Multicast Routing Extensions for OSPF

J. Moy

Abstract

The OSPF protocol has been extended to handle the routing of IP multicast datagrams. This has applications in many areas, such as voice and video conferencing, distributed databases, games and simulations. This paper describes the characteristics of multicast routing in OSPF, together with a description of protocol mechanisms and a brief analysis of protocol performance. Experience with the protocol to date is also presented.

1. Introduction

OSPF, a TCP/IP link-state routing protocol, has recently been enhanced to allow the routing of IP multicast datagrams. The resulting protocol, capable of routing both unicast and multicast traffic, has been called the Multicast Extensions for OSPF or *MOSPF*. (For a brief introduction to TCP/IP routing, and OSPF in particular, see the sidebar on Page 5).

Multicasting is the ability for an application to send a single message to the network and have it be delivered to multiple recipients at possibly different locations. A good example of this is a multi-site audio/video conference. Distributed simulations and games, such as a tank battle simulation involving several geographically separated participants, are other examples. Commercially, distributed databases such as stock quotations generated in a central location and then delivered to many separate traders, are good candidates for multicast.

IP multicast is an extension of LAN multicasting to a TCP/IP Internet. IP Multicast permits an IP host to send a single datagram (called an IP multicast datagram) that will be delivered to multiple destinations. IP multicast datagrams are identified as those packets whose destinations are class D IP addresses (*i.e.*, addresses whose first byte lies in the range 224-239 inclusive). Each class D address is said to represent a multicast group.

The extensions required of a computer running TCP/IP in order to participate in multicasting have been defined for some time. A protocol called the *Internet Group Management Protocol* (*IGMP*) is used by TCP/IP applications in order to join and leave particular multicast groups. An IP datagram addressed to the group address will be delivered to all group members, assuming that there are multicast routers (for example, routers running MOSPF) connecting the source and group members.

MOSPF uses IGMP to discover the location of group members. The group members are then pinpointed in the routing database, which is essentially a map of the internetwork. This in turn allows the MOSPF routers to calculate an efficient path for each multicast datagram.

2. MOSPF Features

The following sections describe MOSPF features in detail. This description is divided into two parts: 1) a characterization of the path a multicast datagram will take in MOSPF and 2) miscellaneous features that allow MOSPF to apply to a large range of environments.

The sample IP internet pictured in Figure 1

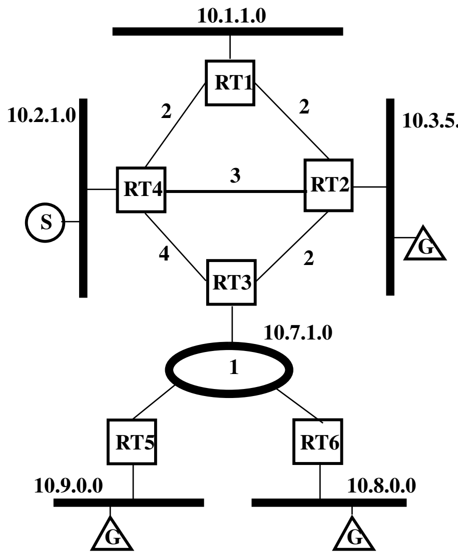

Figure 1: A sample IP internetwork. Six routers are pictured. labeled RT1 through RT6. Routers connect to LAN segments or to other routers via serial lines. Each LAN segment has been assigned an IP network number. Group members are indicated by triangles labeled with the letter G. The source used to illustrate datagram paths is marked with an S. Serial lines and transit LANs are marked with positive integers indicating the cost of transmitting a packet on the network

will be used to illustrate MOSPF's properties.

