Incorporation of Flow & QOS Control in Multicast Routing Architectures *

K. Ravindran && D. Loguinov

Department of Computer Science
City University of New York (City College)
Convent Avenue at 138th Street, New York, NY 10031 (USA)
Contact e-mail address: *ravi@cs-mail.engr.ccny.cuny.edu*

Abstract

Many multicast routing protocols consider only the unicast shortest paths between every pair of source and destination in a group (often based on the number of hops) to setup a multicast tree, such as DVMRP and CBT. Furthermore, they do not offer the ability to consider the resource savings possible in the presence of multiple data flows sharing common paths in a tree, such as CBT and PIM. These aspects hamper resource-efficient routing. In this light, our paper describes flow-based resource allocation control in multicast tree setups. Our proposed model has two aspects: i) exploring alternate paths to connect a joining user to a tree, should the unicast shortest path does not have sufficient bandwidth to sustain the new data flow; and ii) downstream resource allocation to merge a new data flow with the on-going flows in a tree. (i) requires 'multi-path connectivity' information at routers in the network, while (ii) requires flow aggregation information to be exchanged between on-tree routers. The model is evaluated by simulation studies. The proposed model can be useful in 'Integrated Service' networks that contain both low and high bandwidth links, such as Future Internet.

* Part of this work is conducted under the **Information Institute** partnership program between US Air Force Rome Laboratory and CUNY.

1 Introduction

Multicast routing architectures provide the network capability for multi-destination data delivery, as required by multimedia distributed applications (e.g., broadcast TV, multi-user video conferencing). They employ tree-structured communication channels, realized by switching nodes and inter-node links, for point-to-multipoint forwarding of data from a source to a set of destinations [1]. In addition, some architectures also provide for transport level sharing of the underlying network paths, whereby the data units from multiple sources are multiplexed at one or more intermediate nodes to flow towards destinations over common downstream path segments. In Figure 1, the data from sources s_1 and s_2 both flow over the tree rooted at node 4. A routing architecture determines how multicast paths are set up to connect sources and destinations in an application and how various path segments are shared across the multi-source data flows.

Each path segment of a multicast tree (i.e., hop) connects a distinct pair of adjacent nodes, and is realized by a 'native connection' set up over the intervening link. This 'native connection' employs an inter-node data exchange protocol, as supported by the backbone network (e.g., IP link). Besides packet level routing of data (often based on an address carried in packets), a 'native connection' also needs to allocate resources, viz., node buffers and link bandwidth, to transport data packets. The extent of resource allocations is based on the flow characteristics of data, such as data rates (e.g., 64 *kbps* for audio and 2 *mbps* for compressed video). Thus a multicast path embodies:

- *Logical connectivity* between sources and destinations

  This is prescribed by routing table entries at each node in the path which determine out-going network link interfaces based on the group address carried by an incoming packet.

- *Flow-based resource allocations* at on-tree routers

  This manifests as resource allocation control at the network link interfaces of each node in the path that allows receiving of incoming packet streams and sending of outgoing packet streams at the prescribed level of flow & QOS.

These two aspects are interwoven with one another in a complex manner. However, current routing protocols have focused primarily on the logical connectivity aspects, as elaborated below.

Routing protocols such as DVMRP and CBT [2, 3] determine logical connectivity based on the source-destination placement in physical topology of the network. For instance, the join of a destination to a DVMRP tree takes place along the unicast shortest path ('shortest' in the number of hops) to the source. Likewise, the join of a source to a CBT channel is along the unicast shortest path to the 'core' node. Routers maintain these shortest

0-8186-9014-3/98 $10.00 © 1998 IEEE

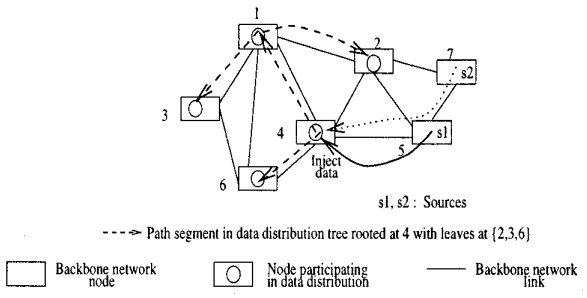

Figure 1: Tree-structured channel for multicasting

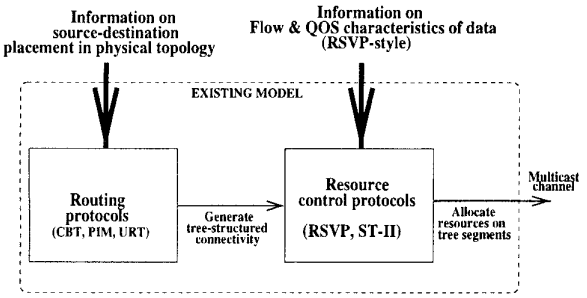

Figure 2: Separating resource reservation and routing

paths based on physical connectivity in the network. The underlying premise in these protocols is that a flow-based resource allocation occurs *after* the routing protocol sets up the tree. See Figure 2.

Non-consideration of resource reservations by a routing protocol limits its knowledge as to how good a path matches the flow characteristics of data to be carried. In DVMRP for instance, the availability of only less-than-10 *mbps* capacity links along the unicast 'shortest' path from a video receiver to the source-rooted tree of a 10 *mbps* video broadcast station will cause a join of this receiver to fail[1], when in fact alternate paths to the tree with the required 10 *mbps* capacity may exist (but not shown in the unicast 'shortest' path tables). The problem arises because a reservation protocol can consider only the paths selected by the routing protocol. The problem becomes significant when the network contains both high bandwidth and low bandwidth subnets.

