CS 268: IP Multicast Routing

Kevin Lai

April 22, 2001

Motivation

- Scalable multi-destination delivery
  - use same bandwidth/link to send to n receivers as 1 receiver
  - deals with flash crowds
  - e.g., video/audio conferencing, news dissemination, file updates
- Unknown destination delivery (logical addressing)
  - sender does not know receivers' location-dependent addresses
  - e.g., service location, mobility, anonymity, naming
- These functions currently served by other mechanisms/systems
  - serial duplicate unicast
  - content distribution networks
  - directory servers (LDAP, DNS)
  - why IP Multicast?

Multicast Service Model [Deering & Cheriton ’90]

- Open group
  - group identified by location-independent address
  - senders and receivers need not know about each other
  - no restriction on number or location of members
  - hosts explicitly join group
  - hosts may leave without notification
  - any source (not necessarily in the group) can multicast to all members in a group
- Packets delivery is best effort

Multicast Service Model

- Advantages:
  - efficient to implement in local area
  - logical addressing
  - allows hosts and applications to fail
- Disadvantage
  - Difficult to protect against unauthorized listeners
  - How to implement routing in the Internet?

Key Design Goals

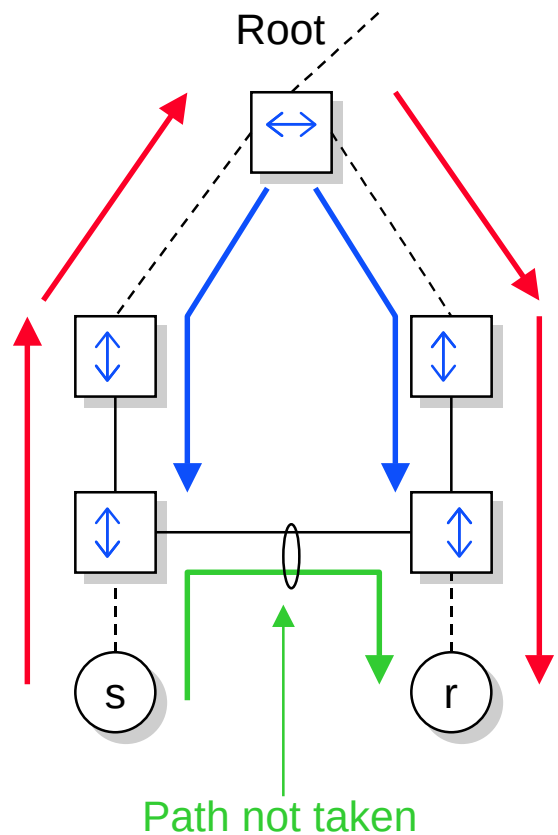
- Low packet delivery latency
- High packet delivery probability
- Low join latency
- Low leave latency

Internet Multicast Routing

- Local area
  - Single spanning-tree (SST) [DC90]
- Intra-domain
  - Distance-vector multicast (DVM) [DC90]
  - Link-state multicast (LSM) [DC90]
- Inter-domain
  - Hierarchical multicast [DC90]
  - Protocol Independent Multicast (PIM) [?]
  - Core Based Trees (CBT) [BFC93]
  - Single Source Multicast (SSM) [HC99]

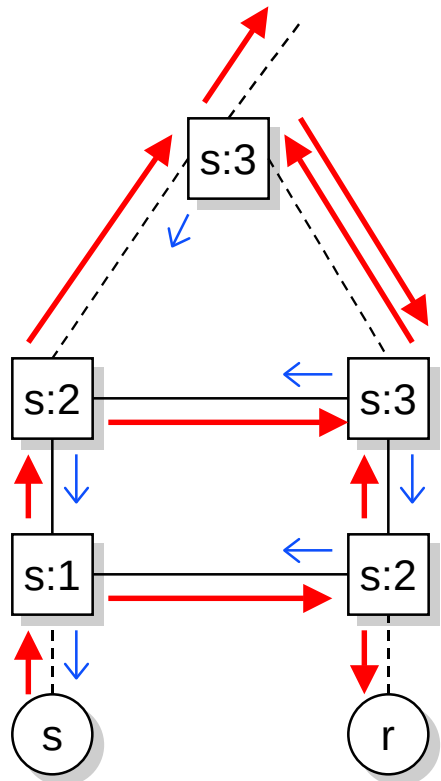
Single Spanning Tree Multicast

- Extension to single spanning tree bridging for LANs
- Bridges compute a single spanning tree
  - necessary for unicast delivery
- Join sent to all bridges
  - Leave breadcrumbs pointing back to new member
- Packet forwarding
  - forwarded towards members
  - may take high latency path
  - not likely to be significant in a LAN



