THE USE OF IP-ANYCAST FOR BUILDING EFFICIENT MULTICAST TREES

Dina Katabi

Massachusetts Institute of Technology Laboratory for Computer Science dinaktbi@mit.edu

Abstract

In this paper, we show that substantial improvement in multicast performance and reliability can be achieved by regarding a multicast group as a network region and using anycast to access it along the shortest path. We introduce the anycast-based tree (ABT), a novel architecture for building efficient shared multicast trees. ABT is a noncore tree; thus, it does not suffer from the traditional problems exhibited by core-based trees, such as traffic concentration or poor core placement. Moreover, ABT exhibits greater robustness and lower bandwidth consumption than other shared multicast trees. Our simulation shows that ABT reduces traffic concentration by 60% and decreases average bandwidth consumption by 25%.

In addition, the paper extends the Core based Tree multicast routing protocol with an ABT mode. The result is a new intra-domain protocol, called ACBT, that conserves the desirable features of the original protocol while reducing bandwidth consumption and alleviating traffic concentration.

1 Introduction

Early IP multicast protocols were based on Deering's work [6] and used source specific shortest path trees. According to this design, a multicast group builds multiple distribution trees, each rooted at one of the group's sources. This approach is simple and robust but the state information it stores in the routers scales as O(S*G), where G is the number of groups and S is the number of senders per group. As multicast has become more popular, and the number of potential senders larger, it has become obvious that sourcespecific-tree protocols, which spread per sender state, have severe scalability problems. This has led to the second generation of multicast protocols, which build one tree per group that is shared among all senders. Shared trees reduce the state overhead; however, their bandwidth consumption depends on the type of the tree built. Ideally, we would like to build a shared tree that minimizes the bandwidth con-

This work was supported by Defense Advanced Projects Research Agency (DARPA) contract number N66001-98-1-8903.

sumed to deliver traffic to all of the group members. However, building such a tree is an NP-complete problem widely known as the Steiner Minimal Tree (SMT) [15].

Since building shared SMTs is highly expensive, Wall suggested the use of core-based shared multicast trees [22]. A core-based tree is a shortest-path tree rooted at a core router and has the receivers as leaves. The core serves as a rendezvous point whereby receivers hear from all group sources, and sources reach all group members. Most of the shared tree multicast routing protocols, such as the Core Based Tree protocol (CBT) [2,3] and Protocol Independent Multicast Sparse Mode (PIM-SM) [9], build core-based trees.

However, core-based trees exhibit two major deficiencies. The first is traffic concentration around the core [4,21] caused by all senders addressing their packets to the core of the tree. This phenomenon leads to a high traffic density around the core router and a threat of congestion. This problem is exacerbated for high bandwidth groups especially when they are mapped to the same core router. Note that core routers are usually located in the center of their domains, which makes them critical waypoints for both multicast and unicast traffic. As such, their congestion harms all sorts of traffic in a domain: video streams, telnet, ftp, and many others. Figure 1 illustrates the traffic concentration problem.

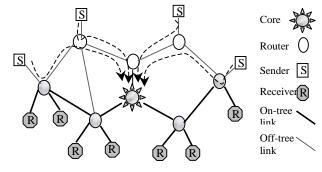


Figure 1: Traffic concentration
All senders address their packets to the core causing congestion.

The second problem is the possibility of a poor core placement [4,8,23]. Since the core constitutes the only

¹ Cores are expected to serve multiple groups since the number of groups in a domain is usually larger than the number of core routers.

point of distribution of traffic to receivers, its ideal location is in "the middle" of the corresponding group's members. However, because the core is chosen independently of the group's topology it might reside far from the receivers, causing the multicast tree to grow unnecessarily larger and to consume more bandwidth and storage space. Note that choosing the optimal core for the group is infeasible because the potential receivers are unknown at the group's creation time. Moreover, moving the core dynamically is a complex and expensive process. Figure 2 illustrates the poor core placement problem.

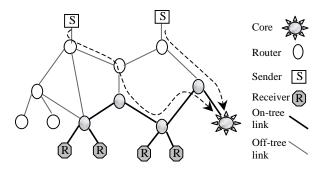


Figure 2: Poor core placement
The core is located faraway from the receivers, which causes
the tree to grow unnecessarily large

In this paper, we propose a novel shared tree architecture called anycast-based tree (ABT). ABT is a non-core multicast tree; thus, it does not suffer from the above mentioned problems. In addition, ABT exhibits lower bandwidth consumption and greater robustness than core-based trees. The paper extends the Core Based Tree multicast protocol with an ABT mode. The result is a new intradomain² multicast protocol called ACBT, which conserves the simplicity of the original protocol while reducing bandwidth consumption and alleviating traffic concentration.

