Implementation and Performance Measurement of an Island Multicast Protocol

Kan-Leung Cheng K.-W. Cheuk S.-H. Gary Chan
Department of Computer Science
The Hong Kong University of Science and Technology
Clear Water Bay, Kowloon, Hong Kong
Email: {klcheng, rcheuk, gchan}@cs.ust.hk

***Abstract*— With the availability and penetration of multicast-capable routers, many local networks in today's Internet are multicast-capable. However, achieving global IP multicast is still hindered by many management and technical difficulties. This is because routers interconnecting these local multicast-capable networks, or so-called "islands," are often either multicast-incapable or multicast-disabled. Traditional application-level multicast (ALM) only makes use of unicast connections to form delivery trees and has not fully taken advantage of the local multicast capability of an island. As a result, these protocols are not very efficient.**

In order to achieve efficient global multicast, we propose and study Island Multicast (IM) where unicast connections are used between islands while IP multicast is used within islands. We present the detailed mechanisms of the IM centralized approach. IM is simple to implement and is based on minimum spanning tree, and hence is applicable to many-to-many communication. We have implemented the protocol and done real measurements on PlanetLab. We show that our protocol significantly improves network performance (in terms of stress, delay and nodal degrees) as compared to using ALM alone.

I. Introduction

With the availability and penetration of multicast-capable routers, many local networks in today's Internet are already multicast-capable. However, routers interconnecting local multicast-capable domains, or so-called "islands," are often multicast-incapable or multicast-disabled. This is mainly due to many technical, management, political and security reasons. As a result, even though IP multicast is efficient, achieving global multicast at the network layer is still not possible today.

In order to achieve global multicast, application-level multicast (ALM) has recently been proposed, where the group members form an overlay network and data packets are relayed from one member to another via unicast. Traditional ALM protocols, though work well, are based on unicast connections and hence have not fully taken advantage of the local multicast capability of islands. Their performance are mostly independent of the number of multicast-capable routers or domains in the network, and hence are not very efficient in terms of delay and bandwidth.

This work was supported, in part, by the Areas of Excellence (AoE) Scheme on Information Technology funded by the University Grant Council in Hong Kong (AoE/E-01/99) and by grants of the Research Grant Council (HKUST6199/02E & HKUST6156/03E) in Hong Kong.

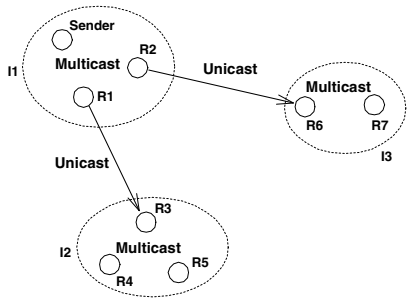


Fig. 1. Island Multicast.

It would, therefore, be beneficial if ALM could make use of the local multicast capabilities in building trees. We hence propose and investigate a scheme called Island Multicast (IM) that integrates IP multicast with ALM. In IM, hosts belonging to the same island use IP multicast for data delivery. To achieve global multicast, hosts belonging to different islands use unicast to interconnect together. We call such overlay link a "bridge".

We illustrate the idea of IM in Figure 1, where seven hosts (labeled R_1 through R_7) belonging to the same multicast group are distributed in three islands, I_1, I_2, and I_3.[1] I_1 is the parent of I_2 and I_3. R_1 and R_3 are the *bridge-nodes* for inter-island delivery (from I_1 to I_2). Likewise, R_2 and R_6 are the bridge-nodes for delivery from I_1 to I_3. We call R_2 and R_6 the *egress* and *ingress* bridge-nodes, respectively, from I_1 to I_3. The sender transmits packets via IP multicast to receivers R_1 and R_2, which in turn forward the packets to islands I_2 and I_3, respectively. R_3, upon receiving a data packet from I_1, relays it via IP multicast to R_4 and R_5. Similarly, R_6 multicasts its received packets to R_7.

It is evident that by making use of the local IP multicast capability in ALM, network efficiency can be greatly improved. The efficiency includes:

- Lower network stress — Since IP multicast does not duplicate unnecessary packets in an island, a lower overall network stress can be achieved as compared to using ALM alone.
- Lower end-to-end delay — Because most IP multicast

[1] In this paper, we use the terms node, host and member interchangeably.

protocols are based on shortest path routing, IM achieves lower delay than using ALM.

- Higher robustness — In traditional ALM, the failure or leaving of a node often leads to an outage of data flow to its descendant nodes. In IM, since fewer nodes are involved in packet forwarding, the system is more robust toward node failure or group dynamics.