2.a. Path Characteristics

As MOSPF forwards an IP multicast datagram from its source to the various group members, the routers and network segments that the datagram transits form a graph without loops or a *tree*. The tree is rooted at the datagram source, and all branches terminate at group members. The datagram is replicated when, and only when, two branches of the path diverge. Postponing replication of the datagram in this way conserves network bandwidth. If a single link leads to multiple group members, only a single copy of the datagram traverses the link; if necessary it will be replicated later. This replication is sometimes performed by a MOSPF router, and is sometimes accomplished by using the multicast capabilities of the data-link layer. For example, the path of a multicast datagram with Source S and Destination Group G in Figure 1's internetwork is displayed in Figure

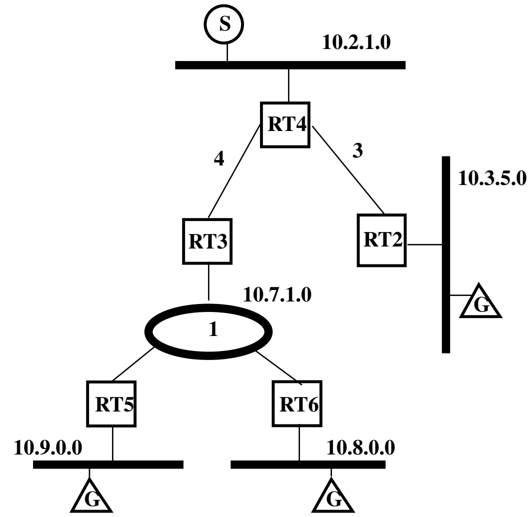

Figure 2: The path of a multicast datagram in MOSPF, originated by Source S and destined for Group G. Note that it forms a tree, rooted at S, with branches terminating at member of Group G.

2. Along the way the datagram is replicated twice, once by Router RT4 and the second time as the datagram is sent onto network segment 10.7.1.0 as a data-link multicast and hence picked up by both Routers RT5 and RT6.

The rest of this section describes additional characteristics of the multicast datagram's path.

Since the multicast's path is calculated as a tree rooted at the datagram source, the path taken by a multicast datagram depends both on the datagram's source and its multicast destination. Called *source/destination routing*, this is in contrast to most unicast datagram forwarding algorithms (including OSPF's) that route based solely on destination. Taking account of the source when making routing decision means a lot more routing calculations (see Section 6 below), however it achieves very good paths in terms of network usage and delay to individual group members.

In fact, the path taken between the datagram's source and any particular destination group member is the least cost path available. In most instances the path of the datagram to an individual group member is identical to that of a datagram sent to the group member's unicast address[1].

However, while MOSPF optimizes the path to any given group member, it does not necessarily optimize the use of the internetwork as a whole. To do so, instead of calculating source-based shortest-path trees, something similar to a minimal spanning tree (containing only the group members) would need to be calculated. This type of minimal spanning tree is called a Steiner tree in the literature. Figure 3 displays what the path of our sample multicast datagram would be in such a scheme. For a comparison of shortest-path tree routing to routing using Steiner trees, see [4] and [7].

IP datagrams are labeled with their *Type of Service* (*TOS*) classification, which can take one of five mutually exclusive values: *minimize delay*, *maximize throughput*, *maximize reliability*, *minimize monetary cost* and *normal service*. The path of a multicast datagram in MOSPF can vary based on its TOS classification. For example, delay sensitive multicast traffic can take separate paths from high throughput multicast applications. As with OSPF unicast routing, TOS-based multicast routing in MOSPF is optional, and routers supporting it can be mixed freely with those that do not.

In an additional optimization, MOSPF provides explicit support for IP multicast's *expanding ring search* procedure. In an expanding ring search, an application finds the nearest server by sending out successive multicasts, each with a larger *Time to live* (*TTL*). (in IP, the TTL indicates the maximum number of routers that are allowed to handle a particular datagram before it is discarded). The first responding server will then be the closest[2]. MOSPF minimizes the network bandwidth consumed by an expanding ring search by ceasing to forward a multicast datagram along its path as soon as the datagram's TTL is determined to be too small to ever reach another group member. As an example, consider the datagram path in Figure 2. Source S could do an expanding ring search to discover the closest member of Group G. The first datagram would be sent with a TTL of 1, and would discover only group members on Source S's own network segment (of which there are none). The second datagram, having TTL of 2, would discover the group member on 10.3.5.0. This second datagram would not even be forwarded to Router RT3,