The next section provides necessary background information. Section 3 is a high-level presentation of the design. In section 4, we explain the details of building and maintaining an ABT. We discuss both the advantages and overhead of the ACBT protocol in section 5. In section 6, we evaluate the design based on our simulation results. Finally, we discuss related work in section 7, and summarize our work in section 8.

2 Background

This section provides a summary of the Core Based Tree multicast routing protocol (CBT) and IP-anycast.

2.1 The Core Based Tree Multicast Routing Protocol (CBT)

The Core Based Tree multicast protocol constructs shared bi-directional trees rooted at a core router [2,3]. An interested receiver informs its next hop router about its desire in joining a particular multicast group. The router generates a join message, which travels towards the core setting up a transient join state in all the routers it crosses. Eventually, the join hits an on-tree router or the tree's core. The router generates a join ack, which traverses the reverse path of the corresponding join message guided by the transient state. When the acknowledgement arrives at the originating router, the branch is fully established.

2.2 IP-Anycast

IP-anycast is a network service whereby receivers that share the same characteristics are assigned the same anycast address. A sender interested in contacting a receiver with those characteristics sends its packet to the anycast address and the routers conspire to deliver the packet to the receiver nearest the sender, where nearest is defined according to the routing system measure of distance. Figure 3 illustrates IP-anycast.

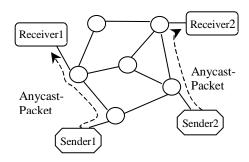


Figure 3: Illustration of IP-Anycast.

Receiver 1 and receiver 2 are in the same anycast group. If sender 1 sends a packet to the anycast address the network delivers it to receiver 1, but if sender 2 sends a packet to the anycst address the network delivers it to receiver 2.

Anycast was first introduced in RFC 1546 [21] as a method for discovering services in the Internet and for providing host auto-configuration.

An IP-anycast service is available under IPv6 [14]. In addition, the absence of this service from IPv4 does not hinder the deployment of our protocol in the current Internet. In fact, IP-anycast does not require any explicit support from routers. Therefore, a domain can provide an

 $^{^2}$ Currently, it is widely accepted that multicast routing will have 2 levels of hierarchy. Locally, a domain runs its choice of an intra-domain multicast protocol such as CBT or PIM. The internal trees are connected together by an inter-domain protocol such as BGMP (Border Gateway Multicast Protocol).

anycast service by choosing some unicast addresses to be used as anycast addresses, assigning them to multiple nodes, and advertising them using its internal unicast routing protocol [16,21]. Thus, ABT works correctly in a domain that does not explicitly support anycast.

3 Idea

The ACBT multicast protocol is engineered to solve the two major problems exhibited by CBT, which are poor core placement and traffic concentration around the core router. The solution relies on the observation that both problems arise only for high bandwidth groups. Low bandwidth groups, even when they are mapped to the same core, do not cause congestion since the total traffic remains low. Also, the increase in bandwidth consumption caused by mapping these groups to distant cores is insignificant. Therefore, using smaller non-core trees for high bandwidth groups solves both problems. This means we need a new rendezvous mechanism to replace the core, and we need a new mechanism that builds smaller distribution trees than core-based trees. Anycast provides us with both mechanisms.

Our anycast-based trees (ABTs) use anycast to locate a multicast tree and access it along the shortest path. In fact, a multicast tree can be viewed as a virtual network region. By assigning the same anycast address to all on-tree routers, the entire tree appears to any off-tree router as one network entity (one node), which can be addressed directly. As a result, data packets sent to the group travel along the shortest path connecting a source to the nearest on-tree router, which minimizes the off-tree part of their journey. Joins also travel along the shortest path from a potential member to the nearest on-tree router; therefore, they build short branches and reduce the size of the tree. The resultant tree considerably reduces the total bandwidth needed to deliver the traffic to all members. Moreover, the access point to a multicast tree is no longer tied to the core router. Each sender sees the nearest on-tree router as the access point to the tree, which prevents traffic concentration on any particular link.

Thus, ACBT is an extension of the CBT multicast protocol that builds any cast-based trees for high bandwidth groups and core-based trees for low bandwidth groups.

4 Design Details

In this section we describe the changes made to CBT to support the establishment of anycast-based trees.