Furthermore, where the routing protocol allows the sharing of a link by multiple data flows (as in CBT and PIM), the extent of sharing can reduce the per-flow communication cost on this link: due to amortization of the overhead of maintaining a single 'native connection', and possibly, 'statistical multiplexing' of bursty flows over the 'connection' bandwidth. Since how many flows share a link

[1] DVMRP sets up a path connecting the receiver to the tree. It is a subsequent resource resevation on this path that fails.

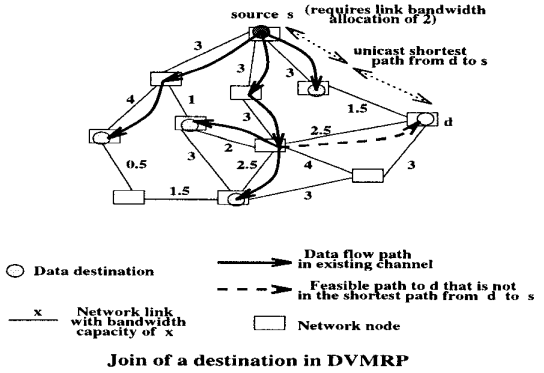

Figure 3: Why flow-based path setup: A scenario

is itself determined by the source-destination placement in physical topology, the effect of this topological parameter on flow-based routing may be inseparable. Referring to Figure 1, the data of s_1 and s_2 share resources allocated in the tree rooted at node 4, but not resources in the path between nodes 5 and 4. By shifting the root of the shared tree to node 5, the overall cost of data distribution can be reduced. With cost reduction due to path sharing across multiple flows, it is possible that there exists paths with more number of hops but with less total cost.

In this paper, we study how routing decisions may be influenced by resource allocations, for path setups. In the earlier scenario, the DVMRP join protocol may be extended to allow examination of alternate paths to the source-rooted tree, whereupon the video receiver can successfully connect through any of these paths. See Figure 3 for illustration of this scenario. Where the routing architecture also allows path sharing across multiple data flows, a user join or leave can affect the number of flows sharing a link, and hence the per-flow resource allocations on this link. Thus, with a network-wide control on resource allocations in various path segments of a channel, resource-efficient paths can be set up and/or the likelihood of successful path setups for user joins can be increased.

The goal of this paper is to identify the global control activities required for path setups. These are:

- Exploring alternate paths to connect a joining user to a tree, should the unicast shortest path does not have sufficient bandwidth to sustain the new data flow;
- Downstream resource allocations from a node to merge a new data flow with on-going data flows in the distribution tree rooted at this node;

These control mechanisms are prelude to the formulation of routing protocols with resource allocations integrated therein. A reservation style (RSVP or ST-II [4, 5]) is then cast into the protocol activities by a routing algorithm.

The paper embarks on studying the impact of this model in routing protocols (such as CBT, PIM, and

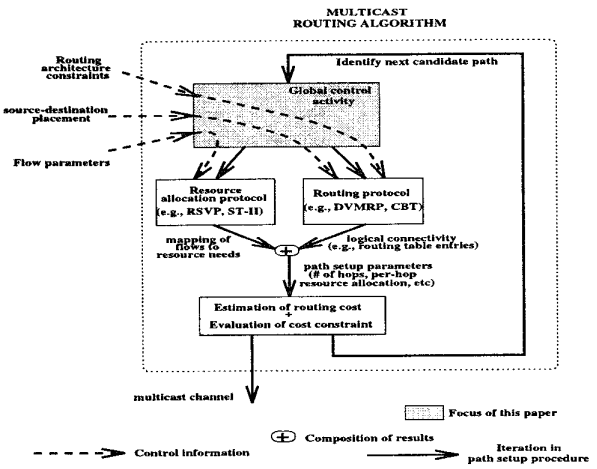

Figure 4: Functional components of multicast routing

DVMRP), in terms of path selection constraints. In PIM 'sparse mode' for instance, the constraint, namely, the data from a source should flow through a designated 'rendezvous-point' node in order to reach a given set of destinations, may force choosing a data path that does not necessarily have the minimum number of hops. The paper analyzes this aspect by casting the model onto a cost-based multicast routing algorithm. Figure 4 illustrates how these functions interact.

We believe that our approach will be useful in evolving the 'integrated services architecture' for the Internet.

2 Network Support for Multicasting

In this section, we present a canonical view of multicast support elements in the network that forms the basis for our proposed protocol extensions.

2.1 QOS-capable multicast channels

The physical topology of the backbone network consists of a set of nodes $\mathcal{V}$, interconnected with one another through a set of communication links $\mathcal{E}$. Users, viz., data sources and destinations, may reside in distinct nodes $\mathcal{U} \subseteq \mathcal{V}$, and form the communication end-points in an application. A user may have only send capability, only receive capability, or both send and receive capability. In an example of conference, source is the initiator of a conversation and destinations are other participants in the conference browsing this conversation. User-level flow requirements may be specified in the form of *data rate* of sources and *acceptable data delays* at destinations.

A multicast channel supports multipoint communication among $\mathcal{U}$ through a set of network nodes $\widehat{\mathcal{U}}$ and links $\widehat{\mathcal{E}}$, where $\mathcal{U} \subseteq \widehat{\mathcal{U}} \subseteq \mathcal{V}$ and $\widehat{\mathcal{E}} \subseteq \mathcal{E}$. The nodes $\widehat{\mathcal{U}}$ implement routing functions, with the interconnections $\widehat{\mathcal{E}}$ realized by transport functions over the links. Data from a source made available at node $u \in \widehat{\mathcal{U}}$ flows over a *distribution tree* rooted at u, and arrives at leaf nodes containing destinations (refer to Figure 1). The tree represents a multipoint path, with each node receiving a data along its incoming link and replicating this data over its outgoing links. With multiple sources and destinations in an application, the channel is a topological superposition of all the trees carrying data from sources to destinations.

