

Linkcast: Fast and Scalable Multicast Routing Protocol

Mozafar Bag-Mohammadi, Siavash Samadian-Barzoki, and Nasser Yazdani

Router Lab., Dept. of EE & Computer Eng., Univ. of Tehran, Tehran, Iran
{mozafarb, s.samadian}@ece.ut.ac.ir, yazdani@ut.ac.ir

Abstract. The current multicast routing protocols require state maintenance in the on-tree routers in order to forward multicast packet properly. Therefore, the routers memory may be easily saturated when there are very large number of low to moderate size multicast groups. In contrast, the explicit multicast protocols offer a stateless design using header space of multicast data packets. In this paper, we introduce new stateless scheme called Linkcast that efficiently eliminates processing overhead of explicit multicast protocols like Xcast. The Linkcast represents the multicast tree by encoding its counterpart links. Simulation results show that Linkcast completely eradicates the required unicast lookups in explicit multicast protocols with less header size overhead.

1 Introduction

IP multicast significantly enhances bandwidth utilization eliminating duplicate packets crossing the network links. Traditional multicast routing protocols like DVMRP [11] and PIM-SM [12] require state maintenance in the on-tree routers, commonly known as Multicast Forwarding Table (MFT), in order to forward multicast packet properly. The state maintenance is performed in a per-group basis at on-tree routers. Therefore, the routers may easily run out of memory when there are very large number of low to moderate size multicast groups.

Although current multicast routing protocols are able to support small number of large multicast groups, they don't scale well when supporting very large number of low to moderate size multicast groups [1]. There are many small to moderate size multi-party applications such as video and audio conferencing, IP telephony and network games, which can not be well serviced in large scale by the current model. Xcast [1] [2] [3] [4], its variations Xcast+ [5] [6], Bcast [8] and ERM [7] are designed to serve this class of applications in a scalable manner.

In Xcast and Xcast+, a list of destination IP addresses is sent with each multicast data packet. In Bcast and ERM, IP addresses of the receivers and branching points of multicast tree are sent with each Bcast packet. These protocols trade-off the header size and processing power to obtain scalability and simplicity. The processing overhead consists of two parts: the number of required unicast lookups and the header processing. In Linkcast, we achieve the same goal with very small processing overhead.

In Linkcast, multicast sender encodes the tree listing tree links in a proper way. We present two complementary coding schemes, which can be used interchangeably in different circumstances. The first encoding scheme is appropriate when distances between the branching routers are short. Otherwise, the other scheme is suitable. Simulation results for various network graphs show that the header size overhead in Linkcast is comparable with Bcast and Xcast. This is more noteworthy if one considers negligible Linkcast forwarding cost.

In section two, we briefly describe the related work. Then, the key components of Linkcast are discussed in section three. Section four deals with the simulation results. Finally, we conclude the paper in section five.

2 Related Work

Xcast [1] [2] [3] [4] was originally designed to overcome scalability and deployment problems of the current multicast routing protocols. In Xcast, multicast sender simply encodes the list of receiver IP addresses in a special header and sends it with all data packets. Therefore, at each router, each destination in the Xcast header of the packet requires a unicast lookup. This processing overhead limits the number of possible destinations to a very small number (less than 10 [14]). Xcast+ [5] [6] introduces a simple modification to Xcast combining it with the well-known IGMP (Internet Group Management Protocol) [13] protocol.

Sender Initiated Multicast (SIM) [9] is another Xcast like scheme which reduces Xcast forwarding cost. SIM capable routers construct an MFT-like table to forward multicast packets. This state maintenance severely limits SIM scalability. Simple Explicit Multicast (SEM) [10] uses the receivers list to construct the multicast distribution tree. SEM also suffers from scalability issues due to the state maintenance at branching routers of the multicast tree.

Bcast [8] efficiently removes unnecessary lookups that take part in Xcast and Xcast+ forwarding mechanism. In Bcast, IP addresses of the receivers and branching points are encoded in Bcast header. Furthermore, destination IP address field in IP header is filled with address of the next branching point. This enables Bcast to use unicast forwarding between each branching point pair.