4.1 Building an ABT

The initiator of a multicast session allocates a multicast address and announces the session with his estimate of the group's bandwidth requirement. A designated session announcement server (SAP server [13]) at an ABT domain uses the bandwidth estimate to decide whether the group should build an ABT or a CBT tree. For the case of an ABT, the designated SAP server acquires an anycast address and advertises the multicast group accompanied by its supporting anycast address on a locally scoped multicast group. Senders and receivers listen to this group to learn the multicast-to-anycast mapping. The details of allocating an anycast address depend on the domain's implementation of IP-anycast [16].

An ABT uses the group anycast address wherever CBT uses the core address. In contrast to CBT, where non-member senders encapsulate their packets and unicast them to the core, in ABT they encapsulate them in IP packets destined to the associated anycast address. The intervening routers deliver them to the nearest on-tree router, which forwards them on the tree using native multicast.

To join an ABT, a host multicasts on its attached link a join request, which contains both the group address and the associated anycast address.³ On receiving this request, a local router invokes the tree joining process (unless it has already) which follows one of two possible procedures. In case the router does not find an entry for the anycast address in its routing table, it infers that it is the first member to join this multicast group and it has the responsibility to initialize the associated anycast group. Therefore, the router follows a special serialization mechanism described in section 4.2 to ensure that no more than one router succeeds in starting the same ABT.

In case the joining router finds the anycast address in its routing table, the join process becomes significantly similar to CBT. More precisely, the router forwards the JOIN_REQUEST to the next hop on the path towards the anycast group. The join message sets up a transient join state in the routers it traverses, which consists of <group, previous hop, next hop, ABT-flag, anycast-address>. The ABT flag is one bit that differentiates between ABTs and CBTs. The anycast-address field is used only for ABTs. Eventually, the join is delivered to the nearest on-tree router, which sends a JOIN_ACK downstream. The JOIN ACK traverses the reverse path of the corresponding join message, which is possible due to the presence of the transient join state. The JOIN ACK causes each router that has a transient state for the corresponding group with the ABT flag set to initialize a timer whose expiration causes the router to start advertising itself as a member of the

³ A new message type can be added to IGMP [11] to enable a host to join an ABT.

anycast group. The timer, which we call the ANYCAST timer, decouples the anycast dynamics from the join dynamics and enables the domain to control the rate of the anycast updates.⁴

Once the acknowledgment reaches the router that originated the join message, the new receiver can receive traffic sent to the group.

To prove that ABT is loop-free, we assume that the existing tree does not contain a loop, and prove informally that a newly added branch does not create a loop. In ABT, all on-tree routers simulate the role of the core. In other words, the whole multicast tree shows the same input and output as a huge core, which has edges connected to all routers adjacent to the tree. Thus, the establishment of a new branch in an ABT simulates building the first branch in a CBT. Thus, if CBT is loop-free then ABT is also loop-free.

4.2 Serializing the initializations of an ABT

It is possible that multiple routers attempt to initiate the same ABT at the same time. Thus, there is a need for a mechanism that ensures no more than one of them succeeds in its attempt. Any serialization mechanism solves the problem. For example, a centralized serialization method based on a serialization server would do.

The solution we recommend is the following: The initiating router multicasts to the ALL-CBT-ROUTERS multicast group⁵ an ABT_INIT message informing the routers about its intention in starting this multicast-anycast group. The router waits for a period greater than twice the domain's average RTT.6 If it does not hear from any other router and still has no entry in its routing table for this anycast address, the router decides that it is the first on-tree router, declares itself the root for the multicast tree, and advertises itself as a member of the corresponding anycast group. On the other hand, if after multicasting the ABT_INIT message, the router hears an ABT_INIT multicast for the same group sent by a router with a smaller IP address, or if the router hears an objection, it fails in its attempt. A router should unicast an objection to the originator of an ABT_INIT multicast in either of the following situations: the router has a smaller IP address, and has itself sent an ABT_INIT multicast for the same

ABT group, or the router is the root of an ABT that has the same multicast address.

The mechanism described above breaks ties among multiple routers trying to initialize the same ABT by favoring the router with the smaller IP address.

Even if the serialization mechanism fails and two routers decide to be roots for the same multicast group, our design provides a mechanism for them to discover the multicast forest and merge it into one tree (section 4.5).

4.3 Pruning

An ABT is pruned both downstream-to-upstream and upstream-to-downstream. The former happens whenever a non-root router loses all of its children (and it is similar to CBT). The latter occurs when a root that has no directly attached members loses all of its children except one. In this case the root is no longer needed to connect the tree and can resign, leaving its position to its only child. To do so, the root leaves the anycast group, sends to its child a ROOT QUIT message, and ROOT QUIT ACK. The root keeps trying to resign as long as it thinks it is connected to its child. Similarly to CBT, the root discovers loss of connectivity from the absence of echo messages and reacts by flushing the tree. A child that receives a ROOT QUIT from its parent replies by a ROOT_QUIT_ACK and declares itself a root.

