A Scalable Destination-Oriented Multicast Protocol with Incremental Deployability

Xiaohua Tian, *Member, IEEE* and Yu Cheng, *Senior Member, IEEE*

Abstract—In this paper, we develop a scalable destination-oriented multicast (DOM) protocol for computer networks where the routers have enhanced intelligence to process packets. The basic idea of DOM is that each multicast data packet carries explicit destinations information, instead of an implicit group address, to facilitate the data delivery. Based on such destinations information, each router can compute necessary multicast copies and next-hop interfaces. A fundamental issue in DOM is to constrain the bandwidth overhead due to explicit addressing, which is tackled with a Bloom-filter based design. Our design incorporates the reverse path forwarding (RPF) concept and the BGP routing information, so that DOM can work efficiently in practical networking scenarios especially with asymmetric inter-domain routing. A critical issue in Bloom-filter based design is the issue of forwarding loop due to false positives. We propose an accurate tree branch pruning scheme, which equips the DOM the capability to completely and efficiently remove the false-positive forwarding loop. Furthermore, we study how the DOM can be deployed in an incremental manner over a network, in which only a small fraction of the routers have DOM-aware intelligence while others are legacy routers. We present extensive simulation results over a practical topology to demonstrate the performance of DOM, with comparison to the traditional IP multicast and the free riding multicast (FRM) protocols.

Index Terms—Multicast, scalability, Next-Generation Internet, incremental deployability

1 Introduction

A scalable multicast protocol has been an open research issue in recent two decades, which could impact the wide deployment of multimedia applications over the next-generation Internet requires one-to-many or many-to-many communications. Proposed by Deering in 1988, IP multicast delivers the shared data along a network-layer based tree structure constructed using a distributed multicast routing algorithm [1], [4], [5], [10]. It is bandwidth efficient in data delivery but poorly scalable in managing the multicast tree [10], [11], [14], [15], since each router needs to maintain the multicast forwarding states for every group passing through; the messaging overhead and the memory cost grow linearly with the number of multicast groups being supported by the router. The more recent overlay multicast establishes the data-dissemination structure at the application layer [12], [13], [24], wherein each overlay link is an end-to-end unicast path between two hosts. Although convenient for deployment as the underlying unicast infrastructure needs no modification, overlay multicast induces redundant traffic at the network layer [12]: it is common that separate overlay links pass through the common physical links in the underlying transport network.

Recently, several schemes, e.g., recursive unicast approach to multicast (REUNITE) [14], [15], explicit multicast (Xcast) [26], free riding multicast (FRM) [27], multicast with adaptive dual-state (MAD) [16], line speed publish/subscribe inter-networking (LIPSIN) [17] and BloomCast [18], have been proposed to improve network-layer multicast service. While these schemes vary in details, they share the same design philosophy: the network routers are enhanced with extra intelligence to exploit more information in the packets and execute more complex operations to realize the multicast functionality. Although these schemes have some favored characteristics, the scalability issue of IP multicast is not thoroughly resolved.

In this paper, we show that the enhanced intelligence of routers can facilitate the development of a scalable destination-oriented multicast (DOM) protocol. The key idea of DOM is that the packet carries the explicit destination addresses, which will facilitate the multicast forwarding process in network routers [19]. To limit the bandwidth overhead for such explicit addressing, we have developed a practical DOM protocol based on Bloom filter [20]. The Bloom filter is a randomized data structure for representing a set and supporting membership queries thus improves the space efficiency in DOM packets [28], [29]; DOM explicit addresses in the packet are encoded in the format of the Bloom filter to reduce the bandwidth overhead. However, the design in [20] is not efficient in dealing with the inter-domain scenario with asymmetric routing policies.

We enhance the basic DOM protocol with a Border Gateway Protocol (BGP) [6]-view based joining mechanism in this paper, where the inter-domain data forwarding path available in the border router of the multicast source domain will be sent to the destination domains first with a proper approach. The joining message can then be delivered upstream along the appropriate path to construct the reverse shortest path tree (SPT) even in the asymmetric inter-

X. Tian was with the Department of Electrical and Computer Engineering, Illinois Institute of Technology, Chicago, and is now with Electronic Engineering Department, Shanghai Jiao Tong University, Shanghai, China, 200240. E-mail: xtian@sjtu.edu.cn.
Y. Cheng are with the Department of Electrical and Computer Engineering, Illinois Institute of Technology, Chicago, Illinois, USA 60616. E-mail: cheng@iit.edu).

domain routing environment. The proposed scheme avoids the complexity of deploying/configuring Multiprotocol Extensions to BGP-4 (MBGP) [7], which is the traditional solution for multicasting with asymmetric inter-domain routing.

In the DOM design, each DOM-aware router maintains certain amount of local states that are independent of the number of groups to facilitate multicasting. Such a design has the significant advantage in reducing the information to be carried by the packet header and thus the false positives in the Bloom filter based implementation. In this paper, we further exploit the local states to achieve fast group joining. In a normal procedure, data forwarding could not be started until a group joining message is received by the source node. With our fast-join mechanism, data forwarding could immediately start when the joining message hit a node on the multicast tree of the requested group. Thus the average data access delay could be reduced.

An inherent issue associated with the Bloom filter based design is the impact of false positives. In this paper, we find that the recent solutions to delete the forwarding loop caused by false positives of Bloom filters in FRM [27] have limitations. LIPSIN [17] can impose heavy reencoding burden on the data source node but can not guarantee that the loop previously detected will be completely eliminated. BloomCast [18] could reduce the false positive forwarding at each node; however, it can only work in the symmetric routing environment, and only mitigate the probability of the forwarding loop rather than totally prevent it. We propose an accurate branch pruning scheme for DOM, which can block the falsely forwarded traffic at the root point in the network with both symmetric and asymmetric routing. We also show that the false positive forwarding loop in DOM can be completely and efficiently removed with the proposed pruning scheme.

Furthermore, we develop an incremental deployment solution for DOM protocol. The evolution to intelligent computer networks needs continuing efforts. It is impractical to upgrade all routers to be aware of DOM simultaneously. We propose a tunnel-based solution, where the DOM-aware routers could be seamlessly integrated with legacy routers to deploy the DOM protocol incrementally. An interesting property of our design is that the incremental deployment will not affect the scalability of DOM. We show in the performance evaluation that, even in the network with a small fraction of DOM-aware routers, the number of forwarding states maintained in each DOM router is still independent of the number of groups being supported by the router, and still remains the same as if all other routers were DOM routers.

Specifically, this paper has the four-fold contribution:

(1) We enhance the basic DOM protocol with a BGP-view based joining mechanism to address the asymmetric inter-domain routing.
(2) A fast group joining mechanism is developed to reduce the data access delay.
(3) We propose an accurate tree pruning scheme for DOM, which makes DOM capable of blocking falsely forwarded traffics and removing the forwarding loop.
(4) A tunnel-based implementation is developed for incremental deployment of DOM over the legacy networks.

The remainder of this paper is organized as follows. Section 2 gives a detailed overview of related work. Section 3 presents the DOM service model and practical design issues. Section 4 describes the BGP-view based joining scheme. Section 5 proposes an accurate pruning scheme to delete the forwarding loop in DOM. Section 6 proposes the incremental deployment solution for DOM. Performance of DOM is evaluated in Section 7. Section 8 gives the conclusion remarks and future work.

2 Related Work

In most of the modern multicast protocols, the group-specific forwarding states maintained at each router, as adopted by those legacy IP multicast protocols [1], [4], [5], [10], is traded with packet-carried information and processing/computation at each router for better scalability. REUNITE [14], [15] keeps multicast forwarding states only at *branching* nodes to improve the scalability; however, the branching node still maintains group-specific information with soft state, which incurs considerable memory and message overheads. The key idea of MAD [16] is to decouple the group membership from the forwarding information. For inactive groups with infrequent data traffic, MAD keeps the information in the overlay tree structure, instead of storing forwarding information in each node on the multicast tree. However, active groups with frequent data traffic are still served with traditional IP-style multicast.

Xcast [26] encodes the destinations list in the packet header and enhances the network router with the capability of replicating the packet and modifying the in-packet destinations list. Although taking the same service model as our DOM, Xcast is limited in scalability, as the maximum number of destinations that can be carried in one packet is restricted by the limited length of the Xcast header. Our DOM systematically studies a series of implementation issues to materialize the destination-oriented service model into a practical and scalable multicasting protocol, with utilizing Bloom filter to compress the destination information. A standard Bloom filter is a bit array representing a data set [28], where initially all bits in the array are set to 0. The data set is mapped to the bit vector with a number of independent hash functions, where each hash function maps an element of the set to the bit vector and set the corresponding position to 1. In this way, the data set is compressed into a bit vector for query [28].