3 Linkcast

3.1 Tree Encoding

The main objective of tree encoding method is to minimize size of generated code. We believe that the arrangement of the tree nodes is the most important parameter that affects the code size. The node in a multicast tree can be partitioned into three categories based on the node degree [15] : 1- Member nodes which have degree 1, 2- Relay nodes which have degree 2, 3- Branching nodes which have degree 3 or more. We introduce two encoding schemes that consider different mix of relay and branching nodes. The first scheme is appropriate when there is large number of relay nodes in the tree. We call this mode Sparse

Branching Mode or SBM for short. The second is suitable for a tree with high average node degree. The second scheme is called Dense Branching Mode or DBM shortly. In either case, the generated code is placed in a special header between IP header and transport layer header.

We explain SBM and DBM encodings through an example. The example tree and its corresponding codes are shown in Fig. 1. The number on each link is the local link ID (see subsection 3.2). In both method, we differentiate between different types of link by means of link ID coding.

Each link in SBM may have one or more pointers which point(s) to the next link(s) in the tree. Obviously, the member links e.g. m2 need not to have any pointer. The incoming link of a relay node e.g. f has only one next link. We arrange the tree links so that the pointer value for this type of links is always one eliminating the need of storing a pointer for relay links. For incoming link of a branching node, the branching factor must be determined. In addition, we need to store a pointer for each of its branches.

In the DBM, each links has a pointer which points to the end node of the link. For member links, the end node is not required to be in the code. The outgoing link(s) of each node begin from the node itself to the next node in the code. The nodes are represented with a special value in the code. The number of required bytes in the SBM and DBM coding can be calculated as:

$$\text{SBM: } (\# \text{ of tree branches}) + \text{total_br} + (\# \text{ of branching nodes}) \quad (1)$$

$$\text{DBM: } (\# \text{ of tree branches} - 1) + \text{total_br} + (\# \text{ of relay nodes}) \quad (2)$$

where total_br is total branching factor of branching nodes. In the above example the code sizes are 30B and 34B for SBM and DBM coding respectively.

The tree has a general pointer P1 which points to current link in the code. When a router receives a Linkcast packet, it examines the value of P1. Then, it finds the next link(s) by interpreting the tree code. Finally, it modifies P1 value accordingly and forwards the packet.

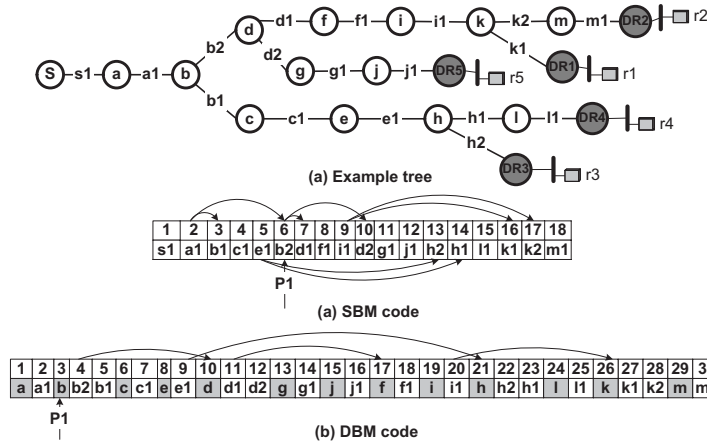


Fig. 1. An example tree and resulted tree codes

3.2 Gathering Tree Information

The sender collects the tree information from the receivers using their *Join* messages. Each new member sends a *Join* message to the source of the multicast session. All routers on the path must examine the message and append incoming link ID, outgoing link ID and their IP addresses to the packet. Having the path information from receivers to the source, the source can construct reverse shortest path tree. The receivers must repeat their *Join* messages periodically in order to refresh their state in sender. Thus, the sender can repair the multicast distribution tree against temporary route changes. If the sender misses three consecutive *Join* messages from a receiver, it will remove the receiver from the tree code. The receivers can also immediately depart the multicast session by sending *Leave* messages to the sender.