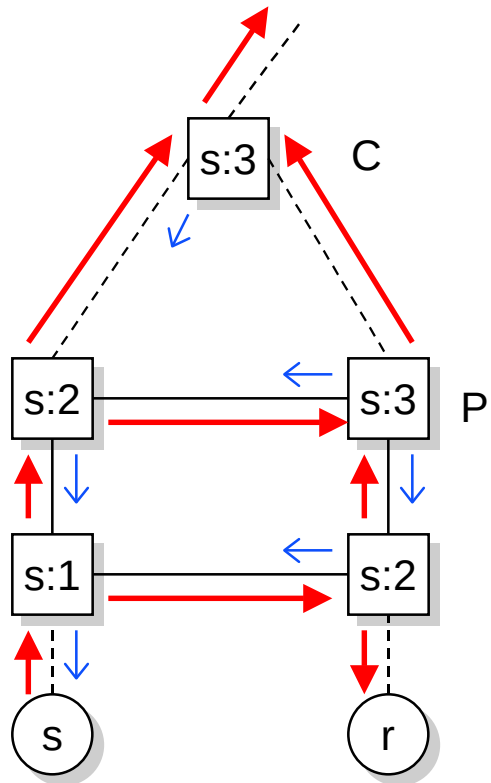
Distance Vector Multicast

- Extension to DV unicast routing
- Routers compute shortest path to each host
  - necessary for unicast delivery
- No join required
  - every link receives a copy, even if no interested hosts
- Packet forwarding
  - iff incoming link is shortest path to source
  - out all links except incoming
  - Reverse Path Flooding (RPF)
  - packets always take shortest path
    - assuming delay is symmetric
  - link may have duplicates



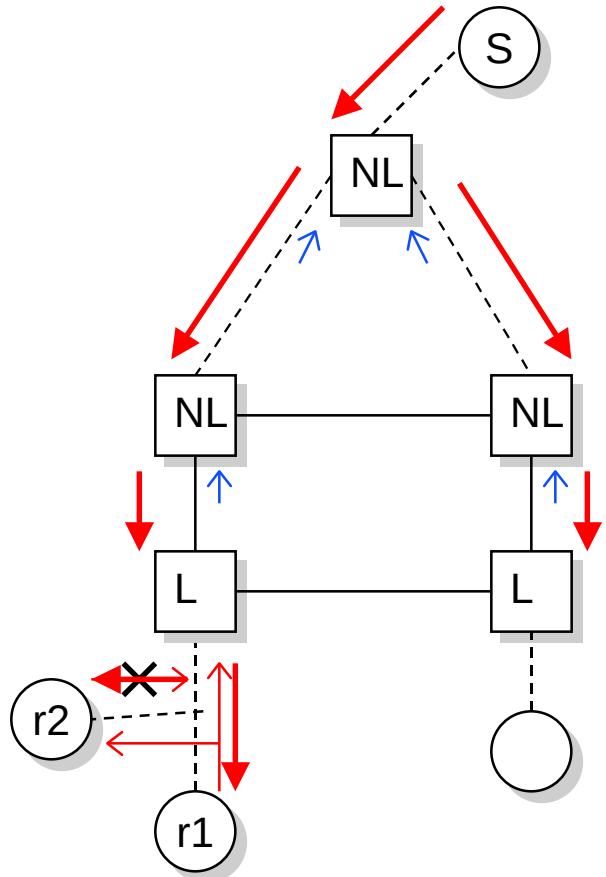
Reverse Path Broadcasting (RPB)

- Extend DV to eliminate duplicate packets
- Combine DV and spanning tree
- Choose parent router for each link
  - router with shortest path to source
  - lowest address breaks ties
  - each router can compute independently from already known information
  - each router keeps a bitmap with one bit for each of its links
- Only parent forwards onto link