Due to the space efficiency, Bloom filter has been utilized in many scalable multicast protocols. The most related work to our study is the FRM scheme [27]. With FRM, the border router of each destination domain will piggyback the active group information, encoded in a Bloom filter GRP_BF, on its regular BGP advertisements to reach the source domain. Based on the GRP_BF information, the border router of the source domain can detect the destination domains for an active group and compute a domain-level

TABLE 1
Features comparison between DOM and FRM

Features	DOM	FRM
Asymmetric routing	Handled with BGP view approach [22]	Source routing with encoded multicast tree branches [27]
Major states at in-netwrok router	Depends on the number of destination domains can be reached through the router [20]	Depends on the number of neighboring links of the router [27]
Bandwidth efficiency	Depends on the number of destination addresses encoded in the packet [20]	Depends on the number of tree branches encoded in the packet [27]
Joining operation	Can be accelerated with fast joining [22]	Has to be completed at source node [27]
Falsely forwarded traffic	Can be completely blocked	Can be constrained with probability [17], [18]
Forwarding loop	Can be completely removed	Can be constrained with probability [17], [18]
Incremental deployment	Can be implemented with tunneling technique	Not proposed yet

multicast tree for that group. The multicast tree will then be encoded using another Bloom filter and inserted into each multicast packet header. When receiving a multicast packet, the transit domain border router examines each of its neighboring domain-level edges against the attached Bloom filter to compute appropriate packet copies and output interfaces. We summarize and compare the major features of DOM and FRM in Table 1.

LIPSIN proposes to deploy a FRM-like multicast protocol in a publisher/subscriber network fabric [17], with enhanced techniques to delete false positive loop incurred by the Bloom filter. When receiving a packet, the LIPSIN router analyzes the in-packet Bloom filter to check if it contains a path that may lead the packet to return. If positive, the packet and its incoming interface will be cached. A loop is detected if the packet with the cached in-packet Bloom filter returns to the router from an interface other than the cached one. Nevertheless, the router caching the suspect packet is not necessarily the origin of the loop; therefore the false positive traffic can not be fully truncated. To deal with the challenge, the caching router has to signal a request upstream towards the data source to insert a different Bloom filter in the packet for the multicast tree, which imposes much burden on the data source node. Moreover, there is no way to guarantee that the re-encoded Bloom filter will never incur forwarding loop at a different router in the network.

BloomCast proposes a *bit permutation* technique to reduce the Bloom filter false positive effect in FRM-like protocols [18]. BloomCast let joining messages record each hop they traveled starting from leaves of the tree, encode the hop in a Bloom filter and re-map the Bloom filter to a different arrangement at each intermediate router. A unique reverse SPT is then created at the data source node by ORing all cumulatively permuted Bloom filters in joining messages. Then the reverse path forwarding is performed [2], where multicast data packets are forwarded along the paths that are reverse to the paths taken by joining messages [11]. During the forwarding, the falsely delivered packet can not be correctly de-mapped through the bit permutation at each hop, so the packet with no matched output interfaces will be dropped. Unfortunately, although BloomCast works smoothly under the symmetric routing assumption, the inter-domain routing is usually asymmetric for the administrative reasons [15]. Moreover, BloomCast still can not identify the origin of the forwarding loop once it occurs, and bit permutation can only mitigate the probability of the forwarding loop rather than totally prevent it.

3 DOM: Service Model and Practical Design Issues

3.1 Service Model

3.1.1 Membership Management

For membership management, a border router of a stub autonomous system (AS) domain is selected as the *designated router* (DR). For convenience, we use RDR (SDR) to denote the DR of a receiver-side (source-side) AS domain.

The RDR basically needs to implement the Internet Group Management Protocol (IGMP) [8] to discover the active groups within its domain. When new groups are activated, the RDR is triggered to send *membership updating messages* (MUMs) to the *data source node* (SRC) in the format as (RDR: GID_1, GID_2, $\cdots$, GID_n), where RDR represents a domain prefix and GID represents the group ID. Conveying the information that the sending RDR's domain is interested in which groups provisioned by the SRC, the MUM will be delivered along the shortest path between the RDR and the SRC, determined by the unicast routing table.

The SRC aggregates the MUM messages it received and maintains a *multicast group list* (MGL). For each group provisioned by the SRC, the MGL establishes a record in the format as (GID: RDR_1, RDR_2, $\cdots$, RDR_n), where each RDR again indicates a domain prefix. The MGL let the SRC know the interested receiver domains for each group it provisions. When the SRC multicasts data over a certain group, it will insert the corresponding MGL into the packet as the destination information in the format of a shim header, which is between the transport layer and the network layer header of the packet. The multicast packets are then forwarded to the SDR for inter-domain multicasting.

3.1.2 Multicast Forwarding Protocol

When receiving a multicast packet, the intermediate transit-domain border router (TBR) performs the following processing: first, check the unicast routing table to determine the output interface for each destination listed in the MGL of the packet, and aggregate destinations with the same output interface into a set; second, replicate the packet for each unique interface found in the first step; third, update the MGL of each packet copy with the aggregated set yielded in the first step, so that the packet copy for a given interface contains only the destinations that can be reached via this interface. For a given interface, destinations to be delivered along other interfaces are removed from the original MGL record. As TBRs perform forwarding based on the updated MGL record, the downstream TBRs will not generate unnecessary packet copies that have been served by other sibling subtrees. Each TBR will execute the same operations of aggregation, replication, and MGL record updating, until one multicast packet reaches a RDR.

3.2 Practical Design Issues

3.2.1 Limit the Explicit Addressing Overhead

In the prototype DOM service model, all the routers involved in the multicast forwarding (other than the DRs) do not need to maintain any state regarding multicasting. The forwarding complexity is totally independent of the number of groups to be supported, resulting in desirable scalability. Nevertheless, considerable bandwidth overhead could be incurred when there are a large number of receivers (RDRs) for each group: the MGL in the packet becomes impractically long, and the number of receivers can be supported is constrained by the packet header size.

3.2.2 Accommodate Longest-Prefix Matching and Route Aggregation

A possible solution of limiting the addressing overhead is to encode the MGL into a Bloom filter [27] which improves the space efficiency. However, the Bloom-filter based design needs to support the features of Internet. Normally, Internet routers apply the longest-prefix matching and route aggregation schemes to control the size of the unicast routing table, thus the same destination network may be represented with different network prefixes in different routers. Since the standard Bloom filter only supports exact query, it is possible that the destination RDR prefixes encoded in the Bloom filter can not match any forwarding entry stored in a SDR/TBR. Instead of directly utilizing the unicast routing table, there is a need to establish the forwarding states that can recognize the Bloom-filter-formatted MGL along the data delivery path.

3.2.3 Work with the Asymmetric Inter-Domain Routing

Most of the multicast protocols in practice [10], [15] establish the forwarding states when the joining request is delivered from the receiver to the source node (or Rendezvous Point), and then forward the data packets along the path that is reverse to the joining path, which is known as reverse path forwarding [13]. However, constructing this reverse SPT requires the symmetric routing environment: the path from the source to a receiver follows the same path used to go from the receiver to the source. Unfortunately, the inter-domain routing is usually asymmetric for the administrative reasons [15]. When designing the DOM, we also have to consider the effect of asymmetric routing on the protocol, so that the proposed protocol can be applied in the practical Internet.

3.2.4 Eliminate Loops Caused by False Positive

The Bloom filter incurs false positive, which means that an element not encoded in the Bloom filter can be falsely detected. In some subtle cases, the false positive can result in forwarding loops, which could cause the partial breakdown in the network. DOM should have the ability to eliminate the loops caused by the Bloom filter false positive, while the cost of the ability should be constrained.

3.2.5 Support Incremental Deployability

The evolution to next-generation Internet needs continuing efforts, thus it is impractical to upgrade all routers to be aware of DOM simultaneously. DOM needs to be incrementally deployable: it should be able to work with even only a small fraction of DOM-aware routers in the network, where the correctness should not be affected but may lose some efficiency.

4 BGP-VIEW ENHANCED DOM

In this section, we first briefly describe the Bloom filter based design in the basic DOM protocol [20], which facilitates the presentation of the enhanced BGP-view based design. While the first two design issues listed in Section 3.2 are well addressed by the basic DOM design [20], the third issue requires the enhanced design presented in this section.

4.1 Bloom Filter Based Design

We are to describe the Bloom-filter based design of DOM according to the upstream procedure (i.e., states establishment) and downstream procedure (i.e., data forwarding), as illustrated in Fig. 1, where Bloom filters are illustrated as shadowed areas.

The left side of Fig. 1 shows how forwarding states are established by joining MUM messages. To reduce the bandwidth overhead for membership updating, the list of active groups in the MUM message is encoded with a *group Bloom filter* (GRP_BF). When an MUM message reaches an upstream SDR/TBR router, the router will retrieve the RDR prefix, and store it as a local forwarding state at the output interface corresponding to the MUM incoming interface; the local states will later be used for reverse path forwarding. By continuously observing the MUMs, each related interface of the TBR/SDR will memorize all the destination domains that can be reached through it, and the reverse SPT from the SRC to subscribing RDRs is constructed. At an output interface, each RDR is stored as a separate Bloom filter, termed as *interface RDR Bloom filter* (IRDR_BF), which will be used to facilitate multicast forwarding.

