)*, Jul. 2006, pp. 913–916.

[14] X. Jin, K.-L. Cheng, and S.-H. G. Chan, "Scalable island multicast for peer-to-peer streaming," *Hindawi J. Adv. Multimedia*, vol. 2007, Article ID 78913, 2007.

[15] K.-L. Cheng, K.-W. Cheuk, and S.-H. Chan, "Implementation and performance measurement of an island multicast protocol," in *Proc. IEEE Int. Conf. Communications (ICC'05)*, May 2005, pp. 1299–1303.

[16] K.-W. Cheuk, S.-H. Chan, and J. Lee, "Island multicast: The combination of IP multicast with application-level multicast," in *Proc. IEEE Int. Conf. Communications (ICC'04)*, Jun. 2004, pp. 1441–1445.

[17] W.-P. Yiu, K.-F. Wong, and S.-H. Chan, "Bridge-node selection and loss recovery in island multicast," in *Proc. IEEE Int. Conf. Communications (ICC'05)*, May 2005, pp. 1304–1308.

[18] S. Y. Shi, J. S. Turner, and M. Waldvogel, "Dimensioning server access bandwidth and multicast routing in overlay networks," in *Proc. ACM Int. Workshop Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'01)*, 2001, pp. 83–91.

[19] T. S. E. Ng and H. Zhang, "Predicting internet network distance with coordinates-based approaches," in *Proc. IEEE INFOCOM'02*, Jun. 2002, pp. 170–179.

[20] F. Dabek, R. Cox, F. Kaashoek, and R. Morris, "Vivaldi: A decentralized network coordinate system," in *Proc. ACM SIGCOMM'04*, Aug. 2004, pp. 15–26.

[21] J. Liebeherr, M. Nahas, and W. Si, "Application-layer multicasting with Delaunay triangulation overlays," *IEEE J. Select. Areas Commun.*, vol. 20, no. 8, pp. 1472–1488, Oct. 2002.

[22] X. Hei, C. Liang, J. Liang, Y. Liu, and K. W. Ross, "A measurement study of a large-scale P2P IPTV system," *IEEE Trans. Multimedia*, vol. 9, no. 8, pp. 1672–1687, Dec. 2007.

[23] J. Rosenberg, J. Weinberger, C. Huitema, and R. Mahy, "STUN—Simple traversal of user datagram protocol (UDP) through network address translators (NATs)," RFC 3489, Mar. 2003.

[24] A. Ganjam and H. Zhang, "Connectivity restrictions in overlay multicast," in *Proc. ACM Int. Workshop Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'04)*, 2004.

[25] W. Wang, C. Jin, and S. Jamin, "Network overlay construction under limited end-to-end reachability," in *Proc. IEEE INFOCOM'05*, Mar. 2005, pp. 2124–2134.

[26] B. Li and J. Liu, "Multirate video multicast over the internet: An overview," *IEEE Network*, vol. 17, no. 1, pp. 24–29, Jan. 2003.

[27] E. Zegura, K. Calvert, and S. Bhattacharjee, "How to model an internetwork," in *Proc. IEEE INFOCOM'96*, Mar. 1996, pp. 594–602.

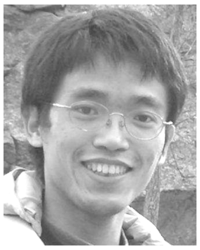
[28] D. Pendarakis, S. Shi, D. Verma, and M. Waldvogel, "ALMI: An application level multicast infrastructure," in *Proc. USENIX Symp. Internet Technology and Systems (USITS'01)*, Mar. 2001, pp. 49–60.

[29] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. E. Anderson, "The end-to-end effects of internet path selection," in *Proc. ACM SIGCOMM'99*, Aug. 1999, pp. 289–299.

Xing Jin received the B.Eng. degree in computer science and technology from Tsinghua University, Beijing, China, in 2002 and the Ph.D. degree in computer science and engineering from The Hong Kong University of Science and Technology (HKUST), Kowloon, Hong Kong, in 2007.

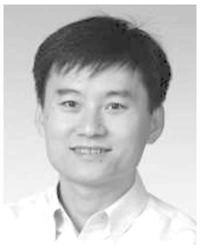
He is currently a Member of Technical Staff in the Systems Technology Group at Oracle, Redwood Shores, CA. His research interests include distributed information storage and retrieval, peer-to-peer technologies, multimedia networking, and Internet topology inference.

Dr. Jin is a member of Sigma Xi and the IEEE COMSOC Multimedia Communications Technical Committee. He has been on the editorial board of the *Journal of Multimedia* since 2006 and the *Canadian Journal of Pure and Applied Sciences* since 2007. He was awarded the Microsoft Research Fellowship in 2005.



Kan-Leung Cheng received the M.Phil. degree in computer science and engineering from The Hong Kong University of Science and Technology (HKUST), Kowloon, Hong Kong, in 2006. He is currently pursuing the Ph.D. degree in the Department of Computer Science at the University of Maryland, College Park.

His research interests include protocols, algorithms, and architectures for networking systems.



S.-H. Gary Chan (S'89–M'98–SM'03) received the B.S.E. degree (Highest Honor) in electrical engineering from Princeton University, Princeton, NJ, in 1993, with certificates in applied and computational mathematics, engineering physics, and engineering and management systems, and the M.S.E. and Ph.D. degrees in electrical engineering from Stanford University, Stanford, CA, in 1994 and 1999, respectively, with a minor in business administration.

He is currently an Associate Professor with the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology (HKUST), Kowloon, Hong, and an Adjunct Researcher with Microsoft Research Asia, Beijing, China. He was a Visiting Assistant Professor in Networking with the Department of Computer Science, University of California, Davis, from 1998 to 1999. His research interests include multimedia networking, peer-to-peer technologies and streaming, and wireless communication networks.

Dr. Chan is a member of Tau Beta Pi, Sigma Xi, and Phi Beta Kappa. He was a William and Leila Fellow at Stanford University during 1993–1994. At Princeton University, he was the recipient of the Charles Ira Young Memorial Tablet and Medal and the POEM Newport Award of Excellence in 1993. He served as a Vice-Chair of IEEE COMSOC Multimedia Communications Technical Committee from 2003 to 2006. He was a Guest Editor for the *IEEE Communication Magazine* (Special Issues on Peer-to-Peer Multimedia Streaming), 2007, and Springer *Multimedia Tools and Applications* (Special Issue on Advances in Consumer Communications and Networking), 2007. He was Co-Chair of the Multimedia Symposium for IEEE ICC (2007). He was the Co-Chair for the workshop on Advances in Peer-to-Peer Multimedia Streaming for the ACM Multimedia Conference (2005) and the Multimedia Symposia for IEEE GLOBECOM (2006) and IEEE ICC (2005).