We study and evaluate in this paper a centralized version of IM. Applications such as multi-party video conferencing are often characterized by many sources of smaller group size. Centralized IM is hence applicable in this setting. In centralized IM, a central controller computes multicast trees and fixes tree partitions. We discuss in this paper the join/leave, data delivery and fault recovery mechanisms for this protocol. This protocol, though less scalable than the distributed one [1], is simpler to implement and has higher bandwidth efficiency. We have implemented the protocol and using real measurements on PlanetLab, we show that centralized IM indeed can significantly improve bandwidth efficiency and reduce delivery latency as compared to using ALM alone.

We briefly review previous work as follows. Most proposed ALM protocols such as Narada, NICE, Delaunay Triangulation (DT), ALMI, etc., assume that none of the routers are multicast-capable and hence do not make full use of the underlying IP multicast capability [2], [3], [4], [5]. Our work addresses how to make use of the local multicast capability to improve network efficiency. The work on distributed IM has been reported in [1]. We extend the work here by studying a simpler centralized version. We also implement the protocol and report in this paper its performance measurement in PlanetLab.

The idea of IM is similar to other previously proposed protocols such as Scattercast, YOID, AMT, Universal Multicast (UM) and Subnet Multicast (SM) [6], [7], [8], [9], [10]. While many of these protocols require some special nodes (such as proxies or routers) or manual configuration of inter-host connections, IM is fully autonomous and does not need that. The major complexity of UM comes from eliminating routing loops, while IM is inherently loop-free and hence is simpler. SM is based on a star topology with only one level of tree. This increases the physical link stress of the network. As shown in our experiment results, IM achieves good stress performance due to its tree-based approach.

This paper is organized as follows. We first present in Section II the mechanisms of centralized IM, followed by an evaluation of it in Section III and conclusion in Section IV.

II. Centralized Island Multicast

In this section, we first present an overview of centralized IM. We then discuss in detail the basic mechanisms in terms of join and leave operations, tree computation, neighbor monitoring and parent selection.

A. Overview

Centralized IM uses a controller to compute a multicast tree for each group. Each multicast tree connects all group members in the group. Members are allowed to send data to all the other members in the group (many-to-many communication). It is built so as to minimize the total delay of the inter-island overlay connections. Note that in the tree, an edge between two members within the same island represents the *logical* relationship for fault recovery and tree maintenance, not the actual data flow.

Whenever the controller computes a new tree, the tree structure information is distributed with a new version number. The number avoids clashing and hence routing loops with the previous trees.

B. Join Operations

A joining node first detects the existence of an existing member in the same island by sending a PING message to an IP-multicast address which is uniquely determined by the session ID. A member, if any, receiving the message replies (using IP-multicast) a PING_REPLY message consisting of its member ID to the same multicast address. The joining node then knows a member of its island. If no reply is received after a few trials (say, three) the joining node concludes no members in its island.

The joining node sends a JOIN_SESSION message consisting of the member ID of its island peer (if any) to the controller. The controller then replies with a JOIN_SESSION_REPLY message which consists of a unique member ID and the parent of the joining node.[2] (How parent nodes are chosen will be discussed later.)

The joining node then sends a GRAFT message to its parent. The parent may reject the request if it is overloaded, in which case the joining node sends a REJOIN_SESSION message to the controller requesting a new parent. The controller then assigns it a new parent with a NEW_PARENT message.

C. Leave Operations

A leaving member sends a LEAVE_SESSION message to the controller, which in turn fixes the tree partition by assigning each of its children a new parent. The controller then sends a NEW_PARENT message to each of the children, which then sends a GRAFT message to its new parent. After restructuring the tree, the controller sends a LEAVE_SESSION_REPLY message to the leaving member, which then stops forwarding data packet and complete the leave operation.

D. Tree Computation and Maintenance

In order to continuously improve tree performance, the controller, based on round-trip time (RTT) measurements reported by members, periodically runs a minimum spanning tree algorithm (Modified MST-Prim algorithm). To avoid overloading of some nodes, we impose an adjustable application-dependent degree limit on all nodes during the tree construction. If the application requires high bandwidth, the degree limit should be set low (so that link bandwidth is shared with fewer members). The consequent spanning tree then would be sub-optimal in

[2]Though the IP address of the host may be used, using a unique ID would reduce the field length in message headers

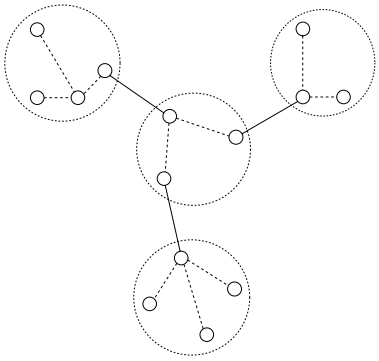


Fig. 2. A centralized IM tree.

cost but more load-balanced. The edge between nodes of different islands is assigned a weight equal to their estimated RTT (infinity if unknown). All other edges (i.e. within the same island) are assigned weight -1. In this way, nodes within the same island are joined together as a tree.