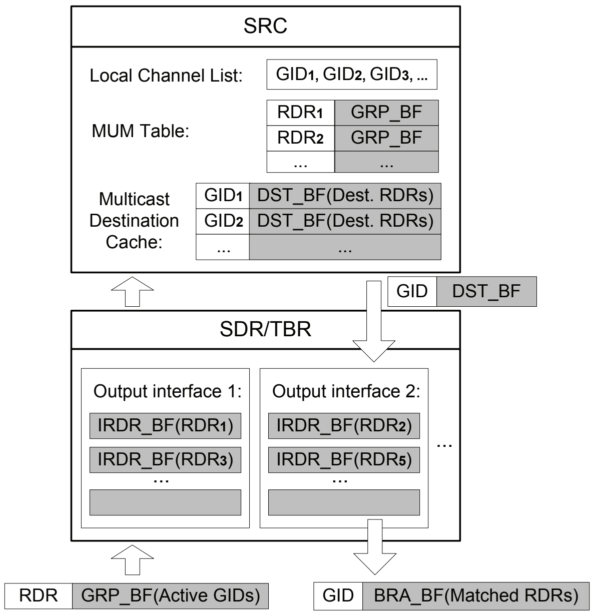


Fig. 1. Bloom-filter based design of DOM.

The upstream MUM messages will finally reach the SRC node, and each message will be stored as a record of the MUM table. The SRC node should have a *local channel list* indicating the multicast groups it provisions. By checking each GID against the MUM table and identifying the matched GRP_BF, the SRC can detect the destination prefixes for a given group. The destinations information under the group ID will be encoded into a *destination Bloom filter* (DST_BF) and stored into the multicast destination cache. Note that the DST_BF in fact encodes the MGL according to the DOM service model.

The right side of Fig. 1 illustrates how multicast packets are forwarded. At the SRC node, the DST_BF for a group will be inserted as the destination information into each multicast packet. In the downstream data forwarding process, each router generally executes the same operations of aggregation, replication, and MGL record updating as introduced in Section III-A. The only difference is that these operations are conducted with Bloom filters in both the packet and the router. Specifically, each TBR/SDR compares the packet's DST_BF with IRDR_BFs at each interface. A packet replica is generated and dispatched along the interface, if the DST_BF in the packet header and the IRDR_BFs installed at the interface have any element matched. The subset of matched prefixes associated with each output interface is then re-encoded into the *branch Bloom filter* (BRA_BF). The BRA_BF will be inserted into the packet replica delivered through that interface, serving as the destination information DST_BF for further downstream forwarding.

With DOM, the forwarding states stored at the router are destination-specific and totally independent of the number of groups passing through the router. For a subscriber domain, DOM stores only one state on each related intermediate router. In comparison, the subscriber domain may join in tens of thousands of groups and each group needs

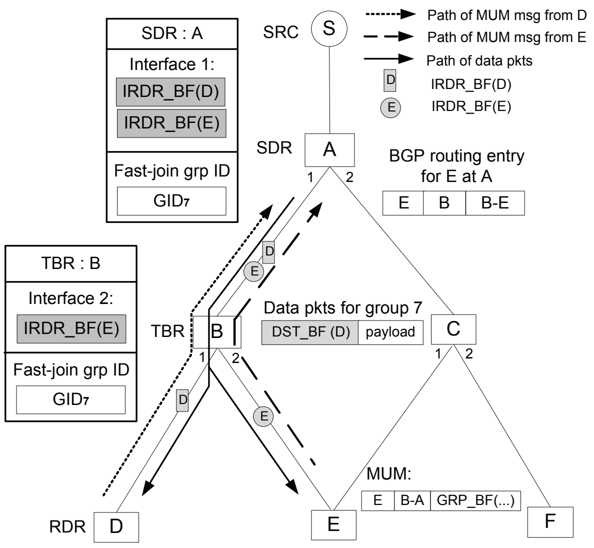


Fig. 2. Membership management and fast join.

a state on the related router under IP multicast. Thus the Bloom filter based design of DOM still achieves desirable scalability.

4.2 BGP-View Based Joining Process

The asymmetric routing issue [15] could be a challenge for the joining process described above. Consider the example in Fig. 2, it is possible that the MUM sent by *E* takes the path *E-C-A* to reach *S*, while the downstream data path is *A-B-E*, thus the forwarding states can not be installed following the RPF concept. To address such an issue, an option is to leverage the MBGP [10], which can announce different unicast- and multicast-capable routes to help the MUM messages take the correct joining path to the SRC but incurs high complexity [7], [10], [15]. We thus propose a low-complexity BGP-view based joining scheme to address the asymmetric routing issue. There are two prerequisites for our approach: 1) the physical links of the data delivery path from the SDR to a RDR must be bidirectional; and 2) the inter-domain routing policy must allow control messages (e.g., MUMs, etc.) going along the path that is reverse to the data delivery path. Considering the investment efficiency for link deployment, as well as the remarkable convenience to be gained in forwarding large quantities of data packets by the inter-domain routing policy, these two conditions are quite realistic.

The BGP-view based joining process could construct the reverse SPT even with asymmetric inter-domain routing, as illustrated in Fig. 2. The service provider designates a BGP-speaking SDR, which allows the SDR to compute the shortest paths from itself to any possible receivers. The information is stored in the local BGP routing table, where each table entry represents the local routing view for a given destination network prefix. For instance, the BGP routing entry for the network associated with E shows that E can be reached through the next-hop B and the path vector B-E in Fig. 2. The BGP routing entry is notified to the corresponding RDR so that the receiver side knows the actual routing view the sender-side can see. Then, the MUM is forwarded along the reverse path indicated by the BGP path vector with source routing, rather than the path indicated by the unicast routing table. In our example, the MUM from E takes the path E-B-A instead of E-C-A to join in S, with the designated route B-A carried in the message, as illustrated with dashed-line in Fig. 2.

A natural question is: how the BGP view seen by the SDR is notified to a RDR? The key observation is that DOM adopts the *source-based* service model [3], [10], where a receiver application must know the SRC information (i.e., SRC IP address, channel number, etc.) before subscribing to a channel. A number of techniques can be used to transport the BGP routing entry from the SDR to a RDR, including via web pages, sessions announcement applications, etc [3].

4.3 Fast Group Joining

4.3.1 Design

With the asymmetric routing issue addressed, we now consider how to further optimize the group joining process. The inefficiency we look at is that the joining process has to be completed at the SRC node, even if the joining message has passed an intermediate router already on the multicasting tree. In FRM-like protocols, the joining process also has to be completed at the data source domain. The DOM design allows easy extension to achieve a fast group joining mechanism. With the fast group joining, it is possible for the RDR to start receiving requested packets before its MUM message arrives at the SRC in DOM.

Specifically, we need the joining message to explicitly enumerate the group IDs qualified for fast joining, rather than coded in the GRP_BF as in the regular joining procedure. When an intermediate router receives such kind of fast joining message, it needs to store the indicated fast-joining group IDs on the receiving interface in addition to the regular IRDR. At the same time, the router further forwards the joining message upstream to the SRC node. If the intermediate router already forwards packets associated with the requested fast-joining group (to other destinations joined before), it could immediately deliver traffic to the requesting destination domain although the joining message has not reach the SRC node yet. Such traffic due to fast joining in downstream could be further forwarded by the auxiliary *GID based forwarding*. Later, after the joining message is processed by the SRC node and the destination information is incorporated into the DST_BF, the auxiliary forwarding process could be replaced by the normal forwarding process.

A sample fast group joining procedure is illustrated in Fig. 2. After sending initial MUMs towards a given SRC, a rudimentary multicasting tree is established between the SRC and subscriber RDRs. A subsequent MUM from E then follows the path marked by the dashed-line to subscribe to a newly activated group within its domain, say group 7, explicitly carrying the group ID and indicating fast joining. Then the TBR B retrieves from the fast-join MUM the requested group ID, i.e., GID_7, and places it at the interface 2. B continues to forward the MUM up to S with the same operation conducted at each router passed. If B is forwarding data packets of group 7 to D along its interface 1, B will discover that the group-7 data packets that are being forwarded via interface 1 match the GID_7 labeled at the interface 2. It will immediately forward the same data packets via the interface 2 to E. Thus E can receive the requested data packets before the MUM arrives at S.

A natural concern is whether the group specific information incurred by fast joining could affect the scalability of DOM. We would note that the fast-join GIDs are only stored in the router temporarily. When the regular forwarding process confirms that the packet with a fast-joining GID_i can now be normally dispatched through an interface, the GID based forwarding process is then stopped and the corresponding label GID_i on the interface can be deleted. In Fig. 2, consider the group-7 packet at SDR A, as DST_BF(D) matches the IRDR_BF(D), the regular forwarding process confirms that the group-7 packet should be dispatched through A's interface 1 anyway, the fast-join state GID_7 can be deleted immediately. In TBR B, however, the GID_7 can be deleted only when the first packet with the updated DST_BF(D,E) arrives later. At that time, the regular forwarding process will confirm that group-7 packets should be forwarded via interface 2, and GID_7 is then deleted.

4.3.2 Discussion

Explicit enumeration of group IDs in packet header for fast joining could incur bandwidth overhead, so the number of group IDs qualified for fast joining should be constrained. The group IDs of popular channels are good candidates, as the intermediate nodes forwarding packets for these channels are more likely to be hit. The popularity of each channel could be acknowledged by corresponding SRC in the channel information retrieval stage. The service provider could set appropriate threshold of the number of groups qualified for fast joining considering the network traffic condition, so that a trade-off between bandwidth and delay efficiency can be achieved. Another implementation detail is that we associate a timer with each GID label, and the GID is automatically deleted when the time goes up, which is for any unforeseen scenarios.