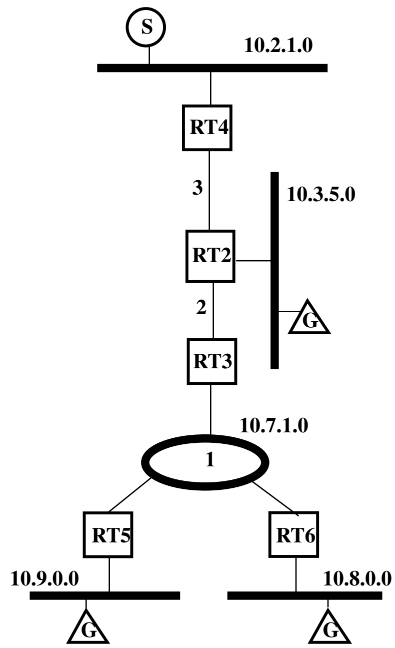

Figure 3: The path of a multicast datagram that would be chosen if one were trying to minimize overall internet-wide cost. This path has a total cost of 6, as compared to the MOSPF path pictured in figure 2 which has a total cost of 8. However, the cost of the path to the Group members on networks 10.9.0.0 and 10.8.0.0 ends up being larger in the above path than in the

1. The cases where the paths are not the same are when MOSPF routers are mixed with non-MOSPF routers, or when datagrams cross OSPF area and/or Autonomous Systems with asymmetric link costs. See Sections 2.b and 3.a below.

2.Closest in terms of hops, but not necessarily in terms of the OSPF metric.

since Router RT4 knows that a datagram with original TTL of 2 could never reach either of the two network segments at the bottom of the figure (10.9.0.0 and 10.8.0.0).

2.b. Miscellaneous Features

Other features of the MOSPF protocol are described in this section:

MOSPF routers can be mixed with non-multicast OSPF routers. When this is done, all routers will interoperate in the routing of unicasts. Unicast routing will not be affected by this mixing; all unicast paths will be the same as before the introduction of multicast. This mixing of multicast and non-multicast routers enables phased introduction of a multicast capability into an internetwork.

All network segment types that are supported by the base OSPF protocol are also supported by MOSPF: *broadcast networks* (e.g., ethernet), *point-to-point networks* (e.g., synchronous serial lines) and *non-broadcast multi-access* (*NBMA*) networks (e.g., an X.25 PDN or a Frame relay network). However IGMP is not defined on NBMA networks, so while these networks can support the forwarding of multicast datagrams, they cannot support directly connected group members. As an example, MOSPF can forward multicast datagrams over an X.25 PDN, but a group member cannot be attached directly to the X.25 PDN.

MOSPF provides algorithms for forwarding multicast datagrams between OSPF areas[3], and an algorithm for importing multicast datagrams from another Autonomous Systems. In addition, a way of forwarding multicast datagrams to the Autonomous System boundary, from where they can be distributed to other Autonomous Systems, is described (see Section 3.a). These features allow MOSPF to be used as part of the worldwide Internet's multicast fabric (see Section 4.a).

Load sharing, the splitting of a single datagram stream across multiple equal cost paths (also sometimes called multipath), is not supported. Between the multicast datagram's source and any particular group member there is at most a single path. This is forced by the very nature of multicast routing: In order to avoid unwanted replication of multicast datagrams, each router must know on which interface a given datagram should be received, and must discard the datagram if it is received on any other interface.

3. Protocol Mechanisms

OSPF is a link-state (unicast) routing protocol. As such, it keeps a database describing a map of the internetwork's interconnections (see sidebar). In order to calculate optimal paths for multicast datagrams, MOSPF augments the OSPF link-state database with new information, and defines an additional calculation to be performed on the database. The rest of this section describes these additions in detail.