Note that within the same island, an edge between two nodes does *not* indicate the forwarding path of data packets; data packets are forwarded according to IP multicast. These edges only imply parent-child *logical* relationship, which are for tree maintenance and fault recovery purposes only. The edges spanning across islands, however, do indicate packet forwarding paths.

We show in Figure 2 an example of such a spanning tree with four islands. Within the islands, packets are multicast without following the edges (the broken line). On the other hand, between islands packets are forwarded via unicast according to the edges (the solid line).

The controller computes a new tree periodically (e.g., every 30 seconds). If its total RTT as compared to the old tree reduces to a certain threshold, the new tree is adopted; otherwise, the newly computed tree is rejected. If the new tree is adopted, the controller informs each of the nodes of its new parent and children via a NEW_PARENT message. The use of the threshold reduces tree instability and the associated overhead caused by frequent tree re-configurations. Our preliminary experiments show that a threshold value of 95% of the original total RTT provides reasonable stability without sacrificing much the efficiency of the resultant tree.

E. Neighbor monitoring

In order to obtain updated RTT measurements between every node-pair, the controller continuously gives each node a neighbor list to ping. The node, based on the list, conducts RTT measurements and identifies some of the failed members. The node then reports these measurements to the controller, which may then reply to the node with another list. Each node also periodically pings its parent and children via unicast to detect node failure and to find out the distance. Upon detecting the failure of its parent, a node requests the controller for a new parent by sending a REJOIN_SESSION message.

To reduce overhead, the controller limits the length of the neighbor list (e.g., 5 in the current implementation). The RTT reporting rate of member is inversely proportional to the group size to make it more scalable. In selecting which nodes to list, the controller selects those with unknown or older RTT values. The controller also removes unresponsive nodes from its tree computation.

TABLE I

SIZES OF ALL CONTROL MESSAGES.

Control Message Type	Size (bytes)
Create Session	44
Create Session Reply	10
Close Session	10
Session Close	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
Ping	10
Ping Reply	8
Peer List	336

F. Parent Selection

In choosing the new parent for a node, the controller favors nodes with the following properties:

- In the same island — The controller would try to choose a node that is in the same island. Choosing a new parent in another island means a separate unicast stream. Furthermore, a node in another island is often farther.
- High responsiveness to ping messages — This is because a node not responsive to ping messages suggests that it is busy or overloaded.
- Low nodal degree — A node with a low nodal degree is preferable as forwarding load would be more evenly distributed among all nodes.

III. EXPERIMENTS

We have implemented the centralized IM as mentioned above. We study its performance by conducting experiments on the PlanetLab testbed [11]. In this section, we discuss the experiments followed by the measurement results. The codes (in terms of library) can be found in [12].

A. Experiment Environment

Our experiments consist of three phases: join phase, stabilization phase and leave phase. In the join phase, a number of hosts randomly join a session over a certain time period. The hosts stabilize into an appropriate overlay tree during the stabilization phase. The overlay tree is recorded and all RTT measurements are made in this period. After this period, is the leave phase, when all members leave the session randomly over a certain time period. The hosts are randomly chosen in the US, Europe and Asia. From our experiment, we find that in general a multicast domain contains a number (between two and three) hosts.

The following metrics are used to evaluate the protocol performance:

- Join and leave latency — This measures how long the join and leave operation take. Join latency is

the delay between the sending of the first PING message of a host to detect island and the receiving of JOIN_SESSION_REPLY message from the controller. Leave latency is the delay between sending of LEAVE_SESSION message of a host and the receiving of LEAVE_SESSION_REPLY from the controller. It is desirable to have low latency even when the number of members is large. We report the mean and distribution of the latencies of all members.