A 'native connection' $e(x, y) \in \widehat{\mathcal{E}}$ constitutes a distinct hop in a data path, realized by a network-specific protocol over the backbone link between adjacent nodes x and y (e.g., 'IP link' between routers). It embodies:

- Resource allocation to sustain the required level of data flow between x and y, viz., buffers in x and y and the bandwidth of link between x and y;
- 'connection' management state in x and y (e.g., routing table entries, refreshing of 'soft state').

A routing algorithm may take into account the amount of resources allocated for data flow between x and y, and[2] the 'connection' management overhead, in setting up the path segment $e(x, y)$.

2.2 Path sharing by multi-source flows

A network link in a multicast channel may carry data injected from one or more sources. The flow of data from multiple sources over a common link may take place over either a single instance of 'native connection', or separate instances of 'native connection', one for each source, set up by the backbone network using its internal transport protocol. The former case depicts the sharing of a path segment across data flows, and the latter case depicts non-shared path segments.

Suppose data from s_1 and s_2 flow at the rate of q_1 and q_2 respectively. In Figure 5-(a), the combined data flow over a single 'connection' set up over the link between nodes x and y. The two data flows share this 'connection', with the network allocating resources to support the aggregate flow $q_1 + q_2$ (also denoted as $+\{q_1, q_2\}$) over the link. In Figure 5-(b), the data from s_1 and s_2 flow over distinct 'connections' that support the data rates of q_1 and q_2 separately. The sharing of a path segment by a set of data flows allows the 'connection' management overhead to be amortized across these flows. Furthermore, when q_1 and q_2 are bursty, the bandwidth needs of $(q_1 + q_2)$ can be less than the sum of the individual needs of q_1 and q_2, if the traffic control policy on a shared 'connection' employs 'statistical multiplexing' of bursty data over the reserved

[2] Where necessary, the backbone network may be augmented to provide 'on-demand resource reservation' capability. For instance, the 'best-effort delivery' model of current Internet is being augmented in the form of multiple service classes parameterizable with delay tolerance and bandwidth specifications [6].

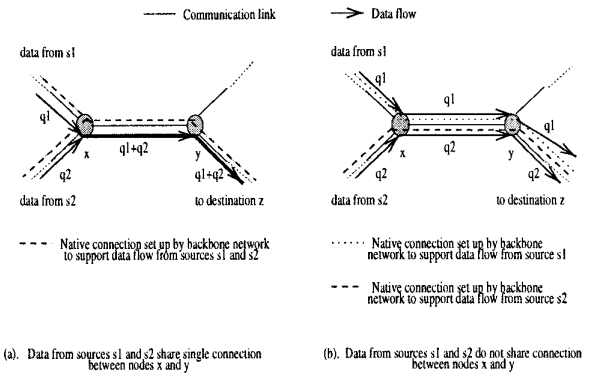

Figure 5: Multi-source flows over 'native connections'

bandwidth. Thus, a reduction in the cost of a flow s can be achieved by increasing the number of flows sharing the path segment with s. From an architectural perspective, 'path sharing' means that when two or more streams get combined at a node, they are routed through common path segments all the way towards destinations.

We now view resource allocations from a routing perspective.

3 Resource allocation and routing

Routers should implement a globally quantifiable abstraction of resources, i.e., a given set of flow parameters should result in the same estimate of resource needs by every router participating in a path setup. Such an implementation allows a model of network-wide activities for path setup that can take into account resource allocations.

3.1 Data flow and transport cost

The amount of resources allocated to support a data transfer (viz., node buffers, link bandwidth) mainly depend on the data flow rates. Assume a functional relation to map data rates to resource needs at a router node:

$$\mathcal{F} : q \rightarrow \mathcal{B} \qquad \text{for } \mathcal{B} \in \mathcal{R}^+ \text{ (i.e., positive real numbers)},$$

where $\mathcal{F}$ maps a data rate q to resource needs $\mathcal{B}$. In a simple form, q may be given by a tuple (average_rate,peak_rate,loss_tolerance). The mapping is such that $\mathcal{F}(q') > \mathcal{F}(q'')|_{q'>q''}$ — see [6] for a definition of the relation '$>$' on q. $\mathcal{F}$ encapsulates a resource allocation policy (e.g., allocating 10% additional bandwidth relative to the specified flow rate, in a conservative scheme). Besides resource allocations, the network also incurs an overhead for maintaining 'native connections' over a link (e.g., routing table entries, lease rental for 'connect time' over link).

The cost of data transport over a link can be determined from the resource allocations and the 'connection' overhead. A simple formulation of cost may be:

$$\text{cost of data flow } q \text{ over hop } p = fc(p) + \mathcal{F}(q), \qquad (1)$$

where fc indicates the fixed cost (we omit normalization constants, for brevity). Using such a formulation, the network-wide cost of data transport from sources to destinations can be estimated by a routing algorithm, by summing up the cost of each hop in the data paths.

An exact form of the relationship between transport costs and resource allocations is beyond the scope of our paper. Without loss of generality, we employ the cost formulation (1) in our study of routing control activities.