MOSPF routers use the Internet Group Membership protocol (IGMP) to establish the location of group members, by sending *IGMP Host Membership Queries* and receiving *IGMP Host Membership Reports* in return. A MOSPF router then distributes the group location information throughout the Autonomous System by flooding a new type of OSPF link-state advertisement, the group-membership-LSA. This LSA labels the pieces of the map having group members. It also condenses the group membership in the process; IGMP keeps track of membership on a per-network segment basis instead of per-host, and MOSPF takes this one step further by not

3.OSPF allows an Autonomous System to be divided into separate pieces called OSPF areas (see sidebar). This can lead to a reduction in the amount of routing protocol overhead. Area boundaries can also be imposed to implement administrative concerns such as information hiding.

OSPF Routing

The *Open Shortest Path First* (*OSPF*) protocol is a TCP/IP routing protocol documented in [2]. In TCP/IP, routers forward (unicast) IP datagrams based on their destination *IP address*. IP addresses are 32-bits in length, and are usually expressed in dotted decimal notation, such as 10.0.0.1. TCP/IP routes to network segment, not individual hosts. Each network segment is assigned a range of IP addresses, and it is these ranges that are advertised by TCP/IP routing protocols.

In TCP/IP, an *Autonomous System* (*AS*) is a collection of routers under a single administrative control and running a common routing protocol. Such routing protocols are classified as *Interior Gateway Protocols* (*IGPs*). OSPF is the recommended IGP for use in the TCP/IP Internet.

OSPF is based on *link state routing* technology. The core of every link state routing protocol is a distributed and replicated database. This database, *called the link state database*, is essentially a dynamic map of the internetwork, describing the internetwork's components and their current interconnections. The constituent pieces of the database are called *link state advertisements* (*LSAs*). Each LSA describes a localized piece of the internetwork. Each router maintains an identical copy of the entire link state database. Synchronization is achieved through a *reliable flooding* mechanism, which ensures that when one LSA changes, all routers shortly receive identical copies of the new information. From this database, each router calculates the set of "best" paths to use when forwarding datagrams. The most common path calculation (and the one OSPF uses) is called the *Dijkstra algorithm* (see Chapter 31 of [9]); this produces a tree of shortest paths rooted at the calculating router.

OSPF's LSAs describe a local portion of the Autonomous System, and are classified by function: *router-LSAs* describe a particular router's working interfaces and *network-LSAs* list all routers connected to a particular network.

An OSPF Autonomous System can be divided into separate pieces called *OSPF areas*. An AS may be broken up into areas for policy reasons, or because doing so can reduce the amount of routing protocol overhead. OSPF areas, which are named by *Area IDs* (expressed in dotted decimal notation), can be viewed as being organized in a two level hierarchy. At the top level is the single backbone area (Area ID 0.0.0.0), to which all other areas connect. Each other area advertises its internal networks to the backbone in *summary-LSAs*, which are distributed throughout the backbone and eventually readvertised into the other non-backbone areas.

Information on how to reach network segments belonging to different Autonomous Systems can also be imported into an OSPF AS via *AS-external-LSAs*.

The OSPF protocol is extended by adding new types of LSAs. In particular, MOSPF has been implemented by adding the *group-membership-LSA*, which describes the location of multicast groups in the OSPF Autonomous System.

labeling leaf network segments (such as network segments 10.9.0.0 and 10.8.0.0 in Figure 1) but instead labeling their servicing MOSPF router (Routers RT3 and RT6 respectively). Condensing membership information reduces the size of the link-state database, and streamlines the MOSPF routing calculation.

The MOSPF routing calculation is then performed in an "on-demand" fashion. The first time a multicast datagram having a given source and destination is received, the MOSPF router calculates a shortest-path tree rooted at the datagram source, using the labeled map as input. Those branches not having group members are pruned, so that a datagram is not forwarded any further than is necessary. In fact, the number of router hops to each group member is also calculated, so that datagrams with insufficient TTLs can also be dropped. The results of these on-demand tree calculations are then cached for later use by subsequent matching datagrams.

The MOSPF routing calculation is very similar to OSPF's unicast routing calculation, both using the Dijkstra algorithm to calculate shortest-path trees. However, in MOSPF there are potentially many different trees calculated, possibly one for every datagram source and destination combination. For any given datagram, tie breakers have been defined ensuring all MOSPF routers calculate an absolutely identical shortest-path tree. This is absolutely essential for correct multicast datagram forwarding.