It is worthy note that the SRC node may require to control the receiving nodes to join the group, which at first glance may be violated by the fast-join scheme proposed. In fact, solutions to address security concerns for source specific multicast (SSM) could be adopted to resolve the issue [3], as they both utilize the source-based model.

5 False Positive and Forwarding Loops

An inherent issue associated with the Bloom filter based multicast protocols is that the Bloom filter incurs false positive [29]: it is possible that an element not encoded

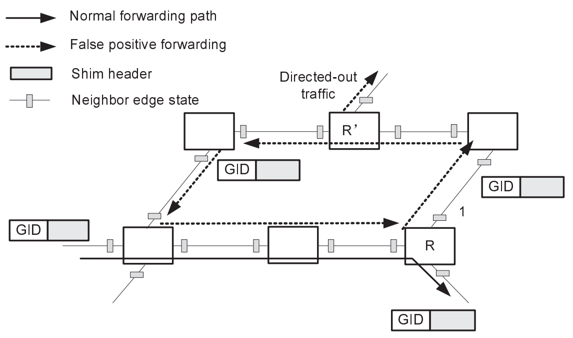
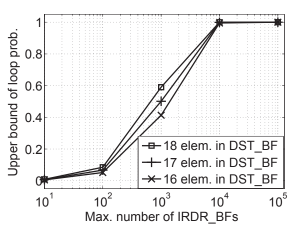
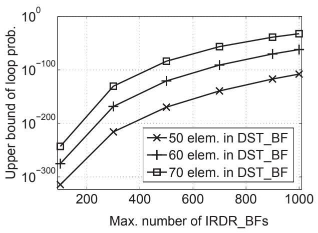


Fig. 3. Loop caused by false positive forwarding.



(a) Loop by general bit-matching



(b) Loop by conservation of bits

Fig. 4. Numerical analysis of the loop in DOM.

in the Bloom filter can be falsely detected. DOM and FRM both adopt the Bloom-filter based design; however, the forwarding loop issue has not been completely resolved for multicast protocols with the FRM flavor as we mentioned earlier. This section analyzes the loop issue in the context of DOM, after which we propose a scheme of blocking the falsely forwarded traffic incurred by false positives and show that the scheme could completely eliminate the forwarding loop due to false positive.

5.1 Analysis of the Forwarding Loop in DOM

DOM may incur the bit mismatching caused by the Bloom-filter false positive. When the falsely forwarded packet keeps mismatching with neighbor edge states installed along the interfaces that constitute the loop topology, as shown in Fig. 3 the loop will be formed.

The forwarding loop in DOM caused by false positives can be automatically eliminated in most cases except in the case of *conservation of bits* [23]. This is because the DOM downstream forwarding scheme normally can keep deleting the 1-bit positions in the in-packet Bloom filter DST_BF with DOM, and the 1-bit positions remained in the Bloom filter will not match any IRDR_BF in the intermediate router eventually, thus the falsely forwarded packet will finally be dropped, and the loop is eliminated. However, in the case of conservation of bits, the DST_BF in the packet happens to set all 1-bit positions the same way as in the pervious-hop DST_BF, the packet with the current DST_BF keeps mismatching along a loop topology and all 1-bit positions remain unchanged; therefore, the DOM downstream forwarding scheme will not be able to eliminate the false positive forwarding loop.

The probability that a forwarding loop is formed due to general bit-matching false positive can be upper bounded [23]. Figure 4 (a) shows the upper bound probability of loop occurrence in this case. We set the Bloom filter size to be 320 bits, and the number of elements in the DST_BF to be 16, 17 and 18, respectively. These number of elements could keep the false positive probability of single bit-matching operation in the order of 10^{-4}. However, the upper bound of loop occurrence probability is dramatically increasing with the number of the maximum number of IRDR_BFs at intermediate routers, as shown in Fig. 4 (a). When the value of X-axis is in the magnitude of 10^4, the loop will definitely occur.

The probability that a forwarding loop is formed with conservation of bits event can also be upper bounded [23]. Figure 4 (b) shows a numerical analysis of the upper bound probability of loop occurrence due to the conservation of bits event. We use the same sized Bloom filter, and the number of elements in the in-packet DST_BF to be 50, 60 and 70, respectively. This is to create an extreme situation for the convenience of demonstration. As shown in Fig. 4 (b), the upper bound of loop occurrence probability is significantly increasing in the magnitude with the number of the maximum number of IRDR_BFs at intermediate routers.

It is necessary to purposely design a scheme for DOM protocol to eliminate forwarding loops; because it is prone that loops occur as indicated by Fig. 4, especially when DOM scales up. Although the DOM downstream forwarding scheme can automatically eliminate these loops, the induced redundant traffic is unpredictable as the falsely forwarded packets at each hop may incur new loops. Moreover, the probability that conservation of bits event happens also increases significantly as DOM scales up, as shown in Fig. 4 (b), and the loop incurred by the conservation of bits can not be automatically eliminated. The key of eliminating loops of any kind is to block falsely forwarded packets, which will be described in the following.

5.2 Pruning False Tree Branches

In DOM, a RDR stored on an interface of a router (in the form of an IRDR_BF) implies that a forwarding path through the interface from the router to the destination domain represented by the RDR exists, according to the joining process and the RPF techniques adopted in DOM. Note that such a fact is true in both symmetric and asymmetric scenarios. Thus if a false positive forwarding happens in the network due to mismatching with a certain RDR_i on an interface, the falsely forwarded packet will finally reach the destination domain associated with RDR_i. The destination domain can then identify that the traffic was due to false forwarding if the group was not requested by it, and subsequently sends a pruning message upstream reverse to the forwarding path to prune the false-forwarding branches and stop the mis-delivered packets.

Implementing the branch pruning design is not trivial. How to ensure the direction of the upstream pruning messages along those false-positive branches? Consider the example shown in Fig. 5, where RDR D and E subscribe to

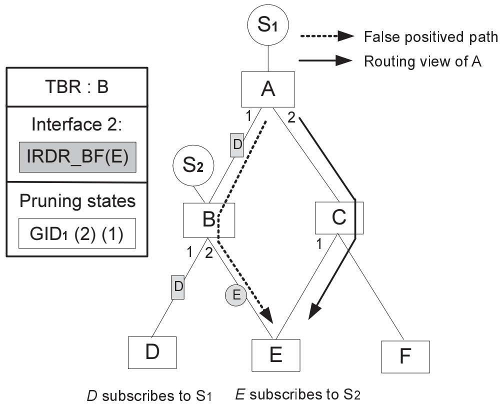


Fig. 5. Pruning false tree branches.

SRC S_1 and S_2 along path D-B-A and E-B, respectively. The packet generated by S_1 may be falsely forwarded to E if a false-positive match with IRDR_BF(E) on interface 2 of B happens. When E received the mis-delivered packet associated with S_1, it can not tell that the false forwarding was along the reverse path of the joining message to which SRC (note that E may have sent joining messages to many different SRCs). Even if E somehow correctly sets S_1 as the destination of the pruning messages, there are multiple paths from E to S_1. If E let pruning messages to S_1 take E-C rather than E-B as the reverse path, the pruning operation still can not block the mis-delivered traffic actually along B-E. With limited knowledge, the destination SRC of the upstream pruning message could not be properly set, and the upstream path could not be determined with the impact of asymmetric routing.

We design the pruning message propagation scheme as follows. In the first hop, the destination domain receiving falsely forwarded packets just sends a pruning message upstream through the interface where the false packets come. The pruning message carries the GID associated with falsely forwarded group. When the pruning message reaches an intermediate router, the false packets will be still coming to the router. By checking the incoming interface for packets associated with the tagged GID, the intermediate router can then easily identify the next upstream hop. Note that the destination domain receiving false packets will keep sending pruning message upstream, according to a certain schedule such as sending one pruning message after receiving n (≥ 1) false packets, until the false packets stop coming.

To facilitate the pruning process, each router involved also manages pruning states. When the pruning message arrives at an intermediate router for the first time, the router creates a pruning state in the format of $\mathrm{GID}_i(\mathcal{F})(\mathcal{N})$, where GID_i is the group identifier of the mis-delivered packets, $\mathcal{F}$ is the set of output interfaces that have falsely forwarded packets with GID_i, and $\mathcal{N}$ is the set of output interfaces that are normally forwarding packets with GID_i. If there are subsequent pruning messages associated with GID_i coming from another interface, the corresponding interface will be moved from set $\mathcal{N}$ to the set $\mathcal{F}$. In the example shown in Fig. 5, $\mathcal{N} = 1$ and $\mathcal{F} = 2$, which means that the interface 1 of B is normally forwarding packets of GID_1; because otherwise there should be another pruning message from interface 1. In this way, B could block the traffic with GID_1 towards E and keep forwarding that towards D.