3.2 Resource implications of path sharing

The resource allocation in a given path segment of a shared tree need to meet the flow requirements of the aggregated data of various sources $s_1, s_2, \cdots, s_N|_{N\geq 2}$. The allocation satisfies 'weak additivity', indicated as: $\mathcal{F}(q_1 + q_2 + \cdots + q_N) \leq \mathcal{F}(q_1) + \mathcal{F}(q_2) + \cdots + \mathcal{F}(q_N)$. The 'weak additivity' captures the possible savings due to 'statistical sharing' of resources across various flows over a path segment. $\mathcal{F}$ allows routers to estimate resource allocations when sources connect to a channel (and resource deallocations when users disconnect).

Consider a multimedia conferencing among 5 participants involving 2 *mb/sec* video and 64 *kb/sec* audio data. The network interface of a participant workstation should allocate a bandwidth of 8.256 *mb/sec* for incoming flows and 2.064 *mb/sec* for outgoing flows. The bandwidth allocations may be lower if 'statistical multiplexing' of flows is taken into account. Our experimental studies over ATM networks indicate that $\mathcal{F}(+\{q_1, \cdots, q_4\}) \geq 6.5$ *mbps* at the input interface, with a 'burst factor' of 0.5 and loss tolerance of 5% for each stream.

It may be noted that even without path sharing, the network may decide on some form of 'statistical multiplexing' of the data streams flowing over a link. An example is the multiplexing of 'cells' of different 'virtual circuits' over 'virtual path' links in ATM networks, based on network-assigned 'bandwidth classes'. So the question is what additional traffic-related information become available to a router that implements path sharing, to enable better routing decisions. This aspect is discussed below.

3.3 link-sharing and 'connection'-sharing

Consider the link-level resource allocation decisions at a router. When, say, two data streams q_1 and q_2 flow over distinct 'native connections' set up over a link, the data rates of q_1 and q_2 need to be supported with separate resource allocations, i.e., $\mathcal{F}(q_1)$ and $\mathcal{F}(q_2)$ respectively. This is the case with non-shared tree routing protocols, such as DVMRP and MOSPF [2, 7].

The non-sharability of path segments even though set up over a common link arises from a lack of knowledge on the end-to-end merging of component flows q_1 and q_2 in the application, and hence, that their downstream links towards each destination are the same. This in turn limits the ability of the routing algorithm to set up a network-wide path with less amount of resource allocations. In particular, the network cannot determine the criteria for 'statistical multiplexing' of q_1 and q_2 over the link with respect to application-wide flow parameters, such as the traffic correlation between audio+video streams in a 'floor-controlled' conference session. So the savings in resource allocation, which amounts to $\mathcal{F}(q_1) + \mathcal{F}(q_2) - \mathcal{F}(q_1 + q_2)$, may be less with non-shared path segments.

Thus 'connection sharing' enables better resource allocation decisions due to the casting of network parameters with end-to-end flow characteristics of data, in comparison to 'link sharing' where a decision is based solely on network internal parameters.

3.4 Multi-path connectivity of nodes

With flow aggregation in multicast trees, the path of a data flow with the minimum number of hops to a 'flow merging point' x may not always be the path of choice. This may be due to a longer path for the flow offering higher degree of 'path sharing' with other flows, resulting in less per-hop resource consumptions that more than offsets the additional hops. Also, non-availability of resources and/or non-guarantee of delay constraints for a flow in the 'shortest path' to x (say, due to congestion) may allow a user join to succeed in other paths. Accordingly, the routing protocol should employ mechanisms to explore more than one path to x, as allowed by physical connectivity in the network. These mechanisms include using a 'multi-path connectivity table' (MPT) at each router and 'message flooding' through the network [8][3].

Separate studies are required to determine how practical it will be to maintain MPTs or resort to message flooding, in large networks (such as the Internet). The bottom-line is that some form of 'multi-path exploration' is required, though we do not know at present — and it is beyond the scope of this paper as to —what is the best way of achieving it.

In the next section, we examine the currently available routing architectures, in light of the need to incorporate flow-based resource allocation control.

[3] The MPT at a router R is a 'distance table' containing one entry for every other router. The entry for a router R' may list one or more outgoing links of R to reach R' and the 'distance' of R' along these links. Alternately, the list may be prescribed in terms of a probability of using the various links in setting up a path from R to R'.

4 Multicast routing architectures

Current routing protocols that generate multicast channels focus on efficiently maintaining logical connectivity between sources and destinations. For instance, protocols that allow path sharing across multiple data flows (such as CBT and PIM) were motivated towards reducing the per-flow 'connection' overheads. We identify the canonical architectural mechanisms embodied in the routing protocols, and then study how flow-based resource allocation control can be integrated into these mechanisms.

The data streams from various sources meet each other and get aggregated at one or more intermediate nodes, on their way towards destinations. The choice of intermediate nodes, as prescribed by a multicast architecture, determines the level of path sharing achievable across streams. We illustrate this aspect in 3 different architectures, using a sample source-destination configuration on a backbone network: sources s_1, s_2 sending data flows q_1, q_2 and destinations d_x, d_y, d_z receiving the combined data flow q_1+q_2. Figure 6 illustrates the architectural differences.

4.1 'single source-rooted' trees (SST)

In this architecture, a separate tree is rooted at each source that connects to all destinations. With multi-source data flows, the trees cannot share their path segments even when set up over common links. In Figure 6-(a), the trees rooted at s_1 and s_2 support q_1 and q_2 respectively over separate 'connections' to d_x, d_y, d_z. A multipoint path (MPP) is a superposition of such non-sharable trees. The MPP then consists of distinct resource allocations on each of the trees — $\mathcal{F}(q_1)$ and $\mathcal{F}(q_2)$ in the above scenario.