While forwarding the multicast datagram along its pruned, source-based shortest-path tree, MOSPF routers make use of link-level multicast services, when they exist. This improves datagram latency, while reducing demands on router processing and on LAN segment bandwidth.

3.a. Using Incomplete Information

The algorithms described in the previous section work fine within a single OSPF area. However, when a multicast datagram crosses OSPF area boundaries, complications arise. This is because an OSPF area's map is not visible external to the area complicating the building of the source-based trees. Adding to the problem is the fact that group membership is not freely advertised across area boundaries.

OSPF areas can be viewed as being organized in a two level hierarchy, with the top level consisting of the single backbone area (see sidebar). The distribution of MOSPF's group-membership-LSAs works as follows. Group membership is summarized to the

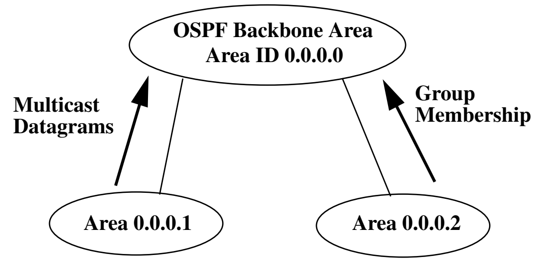

Figure 4: In the MOSPF area model, both group membership and multicast datagrams are propagated to the backbone area. It is the job of the backbone area, which is the only area with complete membership information, to determine whether, and if so, where, the multicast datagrams should be further forwarded.

backbone area, and hence the backbone has complete knowledge of all areas' group membership. However, group membership information is not readvertised back into the non-backbone areas. In other words, a non-backbone area is ignorant of all other areas' group membership. This reduces the size of the area's link-state database. To compensate, the concept of *wild-card multicast receiver* is introduced. Such routers automatically receive all multicast datagrams, regardless of destination. To enable delivery of multicast datagrams across area boundaries, all MOSPF routers connecting non-backbone areas to the backbone advertise themselves as wild-card multicast receivers to their non-backbone areas. The resulting path of multicast datagrams is pictured in Figure 4.

Unfortunately there is no concept of "scope" inherent in IP multicast addresses. For this reason, even if group membership in a particular group is completely contained within a single non-backbone OSPF area, datagrams originating within the area and addressed to that group will unnecessarily be delivered to the area's boundary with the OSPF backbone (where they will then be promptly discarded).

When forwarding datagrams across area boundaries, sometimes the exact details of the source's immediate neighborhood are unknown. In this case, the source's neighborhood is approximated by the summary-LSAs which advertise the source.

The calculation of datagram paths that cross Autonomous System boundaries proceeds analogously. When the datagram source belongs to another Autonomous System, its neighborhood is approximated by AS-external-LSAs. In addition, routers wishing to forward the datagrams to other Autonomous System advertise themselves as wild-card multicast receivers.

4. MOSPF Experiences

As of the writing of this article there is only one released MOSPF implementation (developed by Proteon, Inc.), although several others are in development. The Proteon implementation was released for general use in April 1992. It is a full MOSPF implementation, with the exception of TOS-based multicast routing.

The multicast applications included with the Proteon MOSPF implementation include: a multicast pinger (a common IP diagnostic tool), interactive commands so that the router itself can join and leave multicast groups (and so respond to multicast pings), and the ability to send network management alerts to an IP multicast address. Any of the Internet's IGMP-based multicast applications, such as the voice conferencing program "vat", can be used with the MOSPF implementation. Proteon is also using IP multicast to support certain commercial applications.