5.3 Operations on Pruning States

The pruning states can help a TBR to properly determine whether to continue forwarding the pruning messages upstream, stop the forwarding, or remove the obsolete pruning states. Specifically, a TBR will take the following options upon receiving a pruning message associated with GID_i:

- Continue forwarding upstream the pruning message, if $\mathcal{N} = \phi$ and $\mathcal{F} \neq \phi$. This condition indicates that the TBR is an intermediate router along the false-forwarding branch, so it needs to continue forwarding the pruning message upstream towards the root node that generates the false-forwarding traffic.
- Stop forwarding upstream the pruning message, if $\mathcal{N} \neq \phi$ and $\mathcal{F} \neq \phi$ or the TBR is an SDR of a source domain. The first part of the condition, i.e., $\mathcal{N} \neq \phi$ and $\mathcal{F} \neq \phi$, indicates the current TBR is the root node that generates the false positive match, so there is no need to further forward the pruning message upstream. The branches in set $\mathcal{N}$ indicates the normal paths that correctly forward traffic for group GID_i, while branches in $\mathcal{F}$ indicates paths due to false positive match. Such a situation could be possible only if the current router receives correct traffic but false positive matching happens in the forwarding stage, that is, the current TBR is the root for false positive traffic. The second part of the condition is due to the possibility that the pruning message may go up to the SDR in the source domain if the false positive happened due to the mismatch when checking GRP_BF against the local channel list.

It is not difficult to see that the pruning state associated with a GID on an interface should be removed, if the destination domain ending the false-forwarding path now actively requests traffic from this group. Thus, when a destination domain needs to join a new group, it first check whether the group was involved in the false positive situation before. We let the RDR keep a record of the false positive GIDs it has observed. If the new active GID is found in the record, it needs to use a joining message carrying explicit GID and a pruning-removing flag to remove the pruning states on related interfaces. The destination address of such a joining message is set as the SRC address associated with the GID and then follows the DOM joining procedure (note that BGP-view based joining is applied in the asymmetric case [21]). When a TBR/SDR receives a joining message with a pruning-removing flag, if it has a pruning state associated with the indicated GID, it then just remove it. Note that such a pruning removing procedure is efficient, which just remove the pruning states on related hops that might impact the normal forwarding.

Note that the fast group joining procedure also requires joining message to explicitly carry GID. Thus, we could have a uniformed design using flag bits to indicate whether a joining message is a normal one carrying GRP_BF, or a fast group joining message or a pruning message or both. Considering the number of fast joining groups are limited and the false positive probability is small, the joining messages and computing overheads in intermediate TBRs associated with these groups will not impact the scalability much.

5.4 Loop Elimination

Theorem 1 The false tree branch pruning scheme can stop the redundant traffic and eliminate forwarding loops incurred by the false positive in Bloom filter matching under the condition that the domains associated with the SDR and RDRs are stub domains of the multicast group.

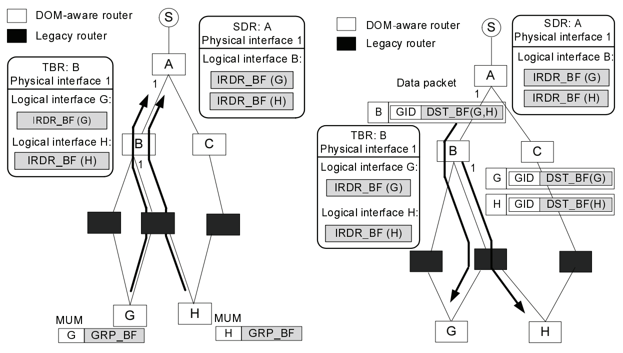
Proof: Because of the condition, the loop can only happen among TBRs [20]. The forwarding loop in DOM is formed if some falsely forwarded packet keeps mismatching its updated DST_BF with IRDR_BFs along the interfaces that constitute the loop topology. Note that any IRDR_BF on a certain interface was placed by a joining message from a destination domain; reverse to the path of the joining message is a data forwarding path to the destination domain according to the DOM design for both symmetric and asymmetric cases. Therefore, for given IRDR_BF that is falsely incorporated into the updated DST_BF, there must exist a TBR in the loop, which will direct at least one of the packet copies out of the loop which has a branch leading to the destination domain that generated the join message associated with the IRDR_BF under consideration. So the redundant traffic due to false positive can definitely be detected by that destination domain and incurs subsequent pruning messages. According to the pruning mechanism design, the pruning message will finally reach the root node that originated the false positive traffic and stop the traffic. Thus all the false positive traffic downstream and the forwarding loop if any will be totally eliminated. ■

Consider the example shown in Fig. 3, the false positive matching is initiated at the router R, and a forwarding loop is further formed as shown in the dashed line. In this example, the router R' has a branch leading the false positive traffic to a destination domain. The destination domain will then identify the situation of false positive and generates the pruning messages. The pruning messages will reach the root node R of the false positive traffic and eliminate the loop.

6 Incremental Deployability

6.1 Design

In the above, we assume all routers in the network aware of DOM for the convenience of presentation. This introduces a deployment problem, as for deployment in a real situation, it is impractical to update the long existing legacy



(a) Install IRDR_BFs at logical interfaces (b) Forwarding via logical interfaces

Fig. 6. Incremental deployment solution.

infrastructures simultaneously. Thus, gradually introducing DOM-aware routers into the network for incremental deployment is more practical. In the scenario where DOM-aware routers and legacy routers coexist, we propose to use IP tunneling technique to establish a DOM-aware tunnel system across the entire network regions. The basic idea is to consider the unicast path connecting two DOM-aware routers as a *logical interface*. When receiving a DOM packet, a DOM-aware router can conduct the same aggregation, replication, and MGL record updating operations but based on the states installed for the logical interfaces. The generated packets copies are then dispatched along the logical interface that leads to the next-hop DOM-ware router. Routers that do not implement DOM protocol will forward the packet as if they are performing regular unicast.

Fig. 6 illustrates how to incrementally deploy DOM. The black rectangles in the figure represent legacy unicast routers that do not implement DOM. When RDR G and H want to subscribe to the SRC S, they send out the MUMs towards S. A little modification required is to define a flag bit to indicate a *logical interface* and let the MUM record the local prefix as the logic interface address each time it is delivered by the DOM-aware router. For example, in Fig. 6(a), when the MUMs are sent from G and H, as they are DOM-aware routers, the prefixes of G and H are recorded as the logical interfaces in the two MUMs, respectively. After the MUMs arrive at B, B discovers that although the two MUMs come from the same physical interface 1, they are actually from two different logical interfaces. B creates two states, i.e., IRDR_BF(G) and IRDR_BF(H), for these two logical interfaces, respectively, with each encapsulated with the corresponding logic interface address. Then, B continues to forward the MUMs up to A. The logical interface address of both MUMs is set to B at this time, since B is also aware of DOM. When the MUMs reach A, A finds out that the two MUMs are from the same logical interface B, and corresponding states are generated as shown in Fig. 6(a). In this way, each intermediate DOM-aware router knows the next-hop DOM-aware router, and the DOM-aware tunnel system is established, which will facilitate the multicast data forwarding.

Fig. 6(b) depicts the forwarding procedure of the incremental deployment solution. When A receives the multicast packet with destinations G and H, it checks the forwarding states installed for each of its logical interfaces as regular DOM forwarding process. A notices that G and H could be reached through the same logical interface B, so only one packet is forwarded through the logical interface B. This packet is firstly encapsulated into an ordinary unicast packet with the logical interface B as the destination (recall that B is actually the network prefix of the transit domain border router), and then unicast to B as shown in Fig. 6(b). After this packet arrives at B, as B is DOM aware, it decapsulates the multicast packet and match the DST_BF against the IRDR_BF over each logic interface. Since there are matched IRDR over both logic interfaces G and H, two packet replicas are to be generated. Each packet replica updates the DST_BF according to DOM procedure and is encapsulated with the logic interface address to be delivered to the next-hop DOM-aware routers. These packets will pass a legacy router in the following forwarding process. The legacy router is unable to recognize the DOM-type packet and does not decapsulate the received packets; it just forwards the packets based on the packets' destination addresses.

6.2 Discussion

An interesting property of the incremental deployment solution is that the number of forwarding states installed at a given DOM-aware router only depends on the number of RDRs which could be reached through this router, and independent of the fraction of DOM-aware routers within the network. As shown in Fig. 6(b), there will be two forwarding states for RDR G and H, no matter the intermediate router towards B is aware of DOM or not.

One important characteristic of current Internet is that IPv4 and IPv6 coexist; however, enabling DOM in such an environment is not a challenge. Nowadays, the widely accepted strategy for transition from IPv4 to IPv6 is to encapsulate IPv6 datagram in IPv4, which is known as tunneling mechanism [9]. We could use the "nested tunnel" to let DOM work in the IPv4-IPv6 mixed environment, which means that the DOM "logical interface" based tunnel is inside the IP tunnel. Take the scenario in Fig. 6(b) for example, the DOM packet is first processed by the DOM tunneling process, where the packet copies for logical interface based forwarding are generated. After that, the yielded packet copies are delivered through the IP tunnel.

With the logical interface based approach, the fast group joining and branch pruning schemes described previously are also applicable, and the only variation is that the GID labels are operated according to the logical interfaces. In the incremental deployment solution, DOM-aware routers generate packet replicas only when it is necessary. The legacy routers do not affect the network regions where DOM-aware routers are concentrated, but they could lead to more redundant traffic (e.g., the multi-unicast at B) without affecting the correctness of DOM protocol.