The DVMRP, ST-II and MOSPF and PIM-'dense mode' protocols [2, 5, 7, 9] employ the SST architecture.

4.2 'single fixed center' tree (SFT)

In this architecture, a tree is rooted at a special 'core' node, connecting all destinations. Data from a source is brought to an on-tree node through a point-to-point path, and is then combined with the data already flowing in the tree towards destinations. In Figure 6-(b), the paths from s_1 and s_2 to the 'core' support q_1 and q_2 respectively, with the combined flow $q_1 + q_2$ distributed to d_x, d_y, d_z over the tree rooted at the 'core'. Thus a MPP consists of a shared tree rooted at the 'core' and non-sharable point-to-point paths set up from sources to the tree. It involves distinct resource allocations on point-to-point paths and a shared allocation on the core-rooted tree — $\mathcal{F}(q_1)$ and $\mathcal{F}(q_2)$, and $\mathcal{F}(+\{q_1, q_2\})$ respectively in the above scenario.

The earlier version of 'core-based tree' (CBT) protocol [3] employs the SFT architecture.

4.3 'multiple fixed center' tree (MFT)

In this architecture, one or more 'rendezvous point' (RP) nodes are designated, each of which roots a tree

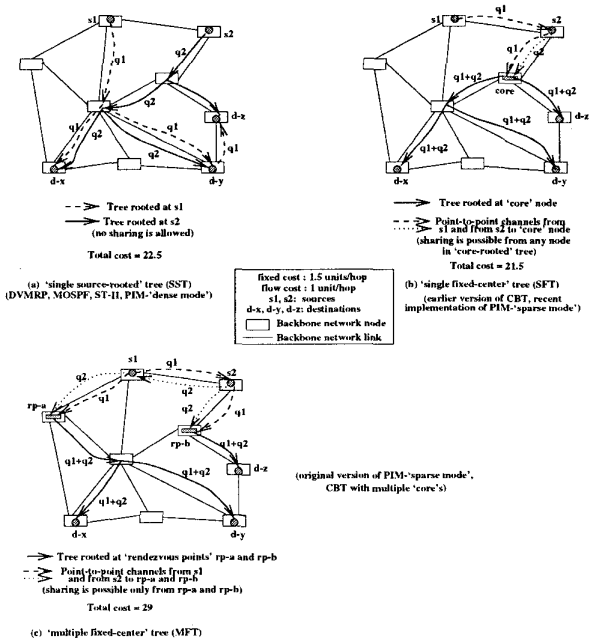

Figure 6: Channels in SST, SFT, MFT architectures

connecting a disjoint subset of destinations. Data from sources are brought to each of the RPs through point-to-point paths, and then the combined data flow over each RP-tree towards its connected destinations. In Figure 6-(c), the nodes rp_a and rp_b serve as RPs. The paths from s_1 and s_2 to rp_a support q_1 and q_2 respectively, with the combined flow $q_1 + q_2$ distributed over the tree rooted at rp_a connecting to d_x, d_y (similarly for q_1 and q_2 flowing to rp_b and then to d_z). Thus a MPP consists of shared trees rooted at various RPs and non-sharable point-to-point paths set up from each source to these RPs. It involves distinct resource allocations on point-to-point paths and a shared allocation on RP trees — $\mathcal{F}(q_1)$ and $\mathcal{F}(q_2)$, and $\mathcal{F}(+\{q_1, q_2\})$ respectively in the above scenario.

The PIM-'sparse mode' protocol [9] and the recent 'multi-core' CBT [10] employ MFT architecture.

Thus the multicast channels in SST, SFT and MFT architectures differ in how the root of a distribution tree connecting to destinations is placed and the extent of sharing among point-to-point paths connecting sources to the tree.

5 Global control activities

A global control activity is a set of protocol actions taken at various routers as part of a network-wide path setup decision. In this section, we describe the activities occurring during the join of users.

We assume that a user join results in the creation of additional path segments to the existing MPP. In other words, the routing algorithm implements 'incremental configuration changes' (INC). Though it is feasible to conceive of 'global configuration changes' where each join causes a reconstruction of the entire MPP, we believe that in large distribution sessions (such as video broadcast over the Internet), only incremental changes will be practical.

5.1 Downstream resource allocation

The join of an entity to a channel can cause a protocol action potentially at every node in the underlying 'data distribution tree' (DDT). Consider, for instance, the join of an entity U at a node x in the DDT — x is the 'data access point' (DAP). Suppose U is a data source. The 'resource check' needs to be carried out at all nodes in the subtree that is rooted at x and has leaves at destinations. This is to determine resource availability at various nodes in the tree, to the extent of $(\mathcal{F}(+\{q\}_x + q') - \mathcal{F}(+\{q\}_x))$, for supporting the flow q' from U, where $\{q\}_x$ is the set of data streams currently flowing over the tree rooted at x. This underscores the 'weak additivity' of resource allocations over a shared path, as in SFT and MFT architectures. In SST architecture though, the 'resource check' is for $\mathcal{F}(q')$, since there is no path sharing. Along point-to-point path connecting U to x however, the resource checks are the same, viz., for $\mathcal{F}(q')$.

As illustration, Figure 7 shows DAP nodes x_1 and x_2 where a source U considers joining. Here, the subtrees for 'resource check' activity of U are rooted at x_1 and at x_2. The 'resource checks' take into account the current flow q over those hops where it merges with q'.