This mechanism helps the tree to move flexibly in the domain and reshape itself according to the changes in the group topology.

4.4 Maintenance

Tree maintenance is performed as in CBT. The only difference is that a flush message causes the receiver to leave the corresponding anycast group, before it propagates the flush downstream.

4.5 Healing from a network partition

In contrast to core-based trees, where a network partition prevents all senders and receivers separated from the core from communicating, ABT enables senders and receivers in a partition to communicate on a local tree and merges these trees quickly when the partition heals. To discover other instances of a multicast tree, ABT root routers listen to the multicast group for Domain Wide Reports [12]. If a root hears its group reported by another router that has a smaller unicast address, it joins that router as if it were joining a core. More precisely, this root sends a ROOT_JOIN_REQUEST towards the root with the

⁴ One periodic timer is sufficient for all pending ABT groups. Moreover, the ANYCAST timer can be made adaptive to allow fast anycast advertisements after a long period with low join-leave activity.

⁵ Recall that ACBT is an extension to CBT. This group is defined by CBT and has been assigned the address 224.0.0.15. The group is used for purposes that are not of interest in the context of this paper. ABT uses this group to perform the serialization and the merge processes.

⁶ The administrator can easily configure the routers with an estimate of the average RTT. The estimate need not be very accurate.

⁷ Flush and echo messages are defined by the CBT protocol [3].

⁸ Domain Wide Reports are periodic messages multicasted by core routers or ABT roots to report their groups. They play a role in integrating intradomain multicast routing with inter-domain multicast routing.

smaller unicast address, to be referred to hereafter as the preferred root. On receiving this message, routers that do not have an entry for this multicast group establish a transient state for the group and propagate the message towards the preferred root. Routers that have an entry for this group check their tables, and if their corresponding root matches the preferred root, they acknowledge the join back causing the transient branch to be confirmed.⁹ If an on-tree router receives a ROOT JOIN REQUEST and upon checking its table finds that the root of its tree differs from the preferred root, it quits its parent, sets up a transient join state, then propagates the join toward the preferred root. Eventually, the ROOT_JOIN_REQUEST either hits an on-tree router whose root is the preferred one. or it is received by the preferred root itself. In both cases the router acknowledges the request. Once a router receives a ROOT JOIN ACK for which it has a transient state, it confirms the state, propagates the ack downstream, and if it has not done it already, it starts advertising itself as a member of the associated anycast group. On the other hand, if a transient ROOT JOIN REQUEST state times out at any non-root router which has children, the router sends a flush message to all of its children, and stops advertising itself as a member of the corresponding anycast group.

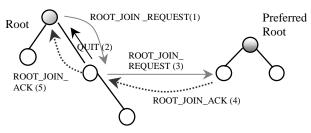


Figure 4: An illustration of the merge process Numbers next to messages indicate their order

4.6 Forwarding Data packets

An on-tree router forwards a multicast packet over all the tree edges except the one on which the packet arrived. It does not accept multicast traffic delivered over an off-tree link. The router accepts all anycast packets addressed to the associated anycast group, decapsulates them and propagates them over all its on-tree interfaces.

5 Discussion and Evaluation

This section analyzes the overheads and benefits of the ACBT protocol, both of which are closely related to the characteristics of ABT trees.

5.1 Overheads

We identify the following as potential sources of overhead in the system:

5.1.1 The effect of ABT on anycast routing

Routers in an ABT domain route the supporting anycast addresses and grant them entries in their routing tables. This puts some overhead on anycast routing. However, the design enables each domain to control the anycast overhead to stay within the limit of the available routing resources. First, by controlling ACBT's operating point the domain controls the growth of the routing tables. More precisely, ACBT operates between 2 extreme points. At one extreme, all of the multicast groups build ABTs. At the other extreme, all groups build CBTs. The closer the operating point is to the case where all trees are ABTs, the greater the reduction in bandwidth and traffic concentration. A domain controls the exact operating point by choosing the maximum number of ABTs and the bandwidth threshold beyond which a group builds an ABT. Second, in addition to controlling the growth in the routing tables, the domain controls the anycast dynamics using the ANYCAST timer so that they never overwhelm the routers.