We have analyzed and explained overheads of the DOM in a point-by-point manner so far. Table 2 summarizes the

TABLE 2
Overheads of DOM

Overheads	Location	When used	Scaling
DST_BFs	SRC_s	per packet	$O(g_{src} \cdot C(g_{src}))$
IRDR_BFs	SDR,TBR	per packet	$O(C_{rdr})$
IDs for Fast-join	SDR,TBR	per group	$O(g_{jid})$
timers for Fast-join	SDR,TBR	per group	$O(g_{jid})$
GID$_i(\mathcal{F})(\mathcal{N})$	SDR,TBR	per group	$O(g_{fid})$

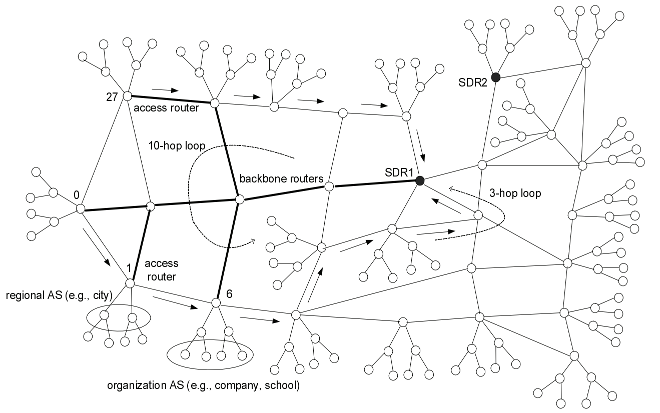


Fig. 7. Simulation topology.

overheads of the DOM and the corresponding scaling to facilitate better understanding, where

- g_{src} is the number of active groups the SRC at domain s (SRC_s) multicasts; $C(g_{src})$ is the average number of receiver domains for groups at SRC_s.
- C_{rdr} is the number of receiver domains for SRC_s, g_{jid} the number of group IDs for fast joining, and g_{fid} the number of group IDs with false forwarding.

A detailed examination of the DOM performance is illustrated in the following section.

7 Performance Evaluation

We use NS2 [31] simulation results to demonstrate the performance of DOM. The network topology for simulation is given in Fig. 7, which is widely used in the literature as a hypothetical US backbone network [25]. In our model, the source and transit domains are represented as backbone routers; regional and organization autonomous systems (ASes) are represented as designated border routers in those domains. The backbone router providing connection service to regional ASes is termed as *access router*, and regional ASes are connected by organization ASes, as illustrated in Fig. 7. The routing of the network is purposely configured asymmetric. For example, the arrows and the highlighted tree shows the upstream and downstream paths between nodes 0, 1, 6, 27 and $SRC1$. When evaluating the loop elimination performance, we will set up two loops as shown in Fig. 7. For each metric to be evaluated, we will try to show the average value of 100 simulation runs and plot the confidence intervals at 95%.

7.1 Memory Overhead under False Positives

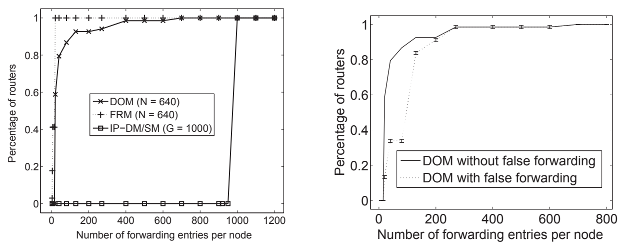
We first evaluate the scalability of DOM with comparison to other multicast schemes. SRCs are attached to SDRs denoted using black nodes with each SRC providing 500 groups. We connect each regional AS node with 16 RDRs, and let all 16 RDRs subscribe to all groups provisioned by both SRC1 and SRC2. Then we study the impact of false positive forwarding on memory overhead of DOM. In the simulation, the false positive forwarding scenario is configured as following: We let all 16 RDRs subscribe to SRC2 and only 2 of them to SRC1. RDRs will change their subscriptions for each run. This setting could establish many IRDR_BFs along links in the backbone network, which increases the probability that the SRC1 data packets are falsely forwarded to SRC2 subscribers. To make the scenario more extreme, we configure the false positive rate for Bloom filter matching to around 20%. The memory overhead is measured by counting *the number of forwarding entries at backbone routers and regional AS border routers that are involved in the multicast*.

The cumulative distribution function (CDF) of the number of forwarding entries per node for the multicast schemes under study are illustrated in Fig. 8 (a). It shows that DOM can significantly reduce the number of forwarding states stored at each node compared with IP multicast. This is because DOM states are destination-specific, instead of group-specific. Consequently, the number of forwarding states per node for DOM is independent of the number of groups being supported by the node. DOM requires comparatively more forwarding states than FRM does, as the number of forwarding entries per node is the AS degree of the node for FRM [27]. However, the states maintained can greatly benefit the bandwidth efficiency, which is to be discussed in Section 7.2.

The impact of false-positive forwarding on DOM is illustrated in Fig. 8 (b). For the DOM with false-positive forwarding, the states $\mathrm{GID}_i(\mathcal{F})(\mathcal{N})$ for pruning false tree branches should also be counted. Fig. 8 (b) shows that the pruning states has limited influence on the scalability of DOM, as the pruning states will be finally installed on the root node along the false forwarding path. In this experiment, the pruning states are primarily installed at each regional AS node due to the subscription configuration. We create this setting because this is an adverse case in the distribution of the pruning states. Imagine that if the pruning states are moving towards the backbone network, some of them will merge at some upstream node according to the pruning states operation; therefore the performance for false tree branch pruning will be better in other scenarios. Even if the pruning states are related to the number of falsely supported groups, the number of resulted states are still much less than that in IP multicast. The scalability of DOM remains with the false tree branch pruning operations.

7.2 Bandwidth Overhead under False Positives

In this experiment, we still maintain the false positive forwarding configuration, and focus on the multicast tree rooted at SRC1. A clip of MPEG-4 video stream is delivered from SRC1 to a number of subscribing RDRs ranging



(a) N = 16 (b) N = 80

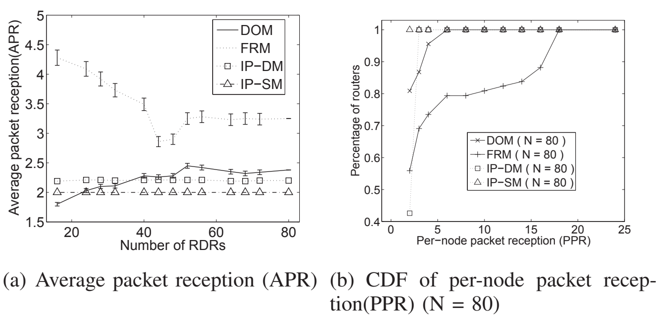
Fig. 8. CDF of the number of forwarding entries per node.

from 16 to 80. We will study the performance of DOM with the impact of false-positive forwarding, compared to other schemes without the impact. The simulation results show that the false positive forwarding does not affect the advantage of DOM in bandwidth utilization over FRM, with the proposed false tree pruning operations. The bandwidth consumption of DOM are primarily caused by the packet-carried DST_BF and the redundantly-transmitted traffic. The specific results and explanations are shown below.

Average Packet Reception (APR) – The average packet reception (APR) is defined as *the average number of packets received by each non-RDR router in the multicast tree in 10 seconds*. Fig. 9 (a) shows the APR versus N (the number of RDRs involved in multicasting) with different multicast schemes compared. We observe that DOM achieves the performance close to IP multicast, and even outperforms IP multicast when N is very small, as the IP multicast has the tree maintenance overhead. DOM has a better performance than FRM, especially when the multicast subtree along one interface of the SDR has many branches. This is because, for FRM, more packets have to be generated to carry subtree branches to cover all destination RDRs.

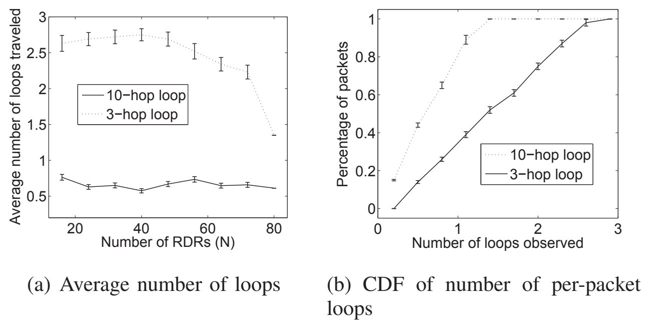
Compared with IP-DM, DOM does not use the "broadcast-and-prune" [10] method to maintain the tree structure. The performance of IP-SM is closely related to the selection of rendezvous point (RP). The data packets are first unicast to the RP and then disseminated to RDRs from the RP, while DOM can deliver packets to RDR directly. This is why DOM can perform even better than IP multicast when $N = 16$.

The reason for the better performance of DOM over FRM is that DOM needs to encode less elements in the Bloom filter than FRM does, as usually it takes more branches to cover the same number of destinations. In addition, we observe that the APR of FRM is decreasing when N is between 16 and 50. This is because the RDRs in these cases are concentrated on subtrees sourced from different neighbor edges of the SDRs. The FRM in-packet Bloom filter happened to be able to contain all branches of each subtree and the number of nodes receiving exact 2 packets increases. However, when N keeps increasing, the number of branches in the subtrees go beyond the capacity of the Bloom filter again, and the number of redundant packets increases. Thus the FRM curve presents the shape of a funnel.



(a) Average packet reception (APR) (b) CDF of per-node packet reception(PPR) (N = 80)

Fig. 9. Bandwidth overhead.



(a) Average number of loops (b) CDF of number of per-packet loops

Fig. 10. Loop elimination.