3.3 Multi-access Links

In a multi-access link, the broadcast nature of the link is source of ambiguity. Since possible end nodes of the link are more than one, it is not clear that what node actually belongs to the tree. To solve this ambiguity, we decompose a multi-access link into $n * (n - 1) / 2$ virtual links connecting each pair of nodes where n is the number of nodes on the link. Each virtual link has a unique ID. All nodes must know the ID of each virtual link. Therefore, the nodes must run a simple protocol to agree on the ID of the resulted virtual links.

3.4 Source Branching

Since the sender generates the encoded tree, it is possible to produce different codes for each sub-tree rooted at the sender when the sender itself is a branching point. This significantly reduces size of the encoded tree. Bcast has the same property as well. This allows them to support larger number of receivers compared to Xcast+. It is worth noting that formula 1 and 2 in subsection 3.1 are applied to largest sub-tree rooted at the sender. Simulation results show that this capability of Linkcast results in lower header size than Xcast+ in most cases.

4 Simulation Result

We have evaluated the Linkcast header overhead in comparison with Xcast and Bcast using NS-2 environment [19]. We performed two sets of experiments, one for small networks and another for large networks. Any router with at least one member is counted as a single receiver. Therefore, the actual number of manageable receivers is slightly more.

4.1 Small Networks

Small networks are generated based on Doar-Leslie random graph model using GT-ITM network topology generator [16]. We changed the network size from 20

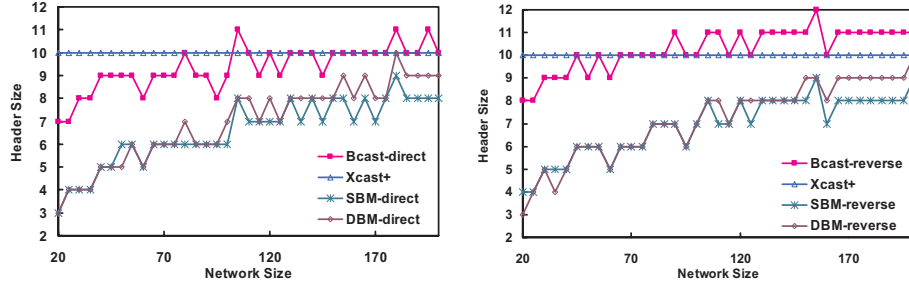


Fig. 2. The header size comparison based on network size in Doar-Leslie flat random topology model.

to 200 while the average node degree is fixed approximately at 3.5. The group size is fixed at 10 because Xcast+ is not supposed to support more than 10 receivers [14]. For each generated network, we intentionally introduced 50% asymmetry in the network links. Then, we considered the reverse and forward shortest path trees in Linkcast and Bcast methods, which resulted in five different graphs. Finally, all plots are normalized to Xcast+. As Fig. 2 suggests, the header size of Linkcast is always smaller than Xcast and Bcast header. Furthermore, the differences between the direct and reverse versions of Linkcast are negligible.

4.2 Large Networks

For the large networks, we generated random graphs based on Barabasi-Albert [17] topology model using BRITE [18] network topology generator. The Barabasi-Albert model takes the power law relationship of Internet into account. We fixed the network size at 1204 nodes and performed the simulations with various group sizes ranging from 10 to 500. The average node degree is fixed at 4. The simulation results with Barabasi-Albert graph model is shown in Fig. 3. As can be seen in this figure, Linkcast has significantly less header overhead than Xcast and Bcast. We believe that Linkcast can support very large number of moderate size group (less than 50). Linkcast header size for 50 receivers is 120B.

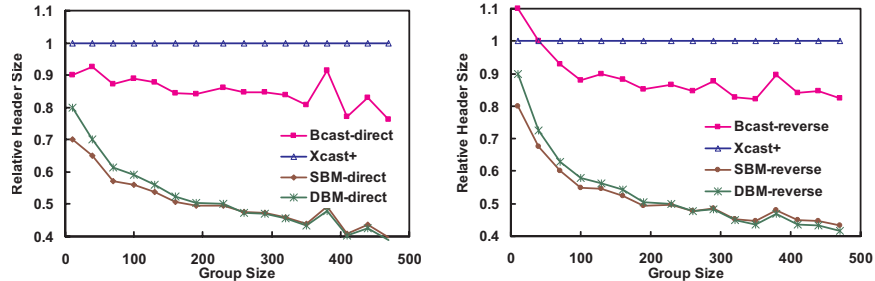


Fig. 3. The header size comparison based on group size in Barabasi-Albert flat random topology model.