4.a. MOSPF in the MBONE

The worldwide Internet has been providing an IP multicast routing service for some time now, through a structure called the *MBONE* (for Multicast Backbone; see [11]). The MBONE is a collection of UNIX workstations running a routing daemon called "mrouted", which is an implementation of another multicast routing protocol called the

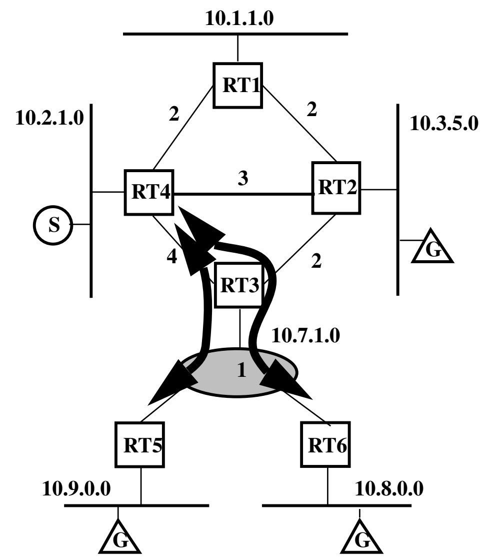

Figure 5: The tunnels that are necessary when RT3 is not-multicast capable are indicated by the dark arrows in the above diagram. These two tunnels result in a doubling of the amount of multicast traffic traversing the network segment 10.7.1.0 (shaded diagonally).

Distance Vector Multicast Routing Protocol (*DVMRP*; see [5]). Unlike MOSPF's link state technology base, DVMRP is based on the distance-vector technology that forms the basis for such routing protocols as IP's Routing Information Protocol (RIP; see [10]).

The MBONE allows applications throughout the Internet to send and receive multicast traffic. Using the MBONE, conference sessions from the IETF and other Internet technical meetings are being seen and heard live around throughout the world. For example, at a recent IETF meeting in the U.S., people from Australia were able to see and listen to the technical talks given in the plenary session, and at then end of the talk were even able to ask the speaker questions.

The MBONE is layered on top of the Internet's unicast topology; connection between the MBONE's UNIX workstations is provided by the same Internet links that are used for the usual applications such as file transfer. As multicast datagrams are forwarded from one MBONE router to another, they are often "tunneled" (i.e., encapsulated within unicast datagrams), so that they can get through the majority of the Internet's router's which are not multicast capable.

However, tunneling is often inefficient, resulting in multiple copies of a multicast datagram transiting a single network segment. Take for example the internetwork in Figure 1, and suppose that Router RT3 is not multicast capable. In order to deliver multicast datagrams from Source S to the Group members on network segments 10.9.0.0 and 10.8.0.0, tunnels would have to be configured as in Figure 5. However, this results in two copies of every datagram being delivered over the transit network segment 10.7.1.0.

The need for tunnels, and hence the inefficiency, is removed by introducing multicast routing capability into the Internet's routers. Towards this goal, the Proteon MOSPF implementation also includes a DVMRP implementation and the "glue logic" allowing routing information to be passed between the two multicast routing protocols.

Not only does this enable more of the Internet's routers to route multicasts without tunneling, but it also allows Autonomous Systems running MOSPF to become part of the MBONE. Using MOSPF instead of DVMRP in parts of the Internet has the following advantages:

- Increased stability. MOSPF's link state technology converges faster and is less prone to looping during the convergence time than DVMRP, which suffers some of the same problems that the Internet's RIP protocol does when faced with topological changes in a redundant topology.
- Using MOSPF in parts of the Internet allows aggregation of sources before the are advertised into DVMRP, keeping the size of the worldwide DVMRP tables smaller.
- Both MOSPF and DVMRP do "pruning", which means to restrict the extent of multicast datagrams to those parts of an internetwork which actually have active group members. However, unlike MOSPF which knows where all the group members are, DVMRP must occasionally probe for their location by flooding datagrams throughout the whole internetwork. This means that DVMRP pruning is not quite as effective as MOSPF's.
- Since a DVMRP router does not know exactly where all the group members are, unlike MOSPF, DVMRP cannot optimize IP multicast's "expanding ring search" behavior (see Section 2.a).