In practice, we expect high bandwidth groups to generate most of the multicast traffic in a domain. Most domains will be able to provide the majority of these high bandwidth groups with ABTs for the following reasons:

- 1- Internal routers in a domain usually have small routing tables and do not suffer from lack of memory or CPU cycles.
- 2- The number of high bandwidth groups is limited by the domain's bandwidth capacity, so the absolute number of such groups should not be large (e.g., a link can carry a limited number of video sessions).
- 3- The overhead of anycast routing is not large. Anycast addresses are routed through the domain's unicast routing protocol. In domains that use RIP [17], the anycast addresses get advertised through RIP's periodic updates and do not generate any additional traffic. In domains that use OSPF [19], which generates triggered updates, the anycast routes might occasionally generate additional messages when there are no other routing changes in the domain. Fortunately, the fact that the anycast route is the only change to report means that the routing is stable enough and routers have resources to process the messages.

⁹ On-tree routers learn the IP-address of the root from the join ack they receive when they become attached to the tree.

5.1.2 Negligible delay in initializing ABTs

The serialization mechanism introduces a small delay in initializing a multicast tree (twice the domain's average RTT). However, this delay constitutes a negligible fraction of the initialization latency, which is mostly caused by multicast address allocation and advertisement. Even if future solutions reduce the time for address allocation and announcement, the absolute delay for the serialization should be small enough to satisfy all or most applications.

5.1.3 Consumption of an anycast address per ABT

An ABT consumes both a multicast and an anycast address. However, the domain controls the number of anycast groups to stay within the limit of the available anycast addresses. Moreover, since ACBT is an intradomain multicast protocol, the anycast addresses it uses need not be unique and can be reused in other domains.

5.2 Advantages

The design has the following advantages:

5.2.1 Alleviation of traffic concentration

The most difficult problem with shared tree protocols is traffic concentration at a core or rendezvous point. The problem can happen at two levels. First, different sources send simultaneously to the same group and cause congestion at the core. Second, multiple multicast groups are mapped to the same core causing their trees to unnecessarily share some links around the core, and their packets to be delivered over the same links. By using ABTs for groups with high bandwidth requirements, ACBT alleviates traffic concentration at both levels. By reproducing the scenario in Figure 1 and comparing it against an ABT approach, Figure 5 illustrates how ABT solves traffic concentration caused by all sources addressing their packets to the core. Similarly, Figure 6-a

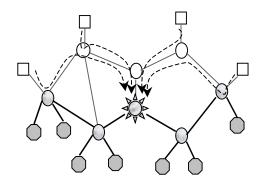
shows two high bandwidth groups mapped to the same core router and overloading one of the links. Figure 6-b shows the same setting with ABTs, which achieves a more balanced traffic distribution.

5.2.2 Reduction in consumed bandwidth

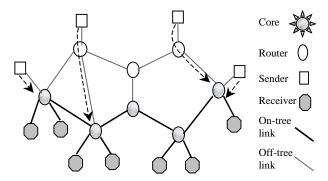
By using ABTs for high bandwidth groups, ACBT considerably reduces the overall consumed bandwidth. ABT attempts always to build the shortest possible branch. Therefore, its bandwidth consumption on average is much better than other shared trees. This characteristic is particularly important for applications that consume a large amount of bandwidth such as video sessions. Figure 7 is a reproduction of the scenario in Figure 2 with both CBT and ABT. It shows that by using ABT we solve the bad core placement problem and build shorter and more efficient trees.

5.2.3 Robustness

Because of the distributed nature of its design, an ABT is more robust than other shared trees. First, there is no single point of failure, as the core is for other shared trees. Second, ABT trees do not rely on a bootstrap router (BSR) to locate their cores [10], and consequently do not fear a mistake made by a BSR that harms all the multicast sessions in the domain. Third, in contrast to core-based trees, where a network partition prevents all senders and receivers separated from the core from communicating, ABT enables senders and receivers in a partition to communicate on a local tree and merges these trees quickly when the partition heals (as quickly as the time it takes to send a domain wide report and join the preferred root). This high robustness indicates that a domain might decide to build ABTs for some low bandwidth multicast groups based on their fault-tolerance requirements.

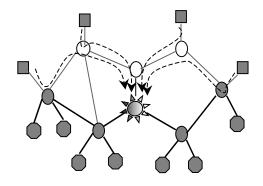


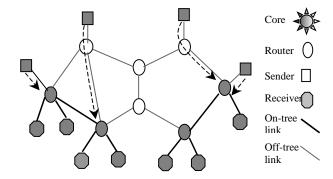
(a) Reproduction of Figure 1 In CBT all senders address their packets to the core causing congestion.



