1. Consider a router that uses a three stage buffered network with 1024 ports and 32 port switch elements. Assume that the system uses dynamic routing that has embedded routing tables for multicast sessions. Suppose we want to configure the system so that each output port can participate in up to 10,000 multicast sessions and that the average fanout per session is 4. What is the maximum number of multicast sessions that the system must support?

The number of multicast sessions can be as large as 10,000×1024/4=2.56 million.

How many bytes of memory does a middle stage SE require for its multicast routing table? 4 bytes for each of 2.56 million entries. So, 10 million bytes.

Suppose the system is implemented using eight parallel planes. How much memory do all the middle stage SEs require? How much is this on a per multicast session basis?

There are 32 middle stage SEs per plane, so we need a total of 32×8×10 million or 2.56 billion bytes altogether. That’s 1000 bytes per multicast session.

Suppose you want to add a new endpoint to a multicast session and that the new endpoint is connected to a third stage switch that none of the other endpoints is connected to. How many table entries need to be changed to add the new endpoint?

A table entry needs to modified in each of the 32×8=256 SEs in the middle stages. In the third stage, we need to modify 8. This gives a total of 264.

2. In the binary tree multicast architecture, when adding endpoint to a multicast session, we need to pick a “relay” point to forward packets through when extending the tree. Assuming a system in which all ports can serve as relay points, how would you go about choosing the best relay point for a given multicast session. Explain your choice.

One approach is to use the port that is currently recycling the least amount of multicast traffic. This seems like the best choice in networks where the session bandwidth is highly variable, since it minimizes the potential for congestion at the relay point. It’s also easy to implement using a heap to allow us to quickly find the port with the smallest amount of multicast traffic.

In systems where multicast session bandwidths vary little or not at all, there is another approach one can take. Here, it makes more sense to select the port that has just enough available bandwidth to accommodate the multicast session’s traffic. By doing this, we pack multicast sessions into a smaller number of ports, reducing the fragmentation of the recycling bandwidth that can otherwise occur. This can reduce the likelihood of blocking a high bandwidth multicast session.

3. When dropping an endpoint from multicast binary tree, we eliminate a relay point reducing the number of passes through the network that a packet must take to reach one of the multicast output ports. This creates an opportunity for packets to get out of order that is not
necessarily handled by the normal resequencing mechanism. Describe how you would handle this problem.

One possible solution is to timestamp multicast packets/cells the first time they are sent through the network, but leave the timestamp unchanged on subsequent passes. The resequencer could then use a resequencing delay that is large enough to handle the maximum number of passes through network, rather than just one pass. This is conceptually simple enough but could significantly increase the delay that many packets/cells experience, and will certainly increase the memory required in the resequencer. The delay problem could be minimized by applying a different delay for unicast and multicast packets, but the increased memory requirement would still apply.

Another approach is to have the port that is “downstream” of the relay point being eliminated, temporarily hold packets received on the “new path”. This can be implemented by having the upstream port send two special packets. The first is a “path-switch” packet which it sends along the new path. The other is a “last-packet-on-path” packet, which it sends along the old path. On getting the path-switch packet, the downstream port starts buffering packets from the upstream port and continues to do so until it gets the last-packet-on-path packet. It then allows the packets from the upstream port to proceed to the outgoing queue. To make this robust, the receiver would have to release the packets after a timeout, since it’s possible for the last-packet-on-path packet to get lost. Note that if the path-switch packet is lost, this mechanism will get packets out of order, but will otherwise continue to operate. If packet losses are infrequent enough, it’s probably ok to ignore these resequencing errors.

4. This question concerns the binary tree multicast architecture. Assume a system with 4,000 ports and a speedup \( S = 1 \) in which the external links operate at 10 Gb/s links. Also, assume that up to 15% of the outgoing traffic can be multicast and that the maximum multicast session bandwidth is 100 Mb/s. If this system is configured using dedicated recycling ports, approximately how many recycling ports are needed to make it nonblocking?

Here \( n = 4000, \delta = .15, \) and \( B = .01, \) so the number of recycling ports needed is \( \frac{600}{.15+1-.01}, \) which is approximately 530.

Suppose there was a multicast session with a fanout of 2000 and a bandwidth of 100 Mb/s. How much load would this session place on the recycling ports (in total)?

A session with a fanout of \( F \) passes through recycling ports \( F - 2 \) times, so the given session would place a load of about 200 Gb/s on the recycling ports.

What is the average load it places on a single recycling port?

\( \frac{200 \text{ Gb/s}}{530} \) or about 380 Mb/s.

Answer the above questions again, assuming that the maximum multicast session bandwidth is 2 Gb/s.

5. In the binary tree multicast architecture, many-to-many multicast can be implemented in one of two ways. Either you can use separate one-to-many multicast trees for each participant, or you can use a single shared tree. Consider a many-to-many multicast with 20 participants (each can both send and receive) and an output bandwidth of 100 Mb/s (that is, each output link that is participating in the multicast sees a total of 100 Mb/s, so each participating input link contributes an average of 5 Mb/s).

Using separate multicast trees, how many multicast routing table entries are needed? How many are needed with a shared tree?
With separate multicast trees, you need \(20 \times 18 = 360\) table entries. With a shared tree, you need \(20 + 19 = 39\) entries.

What is the total recycling bandwidth used with separate multicast trees? What is the recycling bandwidth with a single shared tree?

With separate trees, you need \(20 \times 17 \times 5 = 1700\) Mb/s of recycling bandwidth. With a shared tree, you need \(19 \times 100 = 1900\) Mb/s. However, this is a little misleading, because we are allocating 5 Mb/s to each sender in the case of separate trees, while the combined tree allows the senders to divide the bandwidth in arbitrary ways and to change their shares dynamically. If we were to configure the separate trees to provide this extra flexibility, we would need 20 times as much bandwidth in the case of separate trees or 34,000 Mb/s.

How many control cells are needed to modify routing table entries when adding another participant, if we use separate multicast trees? If we use a single shared tree?

With separate trees, you need to modify \(19 + 2 \times 20 = 60\) entries. With a shared tree, you need to change 3 table entries.