4.b. Other Applications

In addition to the applications mentioned in the Introduction (conferencing, distributed simulations, etc.), multicast has other interesting uses. For example, multicast can be used to implement the encapsulation of other protocol stacks over TCP/IP. As the TCP/IP installed base continues to grow, this has become a common thing to do: using a TCP/IP backbone to extend other network technologies such as Appletalk, IPX and NETBIOS. The routing and service discovery mechanisms of these other network technologies natively use broadcast/multicast on LANs, making their use of TCP/IP multicast when transported over a TCP/IP internetwork a natural extension.

As a concrete example, consider extending an IBM source route bridging domain over a TCP/IP internet. To do so, there would be so-called *brouters* (boxes that are both bridges and routers) on the edge of a TCP/IP internet. These brouters would be acting as IBM source routing bridges on their local token ring segments, and as MOSPF routers over their Wide Area Network (WAN) connections (e.g., over collections of synchronous serial lines, an X.25 PDN or a Frame relay network). In order to tie together the geographically separate token ring segments, the brouters treat the entire TCP/IP WAN domain as if it were a single additional token ring segment. Using MOSPF, this can be done simply and efficiently.

Client PCs in an IBM bridging environment use special packets called *explorer packets* to locate appropriate servers. These explorers must be forwarded throughout the source routing bridging domain. When using TCP/IP to extend the bridging domain, these explorers must be sent to all other participating brouters connected to the TCP/IP internet. Using MOSPF, instead of a separate packet being sent to each brouter, a single packet can be sent to a well-known multicast address. This accomplishes two things. First, it reduces the amount of WAN traffic. Second, it greatly reduces the amount of required configuration, as the brouters can automatically discover each other. After the server's location is discovered, packets traveling between the client and the server carry Routing Information Fields (RIFs) specifying the exact sequence of token ring segments that the packet must traverse. When the RIF indicates that the packet must traverse the TCP/IP internet, multicast can again be gainfully employed, this time to an IP multicast address that is algorithmically derived from the destination token ring segment's bridging address.

Such a system has been deployed using the Proteon MOSPF implementation. In this case, MOSPF provides more than reduced WAN traffic and reduced configuration. It also shields applications from WAN link outages, because of MOSPF's automatic rerouting properties. There is one other unexpected advantage. IBM applications are typically intolerant of packet reordering. As a result, most encapsulation of source routing bridging in a TCP/IP internet is done within the reliable TCP transport. However, multicast in MOSPF proves to be quite resistant to packet reordering -- so resistant that, in many situations, the overhead involved in TCP transport can be avoided altogether.

5. Protocol Analysis

MOSPF uses the Dijkstra algorithm to calculate the path of a multicast datagram through any given OSPF area. This calculation encompasses all the transit nodes (routers and transit networks) in the area; its cost scales as *O(n/log n)* where *n* is the number of transit nodes (same as the cost of the OSPF unicast intra-area routing calculation). This is the cost of a single path calculation; however, MOSPF calculates a separate path for each [source network, multicast destination, TOS] tuple. This is potentially a lot of Dijkstra calculations.

MOSPF deals with this calculation burden by calculating datagram paths in an "on-demand" fashion. That is, the path is calculated only when receiving the first datagram in a stream. After the calculation, the results are cached for use by later matching datagrams. This on-demand calculation alleviates the cost of the routing calculations in two ways:

1. It spreads the routing calculations out over time.
2. The router does fewer calculations, since it does not even calculate the paths of datagrams whose path will not even touch the router.

Cache entries need never be timed out, although they are removed on topological changes.

The effectiveness of this on-demand calculation will need to be proven over time, as multicast usage and traffic patterns become more evident.

As a simple example, suppose an OSPF area consists of 200 routers. Suppose each router represents a site, and each site is participating simultaneously with three other local sites (inside the area) in a video conference. This gives 200/4 = 50 groups, and 200 separate datagram trees. Assuming each datagram tree goes through every router (which probably won't be true), each router will be doing 200 Dijkstras initially (and on internal topology changes). The time to run a 200 node Dijkstra on a 10 mips processor was estimated to be 15 milliseconds in [8]. So if all 200 Dijkstras need to be done at once, it will take 3 seconds total on a 10 mips processor[4]. In contrast, assuming each video stream is 64Kb/sec, the routers will constantly forward 12Mb/sec of application data (the cost of this soon dwarfing the cost of the Dijkstras).