(b) Same scenario in Figure 1 with ABT In ABT each sender addresses his packets to the nearest on-tree router

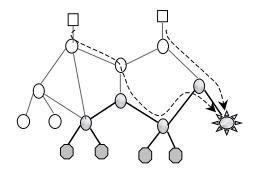
Figure 5: ABT solves traffic concentration caused by multiple senders addressing their traffic to the core router



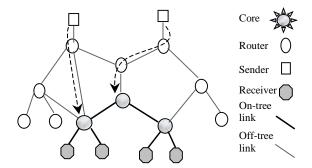


- (a) Traffic concentration in CBT caused by mapping multiple groups to the same core router.
- (b) Same scenario in ABT but there is no concentration because each sender addresses his packets to the nearest on-tree router

Figure 6: ABT solves traffic concentration caused by mapping multiple groups to the same core



(a) Reproduction of Figure 2 In CBT, the core is located faraway from the receivers, which causes the tree to grow unnecessarily large



(b) Same scenario with ABT ABT solves the poor core placement problem and builds smaller trees

Figure 7: ABT solves the poor core placement problem

5.2.4 Reduction in average delay and join latency

ABT reduces average propagation delay and join latency since packets travel along the shortest path connecting a source to a tree and are delivered over a small size tree with short branches. In addition, it reduces the queuing delay because it reduces traffic concentration.

5.2.5 Fanout control

In ABT, a router can control its fanout to some extent by leaving the anycast group. This would prevent the router from attracting additional join requests. However, the router might still receive joins if it is on the shortest path to other on-tree routers.

6 Simulation

This section provides a quantitative description of the potential improvement perceived when ABT is used. Our

simulation shows that ABT reduces traffic concentration by 60% and decreases the average bandwidth consumption by 25%.

6.1 Simulation Environment

The simulations are carried out in ns2 [20] using our own ABT and CBT modules. Our CBT module is based on CBTv2's specifications described in RFC 2189. Note that the simulations compare all groups using CBTs to all groups using ABTs. The actual performance improvement

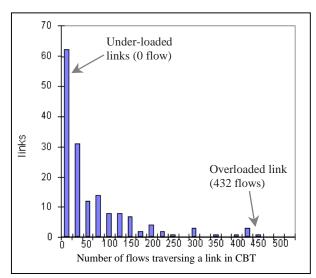
¹⁰ In an earlier CBT proposal described in [2], packets sent toward the core carried an IP option, allowing the first on-tree router on the path to the core to forward them using native multicast. This can reduce the off-tree portion of the routes in CBT. However, this approach was abandoned because IP options are not widely implemented and can slow down packet processing considerably. We have chosen to carry out our performance comparison using the more recent version of CBT (version 2). Nonetheless, ABT would still outperform the earlier CBT as it builds more efficient trees and uses a shorter off-tree path.

of ACBT will depend on the percentage of multicast traffic that uses ABT. While it is impossible to quantify this precisely, as explained above in section 5.1.1, we expect most multicast traffic to use ABT.

The graphs used in the simulation are generated using the Georgia Tech ITM topology generator [5,24], which generates topologies that resemble typical networks. We used both the Doar-Leslie and the Waxman edge connection methods to generate multiple 50-node and 100-node graphs. The average edge degree is around 3 and the number of bicomponents is in the range {11-28}. These parameters are chosen based on the information in [24].

The simulation uses a static anycast routing. We think this is a justified simplification given that the current join-leave statistics collected over the MBone [1] have a much longer time scale than the unicast/anycast routing updates. Even if the join dynamics had a similar time scale, our simplification would still be accurate provided that consecutive joins are not generated by closely located receivers. If future changes in multicast dynamics make static anycast routing a poor approximation, then the advantage of ABT may be reduced but it will still outperform CBT.

Senders and receivers are chosen randomly among the domain's nodes. We choose the CBT core to be the node with the highest edge degree. In case there are more than one of them, we choose the one closer to the center of the graph. This choice is the best given that one doesn't know the future group topology [4]. A random choice of CBT's core would have increased the improvement perceived when ABT is used instead of CBT.



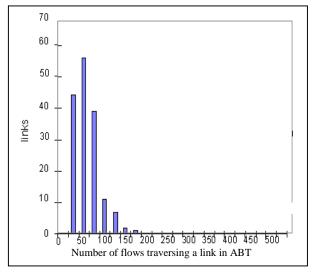
(a) Traffic Distribution in CBT. It shows high traffic concentration on some links (432 flows) while others are underutilized.

6.2 Traffic Concentration

We define a flow to be the stream of packets from a particular sender to a multicast group. Traffic concentration is measured by the maximum number of flows traversing a unidirectional link (the load of the most congested link.) Each run uses 76 groups with 7 senders and 7 receivers per group. We ran 10 simulations. The results, which are given in table 1, show that ABT reduces traffic concentration by 60%.