- Relative Delay Penalty (RDP) — RDP is also known as "stretch," which is defined as the ratio of source-to-receiver overlay latency to source-to-receiver unicast latency. This measures how well an overlay tree matches the underlying multicast topology based on shortest path routing. We use application-layer ping (instead of ICMP ping) to obtain the distances between two hosts because some PlanetLab nodes are behind firewall which blocks all ICMP packets. Furthermore, the values obtained this way are closer to the actual transmission delay as it takes into account the host processing power. We report the mean and distribution of all-pair RDP values.
- Out-degree (Forwarding load) — This is the number of identical packets a node sends. Note that sending a packet into a multicast address counts as one forwarding load. We report the mean and distribution of the out-degree of all members.
- Control traffic overhead — This is defined as the bandwidth used to transmit (send or receive) control packets. The sizes of all control messages are listed in Table I. We report the mean and distribution of the bandwidth overheads for all members. The overhead should remain low even when the number of members increases.

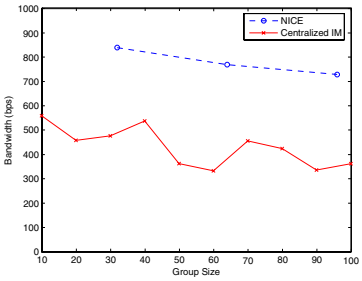
B. Measurement Results

In this section, we present measurement results on our protocol.

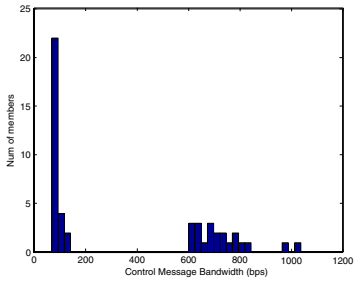
In Figure 3(a), we show the average join and leave latencies and the average RTT values between the controller and all members. Join latency is higher than leave latency because members have to take some time to detect island before contacting the controller. The latencies do not sensitively depend on the group size because the joins and leaves happen randomly. If many members join or leave at the same time, the controller will be overwhelmed and that would significantly increase the latencies.

The distribution of the latencies are shown in Figure 3, given a group size of 50. For join, most of the members experience latencies less than the average, while some of them suffer from high latencies. This may be due to island detection (timeout) and random overload condition at the controller. For leave latency, the tail is shorter, with most of them cluster around the average.

In Figure 4, we show the control overhead of our protocol. In Figure 4(a), we compare it with NICE [3]. The control overhead of NICE has been shown to be low as compared with other efficient protocols like Narada [2]. The overhead of our protocol is lower than NICE and remains low even for large group size. We achieve the low control overhead because our RTT report (from members to controller) frequency decreases with the group size.



(a) Average control message bandwidth over all members.



(b) Mean control message bandwidth distribution (for a group size of 50).

Fig. 4. Mean and distribution of control message bandwidth.

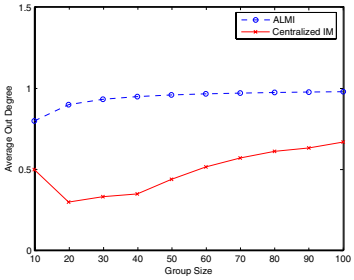
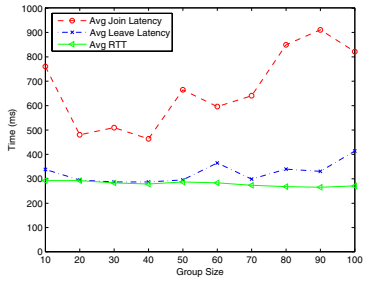


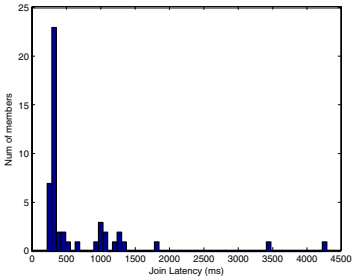
Fig. 5. Mean out-degree versus group size.

The distribution of the overhead is shown in Figure 4(b). Clearly, it is a bimodal distribution. The low overhead (around 100 bits/s) corresponds to the members which are leave nodes on the overlay tree, while the remainder corresponds to those internal nodes. Note that every member (except the root) pings its parent regularly (every second) to ensure that it is alive and the tree is connected. Therefore, the more children a node has, the more ping messages it needs to deal with. If we reduce the nodal degree limit, lower control overhead would result at the cost of RDP.

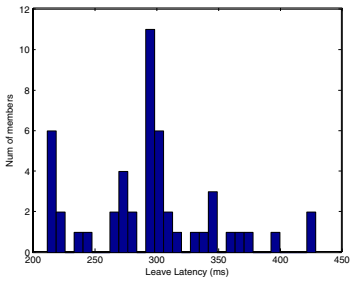
In Figure 5, we compare the average out-degree of IM and ALMI [5]. The out-degree of IM is always lower than ALMI. In term of out-degree distribution, the spread is little (and



(a) Mean RTT and latencies versus group size.