In this example there are also 200 group-membership-LSAs in the area; since each group membership-LSA is around 64 bytes, this adds 64*200 = 12K bytes to the OSPF link-state database.

Other considerations when evaluating the cost of MOSPF's routing calculation are:

- Assuming that the guidelines of [8] are followed and areas are limited to 200 nodes, the cost of the Dijkstra will not grow unbounded, but will instead be capped at the Dijkstra for 200 nodes. One need then only concern oneself with the number of Dijkstras, which is determined by the number of [datagram source, multicast destination] combinations.
- A datagram whose destination has no group members in the Autonomous System can still be forwarded through the MOSPF system. However, the Dijkstra calculation here depends only on the [datagram source, TOS], since the datagram will be forwarded along to wild-card receivers only. Since the number of group members in a 200 router area is probably also bounded, the possibility of unbounded calculation growth lies in the number of possible datagram sources. (However, note that some future multicast applications, such as distributed computing, may generate a large number of short-lived multicast groups).
- By collapsing routing information before importing it into the OSPF area and/or Autonomous System, the number of sources can be reduced dramatically. In particular, if the Autonomous System relies on a default external route, most external sources will be covered by a single Dijkstra calculation (the one using 0.0.0.0 as the source).

One other factor to be considered in MOSPF scaling is how often cache entries need to be recalculated, as a result of a network topology change. It turns out that, the further away the topology change happens from the calculating router, the fewer cache entries need to be recalculated. For example, if an external route changes, many fewer cache entries need to be purged as compared to a change in a MOSPF Autonomous System's internal link. For scaling purposes, this is exactly the desired behavior. Note that [8] complements this when it shows that changes in external routes (on the order of once a minute for the networks surveyed) are much more frequent than internal changes (between 15 and 60 minutes for the networks surveyed).

Acknowledgments

The MOSPF protocol was designed in the MOSPF Working Group of the Internet Engineering Task Force. The protocol is documented in [1]. It is based on the work of Steve Deering in [3].

4.This certainly argues for a multi-processor multicast router, with some processors devoted to the forwarding of multicast datagrams, and others working on the routing calculations. In such a system, some streams can still be forwarded while the new paths for other streams are being calculated.

References

[1] Moy, J., *Multicast Extensions to OSPF*, Internet Draft, March 1993.

[2] Moy, J., *OSPF Version 2*, RFC 1247, July 1991.

[3] Deering, S., *Multicast Routing in Internetworks and Extended LANs*, SIGCOMM Summer 1988 Proceedings, August 1988.

[4] Deering, S., *Multicast routing in a datagram internetwork*, Stanford Technical Report STAN-CS-92-1415, Department of Computer Science, Stanford University, December 1991.

[5] Waitzman, D., Partridge, C. and Deering, S., *Distance Vector Multicast Routing Protocol*, RFC 1175, November 1988.

[6] Deering, S., *Host extensions for IP multicasting*, RFC 1112, May 1988.

[7] Bharath-Kumar, K. and Jaffe, J. M., *Routing to multiple destinations in Computer Networks*, IEEE Transactions on Communications, COM-31[3], March 1983.

[8] Moy, J., ed., *OSPF protocol analysis*, RFC 1245, July 1991.

[9] Sedgewick, R., *Algorithms*, Addison Wesley, ISBN 0-201-06672-6, August 1984.

[10] Hedrick, C., *Routing Information Protocol*, RFC 1058, June 1988.

[11] Eriksson, H., *MBone - the Multicast Backbone*, CACM/INET93.

Author Information

Mr. Moy is a Senior Staff Engineer at Proteon, Inc., where he designs networking protocols and software for routers and bridges. He received an M.S. in Mathematics from the University of Minnesota, and an M.A. in Mathematics from Princeton University. He chairs the IETF's OSPF and MOSPF Working Groups, and is a Pioneer Member of the Internet Society.