CBT	ABT	Ratio
Max link load	Max link load	ABT/CBT
432	153	0.35
430	153	0.36
428	179	0.42
446	186	0.41
445	186	0.41
446	193	0.43
444	193	0.43
425	186	0.44
427	193	0.45
432	193	0.45

Table 1: The ratio of the maximum link load in ABT to that in CBT.



(b) Traffic Distribution in ABT. The maximum link load is significantly less than CBT (186 flows) and it utilizes the links fairly.

Figure 8: Traffic distribution in CBT (a) and ABT (b)

Figure 8 shows the traffic distribution in CBT and ABT for the first entry in Table 1. Note that in the CBT simulation some links are overloaded (432 flows) while others are underutilized (0 flows). On the other hand, the ABT simulation shows good link utilization. The maximum link load has decreased to less than half its value with CBT, which decreases the possibility of congestion. Furthermore, the ABT simulation shows no underutilized links.

6.3 Bandwidth and Delay

This simulation uses 2 sets of groups. The first set has a group size of 10% the domain size. The second set has a group size of 20% the domain size. We run 25 different simulations in each set to measure the average delay, the maximum delay, and the tree's cost for both CBT and ABT.

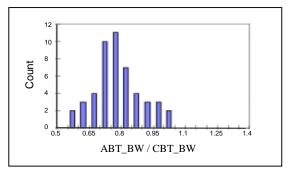


Figure 9: A Histogram of bandwidth consumption in ABT compared to its counterpart in CBT

Figure 9 is a histogram of the ratio of the bandwidth consumed by ABTs to that consumed by the corresponding CBTs. (*ABT_BW/CBT_BW*). The Figure exhibits that ABT reduces average bandwidth consumption by 25%. This reduction constitutes a significant savings especially for high bandwidth groups.

Figure 10 and Figure 11 are histograms of the ratios of the average and maximum propagation delay in ABT to their counterparts in CBT. They reveal that ABT has better delay performance than CBT. Note that we simulate only the propagation delay. In real life pathological delay cases are mostly due to queuing delay and retransmission delay. Since ABT alleviates congestion by eliminating traffic concentration around the core we expect it to improve delay significantly, even more than what our figures indicate. Although ns enables us to simulate queuing and retransmission delays, we chose not to do so because such simulation would heavily depend on the traffic pattern, the queuing model, the maximum queue size and the retransmission timeout. Instead, we simulate propagation delay, which is a lower bound on the expected improvement.

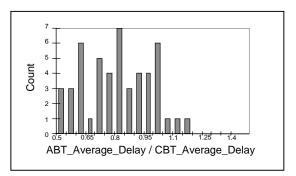


Figure 10: A Histogram of average delay in ABT compared to its counterpart in CBT

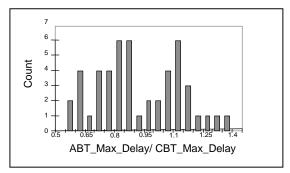


Figure 11: A Histogram of maximum delay in ABT compared to its counterpart in CBT

7 Related Work

The problem of building the best shared-multicast-tree has been repeatedly investigated in the multicast literature. Wall proposed the use of shared trees routed at a core router [22]. His approach, which exhibits a good compromise between simplicity and efficiency, was adopted by all of the known shared tree multicast routing protocols such as PIM-SM [9] and CBT [2,3]. On the other hand, Doar & Leslie proposed in [6] what they call "the naïve multicast algorithm", which computes the multicast route by combining the shortest path across initial members of the group, then joining new members to the nearest attachment point on the tree. However, the algorithm assumes a global knowledge of the network topology, which is usually unavailable.

On the other hand, Estrin and Wei studied the tradeoffs between different multicast trees [23]. Their results show that, on average, core-based trees exhibit good performance; however, they might suffer severe congestion caused by traffic concentration around the core router.

Zegura et al. studied a variety of methods for core selection [4,8]. They concluded that a poor core placement

degrades performance significantly. They proposed core migration to overcome the problem. However, core migration requires periodically evaluating the performance of candidate cores and electing the core with the best performance, which results in a considerable overhead in terms of state and bandwidth. This paper does not address issues such as how senders and new receivers learn about the new core and how to migrate the core without disturbing the communication.

8 Conclusion

In this paper, we show that substantial improvement in performance and reliability can be achieved by regarding a multicast group as a network region and using anycast to access it along the shortest path.