Per-node Packet Reception (PPR) Distribution – The per-node packet reception is *the number of packets received at a given node when multicasting from the SRC to all receivers for 10 seconds*. Fig. 9 (b) plot the CDF of the PPR for different multicast mechanisms when $N = 80$. In DOM, about 80% of the nodes receive less than 400 packets, and there is no node receiving more than 1200 packets in the case of. The redundant traffic in DOM is incurred by splitting the destination set into smaller subsets to fit into the DST_BF. The falsely forwarded traffic in DOM is effectively blocked. In FRM, only about 55% of the nodes receive less than 400 packets. The largest number of packets a node can receive is up to 3500. In IP-DM and SM, almost every node receives exact less than 400 packets, except for a few that receive some redundant packets in the tree maintenance or the source-to-RP unicast. The number of nodes receiving no redundant packets under DOM is close to that under IP multicast and is larger than that under FRM.

7.3 Loop Elimination

To evaluate the performance of loop elimination, we purposely configure all receiving RDRs' joining paths to make it possible that the IRDR_BFs in the intermediate nodes can form a 10-hop and 3-hop loops as shown in Fig. 7. Source nodes SRC1 and SRC2 are set to send packets of group 1, and we increase the number of RDRs that regard packets of group 1 as falsely forwarded packets in each simulation round, and the following two metrics are used.

Average Number of Loops (ANL) – ANL is defined as *the average number of loops the falsely forwarded packets traveled along the loop topology*. Fig. 10 (a) shows the values of ANL with respect to the number of RDRs that may receive falsely forwarded packets. It shows that ANLs in the 10-hop loop are higher than in the 3-hop counterpart. This is because it takes longer for packets to travel 1 loop in the 10-hop loop, and during this time, those packets could be blocked by the proposed loop elimination scheme. The number of loop could be fractional, as the number of hops the packet traveled could be less than the length of the loop. The ANL varies as the number of RDRs changes, since the the loop elimination performance is also related to where the root of the false tree branches is. However, we could observe that the ANL becomes lower as the N increases in general, because it is more likely some RDR could be very near to the root of the false tree branches and the loop is eliminate more quickly.

Number of Loops by Per-Packet Distribution (NLPD) – NLPD is the *CDF of the number of loops traveled by each packet*. The results for $N = 80$ is shown in Fig. 10 (b). For the 10-hop loop, all packets traveled less than 15 hops (1.5 loop) along the loop, but for the 3-hop loop, no packet has traveled along the topology for 3 rounds. The most long-lived packet experienced 7 hop. The proposed loop elimination is efficient as it can quickly find the root of the false tree branches and cut the redundant traffic along the loop topology.

7.4 Joining Delay

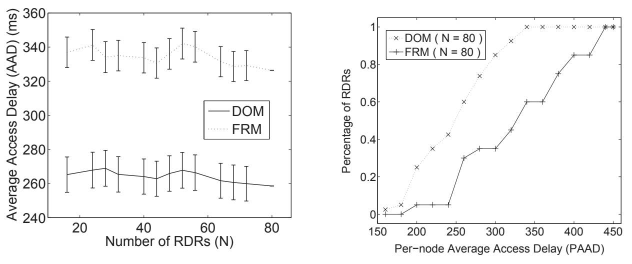
Average Access Delay (AAD) – The average access delay (AAD) is defined as

$$\mathrm{AAD} = \frac{\sum_{i=1}^{N}\sum_{j=1}^{M} d_{ij}}{M \cdot N}, \tag{1}$$

where d_{ij} represents $\mathrm{RDR_i}$'s access delay, i.e., the delay between the MUM is sent out and the first requested data packet arrives, for group j. M is the number of groups $\mathrm{RDR_i}$ subscribes to, while N is the total number of RDRs in the experiment.

Fig. 11 (a) illustrates the resulted AADs for different quantities of RDRs. The use of the fast group joining scheme in DOM can shorten the AADs by approximately 22% on average, compared with the FRM. Since it is possible that the requested data are forwarded to the subscriber before the joining request reach the SRC with DOM fast-join scheme. We can see AADs of the DOM and FRM vary with the number of participating RDRs (N) non-monotonically. This is because the access delay performance is also closely related to the distance between RDRs and the SRCs. With more participating RDRs near to (far from) SRCs, the AAD is reasonably shorter (longer) as the delay caused by rudimentary tree construction is shorter (longer).

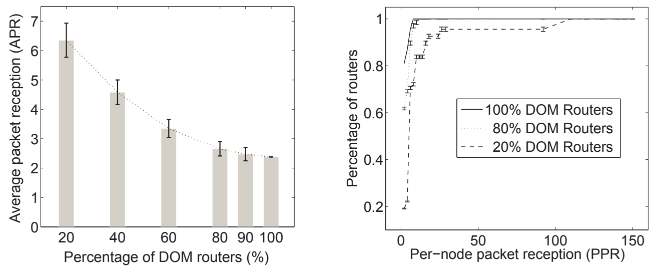
Per-node Average Access Delay (PAAD) Distribution – We here demonstrate the CDF for *the per-node access delay averaging the participated groups*. The results for $N = 80$ are depicted in Fig. 11 (b). The curve representing the DOM is above the corresponding one representing



(a) Average access delay.

(b) CDF of per-node average access delay(PAAD,N = 80)

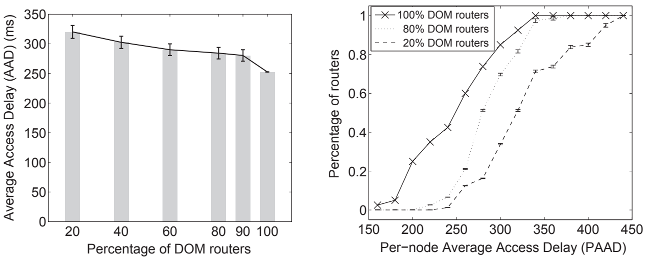
Fig. 11. Joining delay.



(a) APR(N = 80)

(b) PPR Distribution(N = 80)

Fig. 12. Incremental deployment: effect on bandwidth overhead.



(a) AAD(N = 80)

(b) PAAD Distribution(N = 80)

Fig. 13. Incremental deployment: effect on joining delay.

FRM, which indicates that the number of RDRs experiencing the shorter PAAD in DOM is more than that in FRM. These results corroborate the experiment outcome demonstrated in Fig. 11 (a). Half of the participating RDRs have the PAAD that is less than $250ms$ and the upper bound of the PAAD is $340ms$, as indicated in Fig. 11(b). With FRM, RDRs having the PAAD no greater than $260ms$ account for only 25% of the participating RDRs and the PAAD upper bound increases up to $440ms$.

7.5 Incremental Deployability

We here illustrate how DOM performs when only a fraction of routers are aware of DOM protocol in the entire network. We still use the topology in Fig. 7 and vary the percentage of DOM-aware routers from 20% to 100%. There are 2 SRCs and 80 RDRs in the simulation, where RDRs are aware of DOM and the intermediate routers are randomly assigned as DOM-aware routers for each percentage.

Effect on Memory Overhead – The incremental deployment solution does not affect the memory overhead at any given DOM-aware router. This is because the number of forwarding entries at each DOM-aware router only depends on the number of RDRs which could be reached through this router. The variation of the percentage of DOM-aware routers does not affect the number of forwarding entries at any given DOM-aware node as long as the number of RDRs and their locations are fixed.

Effect on Bandwidth Overhead – The average packet reception (APR) and per-node packet reception (PPR) distribution at different percentages are shown in Fig. 12, where the packet receptions of all routers are counted in. In Fig. 12(a), the more routers are aware of DOM, the lower the APR is. This is simply because more DOM-aware routers could reduce the chances for initiating multi-unicast over the network. In Fig. 12(b), it can be seen that the PPR distribution in the 80% case is very close to that in the 100% case. In the 20% case, some nodes receive 30 packets when multicasting a single packet from each SRC to all RDRs, but the correctness of the protocol is not affected.

Effect on Joining Delay – The incremental deployment solution will not affect the correctness of the fast group joining scheme. Fig. 13(a) illustrates the average access delay (AAD) with different fractions of DOM-aware routers. Even in the 20% case, it is still possible that the joining request are processed before it reaches the SRC; therefore, the AAD of DOM is no greater than that of FRM. The more routers become aware of DOM, the more likely the joining request is processed at the intermediate routers before arriving at the SRC; thus the AAD decreases as the percentage of DOM-aware router increases. For the same reason, the number of RDRs perceived generally lower joining delay when the percentage of the DOM-aware router is higher, as depicted in Fig. 13(b). Therefore, the fast group joining scheme works correctly even if most of the intermediate routers can not recognize DOM protocol.

8 Conclusions and Future Work

In this paper, we have developed a scalable destination-oriented multicast (DOM) protocol for computer networks where the routers are expected to have enhanced intelligence to process packets. All the management and addressing information traversing the network is encoded with Bloom filters for memory and bandwidth efficiency. We have incorporated the reverse path forwarding (RPF) concept and the BGP routing information, so that DOM can work efficiently in practical networking scenarios especially with asymmetric inter-domain routing. A fast group joining scheme has also been proposed to further optimize the data access delay performance of DOM. Moreover, an accurate tree branch pruning scheme has been proposed to block the falsely forwarded traffic and completely eliminate the forwarding loop. A tunnel based solution has been developed to incrementally deploy DOM over the legacy infrastructure. We will study the congestion control [32] and security [33] issue for Bloom filter based multicast protocols in the future.