Suppose U is a data destination. The DAP x can be any node in the DDT. For a path connecting U to x, resource availability to the extent of $\mathcal{F}(+\{q\}_x)$ is checked at each hop.

The routing architecture constrains the choice as to which nodes may be candidate DAPs for a joining source. In SFT, any node in the 'core-rooted' tree part of the MPP can serve as a DAP, but nodes in the point-to-point paths from sources to the 'core' node cannot. In MFT, only a 'rendezvous-point' node can serve as DAP. For a joining destination, SFT allows any node in the DDT to serve as DAP. In MFT, the candidate DAPs chosen may be on any of the RP-trees.

The 'resource check' activity may be carried out by the 'path' messages when RSVP [4] is employed as the protocol (likewise, ST-II defines a message type for this purpose)[4].

[4]Though RSVP emphasizes 'receiver-oriented' resource reservations whereby a receiver may 'explicitly select' the sources it wishes to receive from, 'wild card' receivers that wish to receive from all connected sources can still be supported by a downstream resource allocation upon a new source join.

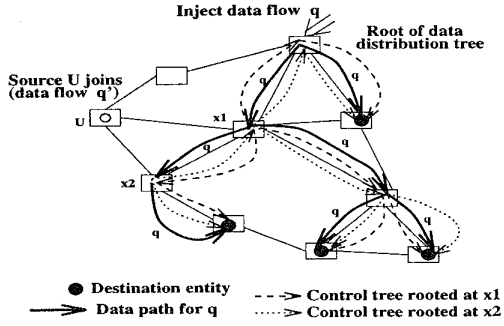

Figure 7: Trees for flow of control activities

5.2 Architecture constraints on routing

As indicated in section 3.4, we assume a physical topology maintenance mechanism that realizes multi-path connectivity between nodes. For our purpose, such a mechanism identifies one or more candidate paths through which a joining entity U can reach a given DAP. With the possibility of multiple candidate DAPs $\{x\}$ — x_1 and x_2 in Figure 7, the routing control activity finally selects a path for connecting U to the MPP. The selection may be based on the cost of various hops in the point-to-point paths connecting U to various $\{x\}$, the fixed cost of various hops, and whether resources are available in these hops.

When U is a source and the MPP is set up under SFT or MFT architectures, the cost of injecting the data flow of U at each of the DAPs $\{x\}$, in the presence of shared downstream resource allocations with the on-going flows is also taken into account. Under SST architecture though, the cost is for connecting all the destinations $\{d\}$ to U through a separate tree.

A 'minimum cost'-based path setup with multi-path exploration may be represented as (c.f. relation (1)):

$$\overset{\min}{\forall x} \; ((\sum_{\forall e \in p_{(\mathrm{U},x)}} [fc(e) + \mathcal{F}(\mathrm{U})] + \sum_{\forall e' \in \mathcal{T}_{(x,\{d\})}} [\mathcal{F}(+\{q\}_x + \mathrm{U}) - \mathcal{F}(+\{q\}_x)]) \; \forall p_{(\mathrm{U},x)} \in \mathrm{MPT}(\mathrm{U},x))$$

when U is a source (under SFT and MFT)

$$(\overset{\min}{\forall x} \; ((\sum_{\forall e \in p_{(d,x)}} [fc(e) + \mathcal{F}(\mathrm{U})] \; \forall p_{(d,x)} \in \mathrm{MPT}(d,x)) \;) \; \forall d$$

when U is a source (under SST)

$$\overset{\min}{\forall x} \; (\sum_{\forall e \in p_{(\mathrm{U},x)}} [fc(e) + \mathcal{F}(+\{q\}_x)] \; \forall p_{(\mathrm{U},x)} \in \mathrm{MPT}(\mathrm{U},x))$$

when U is a destination (under SFT and MFT)

$$\forall s \; [\overset{\min}{\forall x} \; (\sum_{\forall e \in p_{(\mathrm{U},x)}} [fc(e) + \mathcal{F}(s)] \; \forall p_{(\mathrm{U},x)} \in \mathrm{MPT}(\mathrm{U},x))$$

for $x \in \mathcal{T}_{(s,\{d\})}$] when U is a destination (under SST),

where $\mathcal{T}_{(x/s,\{d\})}$ is a tree rooted at node x/s and connecting to destinations $\{d\}$ — note that $\mathcal{T}_{(x/s,\{d\})} \subseteq$ MPP, $p_{(\mathrm{U},x)}$ is a path connecting nodes U and x, and MPT(U, x) is a procedure that generates a list of paths to connect nodes U and x. In an underlying protocol, the response to 'resource check' messages from DDT leaf nodes may carry the fc and flow aggregation information at various hops in the tree. These information allow (2) to be evaluated.

When a DAP is finally chosen, resource allocations are committed at the chosen paths[5].

We now present a simulation study of INC algorithms on various architectures.

6 Simulation study

We use 'random graph' technique to generate a physical topology involving a large number of nodes (25-100 nodes) and containing a source-destination configuration. The simulation parameters are: i) number of nodes in physical topology, ii) placement of sources and destinations, iii) average number of links per node (i.e., node degree), and iv) flow rate of sources. Each network link is assigned a finite bandwidth capacity. The flow rates and link bandwidths are given as normalized numbers, in the range [1,10].

6.1 Simulation procedures

The sequence of steps in the simulation procedure are:

1. Generate physical topology of network along with fixed costs of links (same cost for each link);
2. Designate the placement of source entities — in distinct nodes — and their flow rates;
3. Select a source-destination configuration with a given number of destination entities (placed in distinct nodes);
4. Run INC algorithm for each of the architectures (SST, SFT and MFT) with the selected source-destination configuration — 2 RPs are used in MFT;
5. Repeat step 4 for various source-destination configurations, and estimate the cost in each case.