We introduce the anycast-based tree (ABT), a novel architecture for building efficient shared multicast trees. ABT is a non-core tree; thus, it does not suffer from the traditional problems exhibited by core-based trees, such as traffic concentration and bad core placement. Moreover, ABT is more robust than other shared multicast trees.

To build an ABT, we make all the on-tree routers join the same anycast group. As a consequence, the multicast tree appears to all off-tree routers as one addressable network entity. We use this property to direct data packets and join requests along the shortest path to the tree, decreasing bandwidth consumption and join latency. Our simulation shows that ABT reduces traffic concentration by 60% and decreases average bandwidth consumption by 25%. Although our design uses IP-anycast, the absence of the service from IPv4 does not hinder the deployment of ABT in the current Internet. ABT works correctly in a domain that does not support anycast explicitly.

In addition, the paper extends the Core based Tree multicast protocol with an ABT mode. The resultant protocol, which we call ACBT, builds ABTs for high bandwidth multicast groups and CBTs for low bandwidth multicast groups. As a result, ACBT conserves the desirable features of the CBT protocol while reducing bandwidth consumption and alleviating traffic concentration.

Acknowledgments

The author is indebted to John Wroclawski for extensive discussions that greatly improved this paper. Also, the author thanks Lee Breslau, Chandrasekhar Boyapati, Charles Blake, Dorothy Curtis, Mangesh Kasbekar, and the Global Internet reviewers for their constructive comments on different versions of this paper.

References

- K. Almeroth and M. Ammar, "Multicast Group Behavior in the Internet's Multicast Backbone (MBone)", *IEEE Communications*, Jun. 1997.
- A. Ballardie, P. Francis, and J. Crowcroft, "Core Based Trees (CBT) an Architecture for Scalable Multicast Routing," *In Proc. of SIGCOMM'93*, 1993.
- 3. A. Ballardie, "Core Based Trees (CBT version 2) Multicast Routing --Protocol Specifications--," RFC 2189, Sep. 1997.
- K. Calvert, E. Zegura, and M. Donahoo, "Core Selection Methods for Multicast Routing," In Proc of ICCCN'95, 1995
- K. Calvert, M. Doar and E. W. Zegura, "Modeling Internet Topology," *IEEE Communications*, Jun. 1997.
- 6. S. Deering, "Multicast Routing in a Datagram Internetwork," PhD Thesis, Sanford Univ., Dec. 1991.
- 7. M. Doar and I. Leslie, "How Bad is Naïve Multicast Routing?," *In Proc. of INFOCOM'93*, 1993.
- 8. M. Donahoo, E. Zegura, "Core Migration for Dynamic Multicast Routing," *In Proc. of ICCCN'96*, 1996.
- 9. D. Estrin et al, "Protocol Independent Multicast (PIM) Sparse Mode" Internet-Draft, 1996.
- D. Estrin et al., "A Dynamic Bootstrap Mechanism for Rendezvous-based Multicast Routing," In Proc. INFOCOM'99, 1999.
- 11. W. Fenner, "Internet Group Management Protocol version 2 (IGMPv2)," Internet-Draft, 1996.
- 12. W. Fenner, "Domain Wide Multicast Group Membership Reports," Internet-Draft, Aug. 1999.
- M. Handley, "On Scalable Internet multimedia Conferencing," PhD Thesis, University College London, Nov. 1997.
- R. Hinden, S. Deering, "IP version 6 Addressing Architecture," RFC 2373, Jul. 1998
- 15. M. Imase and B. M. Waxman, "Dynamic Steiner Tree Problem," SIAM Journal Disc. Math, Aug. 1991.
- D. Katabi and J. Wrocławski, "A Framework for Global IP-Anycast," Internet-Draft, Jun. 1999.
- 17. G. Malkin, "RIP Version 2," RFC 1723, Nov. 1994.
- 18. D. Meyer, "Some Issues for an Inter-domain Multicast Routing Protocol," Internet-Draft, Nov. 1997.
- 19. J. Moy, "OSPF Version 2," RFC 2328, Apr. 1998.
- Network Simulator ns (version 2), http://www-mash.cs.berkeley.edu/ns/
- C. Partridge, T. Mendez, and W. Milliken, "Host Anycasting Service," RFC 1546, Nov. 1993.
- D. Wall, "Mechanisms for Broadcast and Selective Broadcast," PhD Thesis, Stanford Univ., Jun. 1980.
- L.Wei and D. Estrin, "The Trade-offs of Multicast Trees and algorithms," In Proc. of ICCC'94, 1994.
- E. W. Zegura, K. Calvert and S. Bhattacharjee, "How to Model an Internetwork," *In Proc. of INFOCOM* '96, 1996.