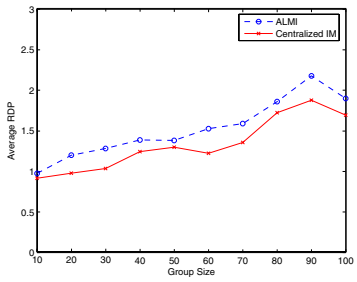


(b) Join latency distribution.

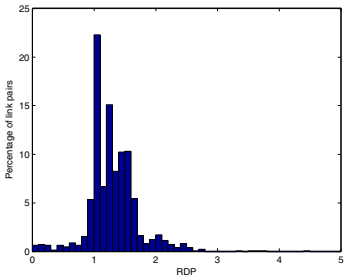


(c) Leave latency distribution.

Fig. 3. Mean and distribution(for a group size of 50) of join and leave latency.



(a) Mean all-pair RDP versus group size.



(b) All-pair RDP distribution (group size of 50).

Fig. 6. Mean and distribution of all-pair RDP.

hence uninteresting and not shown). For example, for a group size of 50, we observe 34 members with degree of zero (either at the leaves of the tree or within the island), 10 members with degree 1 (bridge-nodes) and 6 members with degree 2. The out-degrees are low because the forwarding load is well-balanced among all the members.

Figure 6(a) depicts the RDP performance of IM as compared with ALMI. IM achieves lower RDP since it takes advantages of the underlying IP multicast. In Figure 6(b), we show the distribution of the all-pair RTT values of IM (for a group size of 50). Though most of the pairs are around the average value, some RDP is very high, corresponding to the nodes at the leaves. Furthermore, some nodes even have RDP less than one. This is because in real Internet, there are route inefficiency where shorter alternate paths may exist between two hosts. Therefore, an indirect path between two hosts via some other hosts may be shorter than a direct path between them.

IV. Conclusion

The Internet today consists of multicast-capable "islands" interconnected by multicast-incapable routers. In order to enable global multicast and to achieve network efficiency, these islands should be interconnected by unicast connections while multicast capability should be used within an island taken advantages of. This is called Island Multicast (IM) in our study. In this paper, we have presented the design and implementation of a centralized version of IM. It uses an additional controller to centrally compute the multicast tree. We present the basic mechanisms of the protocol, and have implemented it in PlanetLab. Our experimental measurements show that our protocol and implementation is efficient in terms of average stress, stretch, latencies and control overhead.

References

[1] K.-W. Cheuk, S.-H. Chan, and J. Y.-B. Lee, "Island Multicast: The Combination of IP-Multicast with Application-Level Multicast," in *Proceedings of IEEE International Conference on Communications (ICC)*, June 2004, pp. 1441–1445.

[2] Y.-H. Chu, S. G. Rao, S. Seshan, and H. Zhang, "A Case for End-system Multicast," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 8, pp. 1456–1471, Oct. 2002.

[3] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable Application Layer Multicast," in *Proceedings of ACM SIGCOMM*, Aug. 2002.

[4] J. Liebeherr, M. Nahas, and W. Si, "Application-Layer Multicasting with Delaunay Triangulation Overlays," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 8, pp. 1472–1488, Oct. 2002.

[5] D. Pendarakis, S. Shi, D. Verma, and M. Waldvogel, "ALMI: An Application Level Multicast Infrastructure," in *Proceedings of the 3rd USENIX Symposium on Internet Technologies and Systems*, Mar. 2001.

[6] Y. Chawathe, S. McCanne, and E. Brewer, "An Architecture for Internet Broadast Distribution as an Infrastructure Service," *PhD thesis, University of California, Berkeley*, Dec. 2000.

[7] P. Francis, "YOID: Your Own Internet Distribution," Dec. 2004. [Online]. Available: http://www.icir.org/yoid/

[8] D. Thaler, M. Talwar, L. Vicisano, and D. Ooms, "IPv4 Automatic Multicast Without Explicit Tunnels (AMT)," *Internet Draft, IETF*, Feb. 2004.

[9] B. Zhang, S. Jamin, and L. Zhang, "Universal IP multicast delivery," in *Proceedings of the International Workshop on Networked Group Communication (NGC)*, Oct. 2002.

[10] J. Park, S. J. Koh, S. G. Kang, and D. Y. Kim, "Multicast Delivery Based on Unicast and Subnet Multicast," *IEEE Communications Letters*, vol. 5, no. 4, pp. 1489–1499, Apr. 2001.

[11] "Planetlab," http://www.planet-lab.org/.

[12] "Library of centralized island multicast," http://mwnet.cs.ust.hk/im/.