9 Acknowledgements

Xiaohua Tian's work in this paper is partially supported by NSFC 61202373, China Postdoctoral Science Foundation Grant 2011M500774 and 2012T50417, and STCSM Grant 12JC1405200.

References

[1] S.Deering and D.Cheriton, "The PIM architecture for wide area multicasting," *ACM Trans. Computer Systems*, vol. 8, no. 2, pp. 85–110, May. 1990.

[2] Y. K. Dalal and R. M. Metcalfe, "Reverse path forwarding of broadcast packets," *Communications of the ACM*, vol. 21, no. 12, pp. 1040–1048, Dec. 1978.

[3] H. Holbrook, "Source-specific multicast for IP," IETF RFC 4607, Aug. 2006.

[4] A. Adams, J. Nicholas and W. Siadak *et. al.*, "Protocol independent multicast-dense mode (PIM-DM): protocol specification (Revised)," IETF RFC 3973, Jun. 1998.

[5] D. Estrin *et. al.*, "Protocol independent multicast-sparse mode (PIM-SM): protocol specification," IETF RFC 2362, Jun. 1998.

[6] Y. Rekhter *et. al.*, "A Border Gateway Protocol 4 (BGP-4)," IETF RFC 1771, Mar. 1995.

[7] T. Bates *et. al.*, "Multiprotocol Extensions for BGP-4 ," IETF RFC 2858, Jun. 2000.

[8] B. Cain, S. Deering, I. Kouvelas, B. Fenner, A. Thyagarajan, "Internet group management protocol, version 3," IETF RFC 3376, Oct. 2002.

[9] E. Nordmark, "Basic transition mechanisms for IPv6 hosts and routers," IETF RFC 4213, Oct. 2005.

[10] K.C. Almeroth , "The evolution of multicast: from the MBone to interdomain multicast to Internet2 deployment," *IEEE Network*, vol. 14, no. 1, pp.10–20, Jan.-Feb. 2000.

[11] D. R. Kumar, W. A. Najjar and P. K. Srimani, "A new adaptive hardware tree-based multicast routing in K-ary N-cubes," *IEEE Trans. Computers*, vol. 50, no. 7, pp. 647–659, July 2001.

[12] S. Fahmy and M. Kwon , "Characterizing overlay multicast networks and their costs," *IEEE/ACM Trans. Networking*, vol. 15, no. 2, pp.373–386, April 2007.

[13] C. Francalanci and P. Giacomazzi, "A high-performance deadlock-free multicast routing algorithm for K-ary N-cubes," *IEEE Trans. Computers*, vol. 59, no. 2, pp.174–187, Feb. 2010.

[14] I. Stoica, T.S.E. Ng and H.Zhang, "REUNITE: a recursive unicast approach to multicast," in *Proc. IEEE INFOCOM*, Mar. 2000, pp.1644-1653.

[15] L. Costa, S. Fdida and O. Duarte , "Incremental service deployment using the hop-by-hop multicast routing protocol," *IEEE/ACM Trans. Networking*, vol. 14, no. 3, pp.543–556, Jun. 2007.

[16] T. W. Cho, M. Rabinovich, K.K. Ramakrishnan, D. Srivastava, Y. Zhang, "Enabling content dissemination using efficient and scalable multicast," in *Proc. IEEE INFOCOM*, Mar. 2009, pp.1980-1988.

[17] P. Jokela, A. Zahemszky, C. E. Rothenberg, S. Arianfar, and P. Nikander, "LIPSIN: Line speed publish/subscribe inter-networking," in *Proc. ACM SIGCOMM*, Aug. 2009, pp. 195–205.

[18] M. Särelä, C. E. Rothenberg, T. Aura, A. Zahemszky, P. Nikander and J. Ott, "Forwarding anomalies in Bloom filter-based multicast," in *Proc. IEEE INFOCOM*, Apr. 2011.

[19] X. Tian, Y. Cheng, K. Ren, and B. Liu, "Multicast with an application-oriented networking (AON) approach," in *Proc. IEEE ICC*, May 2008, pp.5646–5651.

[20] X. Tian, Y. Cheng, and B. Liu, "Design of a scalable multicast scheme with an application-network cross-layer approach," *IEEE Trans. Multimedia*, vol. 11, no. 6, pp.1160–1169, Oct. 2009.

[21] X. Tian, Y. Cheng, and X. Shen, "DOM: A scalable multicast protocol for next-generation Internet," *IEEE Network*, vol. 24, no.4, pp.45–51, July 2010.

[22] X. Tian, Y. Cheng, and B. Liu, "A fast-join mechanism for Inter-domain multicasting," in *Proc. IEEE GLOBECOM*, 2010, pp. 1–5.

[23] X. Tian and Y. Cheng, "Loop mitigation in Bloom filter based multicast: a destination-oriented approach," in *Proc. IEEE INFOCOM*, 2012, pp. 2131–2139.

[24] Y. Chu, S. Rao, S. Seshan, and H. Zhang, "Enabling conferencing applications on the Internet using an overlay multicast architecture," in *Proc. ACM SIGCOMM*, Aug. 2001, pp. 55–67.

[25] R. Doverspike, G. Li, K. Oikonomou, K.K. Ramakrishnan and D. Wang, "IP backbone design for multimedia distribution: architecture and performance," in *Proc. IEEE INFOCOM*, May 2007, pp. 1523–1531.

[26] R. Boivie *et. al.*, "Explicit multicast (Xcast) basic specification," Internet draft, Mar. 2001.

[27] S. Ratnasamy, A. Ermolinskiy and S. Shenker " Revisiting IP multicast, " in *Proc. ACM SIGCOMM*, Aug. 2006, pp. 15–26.

[28] Y. Hua, B. Xiao, B. Veeravalli and D. Feng, "Locality-Sensitive Bloom Filter for Approximate Membership Query," *IEEE Trans. Computers*, vol 61, no. 6, pp. 817–830, June 2012.

[29] A. Broder and M. Mitzenmacher, "Network applications of Bloom Filters: a survey," *Internet Mathematics*, vol. 1, no. 4, pp.485-509, May. 2004.

[30] B. Xia and Z. Tan, "Tighter bounds of the first fit algorithm for bin-packing problem," *ELSEVIER Discrete Applied Mathematics*, vol. 158, no. 15, pp. 1668–1675, Aug., 2010.

[31] The Network Simulator - ns-2, http://www.isi.edu/nsnam/ns

[32] Y. Gao, J. C. Hou and S. Paul, "RACCOOM: A rate-based congestion control approach for multicast," *IEEE Trans. Computers*, vol 52, no. 12, pp. 1521–1534, Dec. 2003.

[33] J. Wang, M. Yang, B. Yang and S. Q. Zheng, "Dual-homing based scalable partial multicast protection," *IEEE Trans. Computers*, vol 55, no. 9, pp. 1130–1141, Sept. 2006.

Xiaohua Tian received his B.E. and M.E. degrees in communication engineering from Northwestern Polytechnical University, Xi'an, China, in 2003 and 2006, respectively. He received the Ph.D. degree in the Department of Electrical and Computer Engineering (ECE), Illinois Institute of Technology (IIT), Chicago, in Dec. 2010. He is currently a post-doc in Department of Electronic Engineering of Shanghai Jiaotong University, China. He won the Highest Standards of Academic Achievement 2011 of IIT, Fieldhouse Research Fellowship 2009 of IIT, which is awarded to only one student over IIT each year, and the IEEE EFSOI 2010 Student Travel Grant Award. His research interests include application-oriented networking, Internet of Things and wireless networks. He serves as the guest editor of International Journal of Sensor Networks, publicity co-chair of WASA 2012. He also serves as the Technical Program Committee member for Communications QoS, Reliability, and Modeling Symposium (CQRM) of GLOBECOM 2011, Wireless Networking of GLOBECOM 2013 and WASA 2011.

Yu Cheng received the B.S. and M.S. degrees in Electrical Engineering from Tsinghua University, Beijing, China, in 1995 and 1998, respectively, and the Ph.D. degree in Electrical and Computer Engineering from the University of Waterloo, Waterloo, Ontario, Canada, in 2003. Since August 2006, he has been with the Department of Electrical and Computer Engineering, Illinois Institute of Technology, Chicago, USA, and now an Associate Professor. His research interests include next-generation Internet architectures and management, wireless network performance analysis, network security, and wireless/wireline interworking. He received a Postdoctoral Fellowship Award from the Natural Sciences and Engineering Research Council of Canada (NSERC) in 2004, and a Best Paper Award from the conferences QShine 2007 and ICC 2011. He received the National Science Foundation (NSF) CAREER award in 2011. He served as a Co-Chair for the Wireless Networking Symposium of IEEE ICC 2009, a Co-Chair for the Communications QoS, Reliability, and Modeling Symposium of IEEE GLOBECOM 2011, a Co-Chair for the Signal Processing for Communications Symposium of IEEE ICC 2012, and a Technical Program Committee (TPC) Co-Chair for WASA 2011. He is an Associated Editor for IEEE Transactions on Vehicular Technology and the New Books & Multimedia Column Editor for IEEE Network.