5 Conclusion

Traditional multicast protocols fail to support very large number of any size multicast groups mainly due to their state-full design. Although Xcast and Bcast benefit from stateless design, they have two main difficulties to support moderate size multicast groups. First, their header size grows rapidly. Second, they need more unicast lookups in intermediate on-tree nodes when number of multicast members increase. Linkcast solve the header size problem of Xcast and Bcast without having to perform any form of table lookup. We believe that Linkcast is more appropriate than Xcast and Bcast in supporting huge amount of moderate and fairly large size multicast groups (less than 70). Linkcast fill the gap between Bcast, Xcast and traditional multicast for moderate size multicast groups.

References

1. R. Boivie, et al, "Explicit Multicast (Xcast) Basic Specification, IETF Internet-Draft, 2003
2. R. Boivie, N. Feldman, "Small Group Multicast", IETF Internet-Draft, July 2000.
3. R. Boivie, N. Feldman, C. Metz "Small Group Multicast: A New Solution for Multicasting on the Internet", Internet Computing, Vol. 4, No. 3, May/June 2000.
4. D. Ooms, W. Livens, "Connectionless Multicast", IETF Internet-Draft, April 2000.
5. M.K. Shin, Y.J Kim, K.S Park, S.H Kim, "Explicit Multicast Extension (Xcast+) Supporting Receiver Initiated Join", IETF Internet-Draft, October, 2002.
6. M.K. Shin, Y.J Kim, K.S Park, S.H Kim, "Explicit Multicast Extension (Xcast+) for Efficient Multicast Packet Delivery", ETRI journal, Vol. 23, No. 4, Dec. 2001.
7. J. Bion, D. Farinacci, M. Shand, A. Tweedly, "Explicit Route Multicast (ERM)", IETF Internet-Draft, June 2000.
8. M.Bag-Mohammadi, S.Samadian-Barzoki, N.Yazdani, "Using Branching Points for Multicast Data Distribution", submitted to SIGCOMM 2004.
9. V. Visoottiviseth, H. Kido, Y. Kadobayashi, S. Yamaguchi, "Sender-Initiated Multicast Forwarding Scheme", Proc. of IEEE ICT'2003, Tahiti, Feb. 2003.
10. A. Boudani, B. Cousin, "SEM: A New Small Group Multicast Routing Protocol", Proc. of IEEE ICT2003, Tahiti, Feb. 2003.
11. D.Waitzman, C.Partridge, S.Deering, "Distance Vector Multicast Routing Protocol", RFC 1075, Nov.1988
12. S.Deering, et al, "The PIM architecture for wide-area multicast routing", IEEE/ACM Trans. on Networking, Vol.4, No.2, April 1996
13. B. Cain, S. Deering, I. Kouvelas, B. Fenner, A. Thyagarajan, "Internet Group Management Protocol, Version 3" RFC 3376, October 2002.
14. O. Paridaens, D. Ooms, "Security Framework for Explicit Multicast", IETF Internet-Draft, November 2000.
15. J. Pansiot and D. Grad, "On routes and multicast trees in the Internet", ACM Computer Communication Review, vol. 28, no. 1, pp. 41-50, Jan.1998.
16. E. W. Zegura, K. Calvert, S. Bhattacharjee., "How to model an Internetwork.", Proc. of IEEE Infocom'96, San Francisco, CA
17. A.L. Barabasi, R. Albert, "Emergence of Scaling in Random Networks". Science, 286:509-512, October 1999.
18. A. Medina, A. Lakhina, I. Matta, J. Byers "BRITE: An Approach to Universal Topology Generation", In Proc. of MASCOTS'01, Cincinnati, Ohio, August 2001.
19. "The Network Simulator - ns - 2", <http://www.isi.edu/nsnam/ns/>