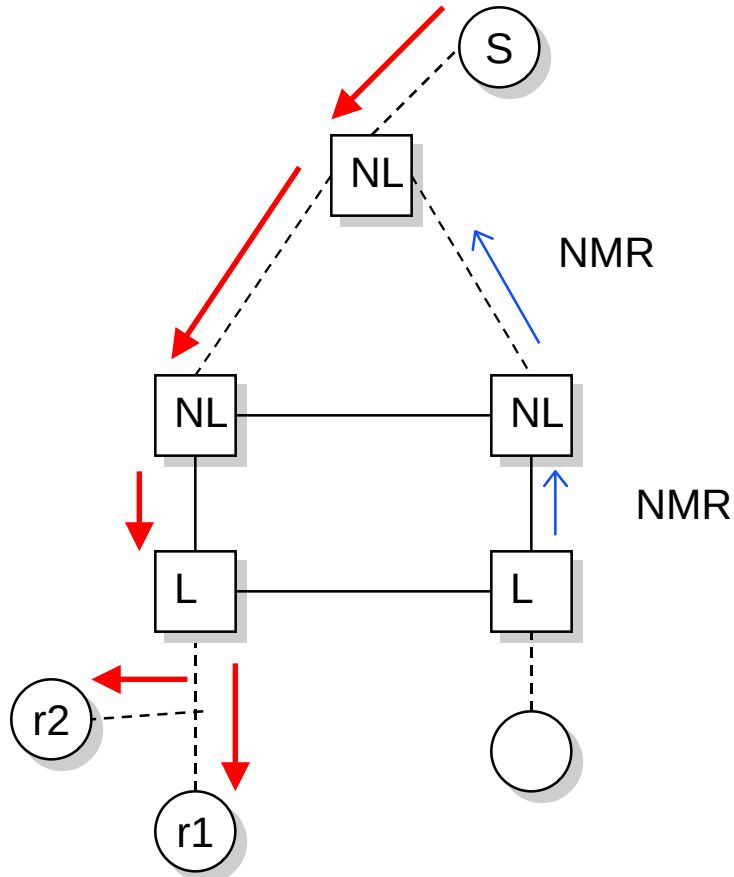
Truncated Reverse Path Broadcasting (TRPB)

- Extend DV/RPB to eliminate unneeded forwarding
- Identify leaves
  - routers announce that a link is their next link to source S
  - parent router can determine that it is not a leaf
- Explicit group joining
  - members periodically (with random offset) multicast report locally
  - hear an report, then suppress own
- Packet forwarding
  - iff not a leaf router or have members
  - out all links except incoming



Reverse Path Multicasting (RPM)

- Extend DV/TRPB
- Propagate lack of members up tree
  - no members → send Non-Membership Report (NMR) up tree
  - receive NMR → prune branch
  - on timeout, recreate branch of tree



RPM Details

- How to pick prune timers?
  - Too long → large join time
  - Too short → high control overhead
- What do you do when a member of a group (re)joins?
  - Issue prune-cancellation message (grafts)
- Both NMR and graft messages are positively acknowledged
  - recover from lost graft faster
  - prevent multiple NMR before timeout
- Why not build tree incrementally instead of building the whole thing and then pruning?
  - want to handle pervasive groups with many senders
  - many senders increases complexity
- Router state requirements:
  - O(Sources $\times$ Groups) active state

Core Based Trees (CBT)

- Ballardie, Francis, and Crowcroft, “Core Based Trees (CBT): An Architecture for Scalable Inter-Domain Multicast Routing”, SIGCOMM 93
- Similar to Deering’s Single-Spanning Tree
- Unicast packet to core and bounce it back to multicast group
- Tree construction is receiver-based
  - One tree per group
  - Only nodes on tree involved

CBT Characteristics

- Router state scales O(G) instead of O(S x G)
- Sub-optimal delay
- Single point of failure
  - Core goes out and everything lost until error recovery elects a new core
- Small, local groups with non-local core
  - Need good core selection
  - Optimal choice (computing topological center) is NP complete

Problems with IP Multicast Model [Holbrook & Cheriton ’99]

- Few groups have many senders
  - difficult to construct optimal tree for many senders
- Violates ISP input-rate-based billing model
  - No incentive for ISPs to enable multicast
- No indication of group size (again needed for billing)
- Hard to implement sender control → any node can send to the group (remember open group semantic?)
- Multicast address scarcity

Solution: EXPRESS

- Limit to single source group
  - Use a session rely approach to implement multiple source multicast trees
  - sender is like core in CBT
    - example of fatesharing
- Add a counting mechanism
  - a recursive CountQuery message
  - for billing and group size indication
- Sender controls membership
- Use both source and destination IP fields to define a group
  - Each source can allocate 16 millions channels (i.e., multicast groups)
- Use RPM algorithm

Summary

- Large amount of work on multicast routing
- Multicast still not deployed
- Economic incentives play a major role in deploying a technical solution
- Original IP Multicast model may have been too general
  - sometimes not clear initially what is the most useful semantic that can still be implemented efficiently and deployed economically