Note that the flow rates do not impact the simulation results, because we analyze the flow of control activity in the tree, but not the effect of 'data flow parameters' that the activity instantiates at various routers.

The simulation program is run for a topology containing 25 nodes. The simulated application configuration has 5 sources, and the number of destinations is varied between 1 and 15. We examined 300 placements of sources and destinations in the physical topology (20 different placements for a given number of destinations).

[5] The path selection constraint given by (2) may be modified to incorporate end-to-end delays, wherein the sum of per-hop delays in a path should be less than the acceptable delay prescribed for a given data flow.

6.2 Performance index

When a user joins a channel, the ensuing control activity causes a protocol action at each hop in the channel. For instance, a distributed protocol executed by the activity may probe each downstream node, to determine if enough resources are available for the user, and then commit the actual reservation at these nodes. So a meaningful performance measure is the number of network-wide actions taken to effect a resource allocation activity.

In a normalized form, the flow of control in the activity along each network link may be considered as causing 1 unit of protocol action at the concerned router. Given this, a relevant index of interest is the number of distinct hops created in the MPP for various source-destination configurations. This index, which represents the 'channel size' (SIZE), is the sum of the number of hops in each non-shared path and the number of hops in the shared path of the channel. Since resource allocation occurs in each path segment, SIZE may indicate network-wide overhead in making the resource allocation (such as number of messages required to commit a reservation). Note that SIZE does not include the overhead for multi-path exploration[6].

For a given user join in an application configuration, SIZE depends on the source-destination placement in physical topology and the architectural constraints on path selection. For instance, an increase in the geographical spread-out of a configuration, say, by placing destinations far apart from one another, or, an increase in the path length without increasing the geographical spread-out, may cause an increase in SIZE. This observation can also be corroborated from [11].

Thus, how SIZE varies with respect to changes in source-destination configurations can provide meaningful measures of performance. To determine SIZE, we average the total number of hops created for a given set of destinations across their various placements in physical topology.

From the plot of SIZE versus number of destinations (Figure 8), we observe that as the number of destinations increases, SIZE also increases[7]. This is because additional hops need to be created to connect a joining destination.

For reasonably sized configurations (number of destinations > 5), we observe that SIZE in SST increases the fastest among all architectures with respect to the number of destinations. This is due to the inability to employ path sharing among various data flows in SST.

[6] Though our global control activity assumes some form of multi-path connectivity between nodes to locate DAPs, it does rely on a specific protocol to achieve this (such as the use of MPT-based search or resorting to 'message flooding'). So results on the overhead for multi-path exploration are less meaningful for our paper. We expect, however, that the overhead will much depend on the type of search protocol employed.

[7] The plot shows the results for a 'multiple dynamic-center' tree (MDT) architecture also, where every node can serve as a DAP [12]. For brevity, this paper does not describe MDT.

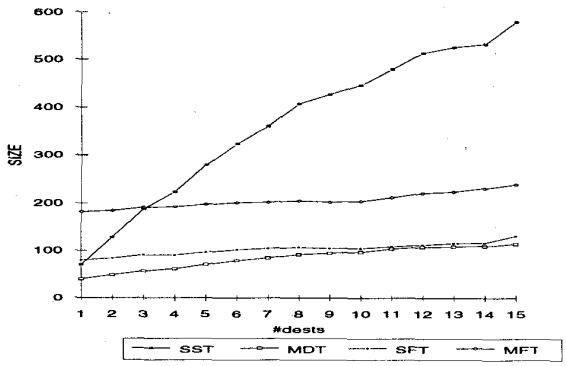

Figure 8: INC algorithm on a 25-node topology

6.3 Comparison of SFT and SST

We expect that the reduction in SIZE for SFT over SST will be somewhat more as the size of physical topology increases. The reason is as follows. For a given placement of destinations in physical topology, let $ch_{s,N}$ be the average size of channel configuration across all possible placements of sources $s_1, s_2, \ldots, s_N|_{N \geq 2}$. The variation of $ch_{s,N}$ with respect to N may be given as:

- $\mathcal{O}(N^{c_1})$ in SST architecture, where $0 < c_1 \leq 1.0$, depicting that each SST channel has at least one hop, and, in the extreme, all SST channels have the same number of hops;
- $\mathcal{O}((\frac{N}{R})^{c_2})$ in SFT architecture, where R is the network-wide average number of flows sharing a link in the core-rooted tree ($1 \leq R < N$), and $c_1 \leq c_2 \leq 1.0$, depicting possible longer paths to destinations than with separate SSTs.

Thus, the ratio of the number of paths to be created to the number of alternate paths that can be explored is often lower with SFT architecture than with SST architecture (i.e., the INC algorithm has more 'freedom' in creating paths in SFT architecture). Intuitively, as the physical topology size increases, the possible number of alternate paths to be explored becomes higher, and hence it is more likely for INC algorithm to find lower length paths in SFT architecture than in SST architecture.

6.4 Comparison of MFT and SFT

SIZE is less for SFT than MFT. The reduction becomes somewhat less as the number of destinations increases. These observations may be explained as follows:

- In SFT architecture, all destinations are served by a single 'core'-rooted tree, while in MFT architecture,

each RP tree serves a disjoint subset of the destinations. For a given placement of sources in physical topology, let $ch_{d,M}$ be the average size of 'core'-rooted tree across all possible placements of destinations $d_1, \ldots, d_M|_{M \geq 1}$. The variation of $ch_{d,M}$ with respect to M may be given as $\mathcal{O}(M^{c_3})$ in SFT, where $0.0 \leq c_3 \leq 1.0$, depicting an out-degree of the core-rooted tree in the range $[1, M]$. The number of hops in a RP-tree is then $\mathcal{O}((\frac{M}{k})^{c_3})$, where k is the number of RPs (≥ 1), assuming that each RP serves an equal number of destinations on an average (for large M).

- The number of hops in the point-to-point paths from sources to RPs is $\mathcal{O}(k.N^{c_4})$ in MFT architecture and that to the 'core' is $\mathcal{O}(N^{c_5})$ in SFT architecture, where $0 < c_5 \leq 1.0$, depicting that point-to-point paths have at least 1 hop each, and in the extreme, have equal number of hops, and $c_5 \leq c_4 \leq 1.0$, depicting the 'short-cutting' of paths at on-tree routers in SFT. So the contribution to SIZE from point-to-point paths will be less in SFT than in MFT.

When M is small, the length of point-to-point paths dominates the size of tree(s). As M increases, the size of tree(s) becomes significant. So the difference in SIZE between the SFT and MFT architectures becomes somewhat less.

Overall, the goal of this paper is to provide a methodology for resource allocation control in multicast architectures rather than on analyzing the specifics of routing protocols. The explanation of simulation results in terms of architecture-specifics should be viewed in this context.

7 Conclusions

The paper described a model of resource allocation control for multicast path setups. It is based on tree-structured channels in the network to which sources and destinations can connect to exchange data. The data flow requirements include widely varying data rates and large volume data transfers across users, and multi-source broadcasting of data to a set of destinations (e.g., broadcast video, multimedia conferencing).

The model maps the user-specifiable flow parameters to allocation of resources, viz., node buffers and link bandwidths, in various path segments of the channel. Based on the mapping, the paper prescribes the elements of a global control activity to incorporate flow-based resource allocations in routing architectures.

A main contribution of the paper is the *integration* of routing and resource allocation functions, whereby the setting up of a tree is itself directly influenced by flow & QOS parameters. The existing implementation proposals for RSVP (and ST-II) provide for resource reservations after a tree is set up. To support our approach, we cast the routing subsystems with resource allocation mechanisms.

To illustrate the usefulness of our model, we studied the control activities for setting up data paths in shared-tree based and 'single-source rooted' tree architectures. To aid the study, we chose a 'incremental configuration' algorithm, which is cast with architecture-specific constraints to set up data paths.

A simulation study provides 'proof-of-concept' for the model. Since the routing architectures are employed in many multicast protocols (viz., DVMRP, CBT, ST-II, MOSPF, PIM, URT, [2, 3, 5, 7, 9, 12]), in one form or the other, we believe that the results of our analysis will be useful for incorporating resource allocation in these protocols. This will particularly be useful for large networks, such as Future Internet.

Acknowledgement

The authors acknowledge **Mr. Ting-Jian Gong** of Clear Communications Corporation (Lincolnshire, IL) for providing the simulation results reported in this work.

References

[1] S. E. Deering and D. R. Cheriton. **Multicast Routing in Datagram Internetworks and Extended LANs.** In *ACM Transactions on Computer Systems*, Vol.8, No.2, pp.85-110, May 1990.

[2] D. Waitzman, C. Patridge, and S. Deering. **Distance Vector Multicast Routing Protocol** In RFC 1075, Nov. 1988.

[3] T. Ballardie, P. Francis and J. Crowcroft. **Core Based Trees (CBT): An Architecture for Scalable Inter-Domain Multicast Routing.** In Proc. *Comm. Architectures, Protocols and Applications*, ACM SIGCOMM, pp.85-95, Sept. 1993.

[4] L. Zhang, S. E. Deering, D. Estrin, S. Shenker, D. Zappala. **RSVP: A New Resource ReSerVation Protocol** *IEEE Network*, pp.8-18, Sept. 1993.

[5] C. Topolcic. **Experimental Internet Stream Protocol, version 2 (ST-II).** In *RFC 1190*, CIP Working Group, Oct. 1990.

[6] S. Shenker. **Fundamental Design Issues for the Future Internet.** In *IEEE JSAC*, Special Issue on *Global Internets*, 13(7), pp.1176-1188, Sept. 1995.

[7] J. Moy. **Multicast Extension to OSPF.** In *Internet Draft*, Sept. 1992.

[8] D. Bertsekas and R. Gallager. **Routing in Data Networks.** Chapter 5 in *Data Networks*, Prentice Hall Publ. Co., 1992.

[9] S. Deering, D. Estrin, D. Farinacci and V. Jacobson. **An Architecture for Wide-Area Multicast Routing.** In Proc. *Comm. Architectures, Protocols and Applications*, ACM SIGCOMM, 1994.

[10] Y. C. Chang, Z. Y. Shae, and M. H. W. LeMair. **Multiparty Video Conferencing using IP Multicast.** In *IS & E / SPIE Symp. on Electronic Imaging: Science and Technology*, 1996.

[11] J. Kadirire. **Minimizing Packet Copies in Multicasting by Exploiting Geographic Spread.** In *Computer Communication Review*, ACM SIGCOMM, vol.24, no.3, pp.47-62, July 1994.

[12] K. Ravindran. **A Flexible Network Architecture for Data Multicasting in High Speed 'Multiservice Networks'** . In *IEEE JSAC*, Special Issue on *Global Internets*, Oct. 1995.