Group Communication

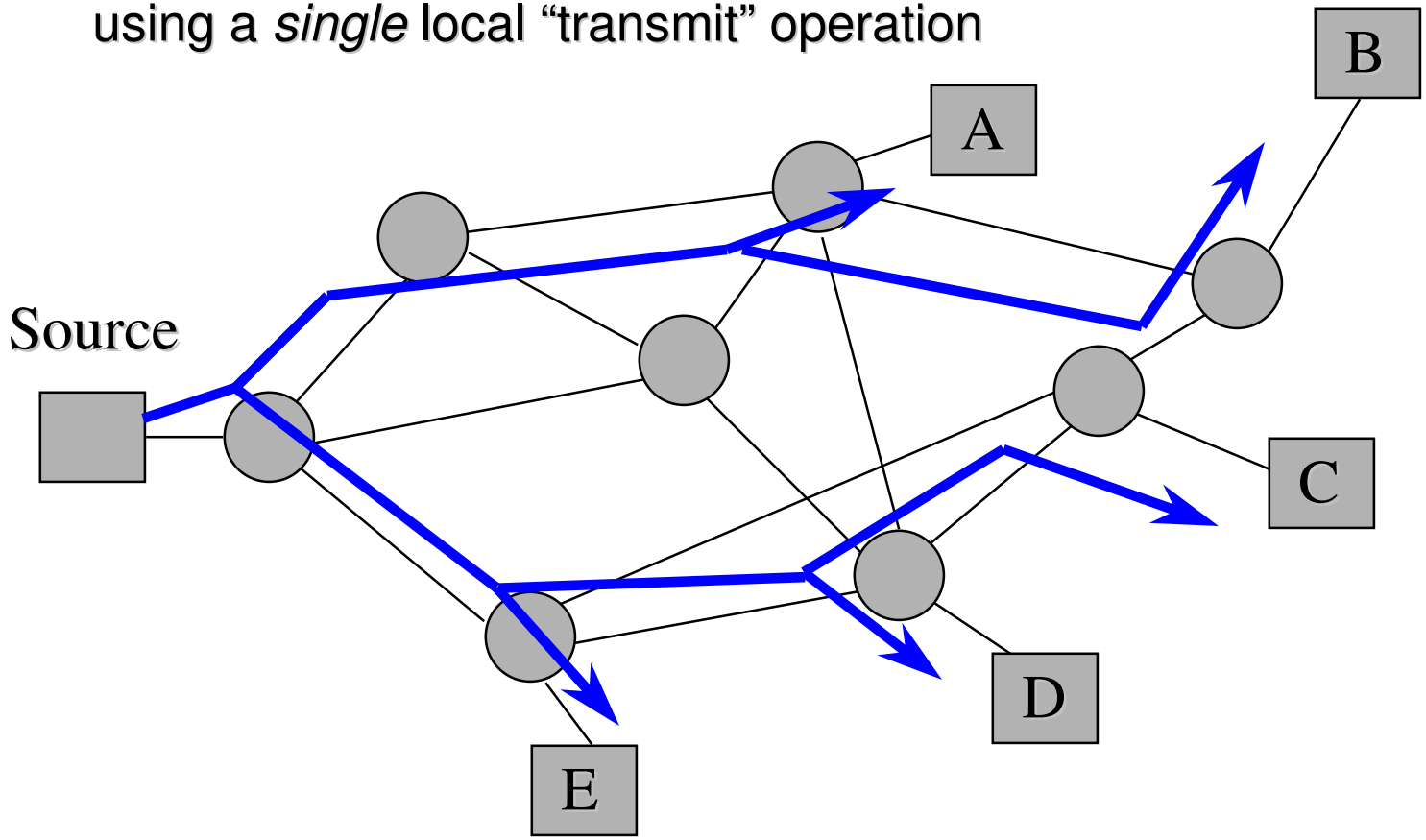
Bloc 6, INF 586

Walid Dabbous

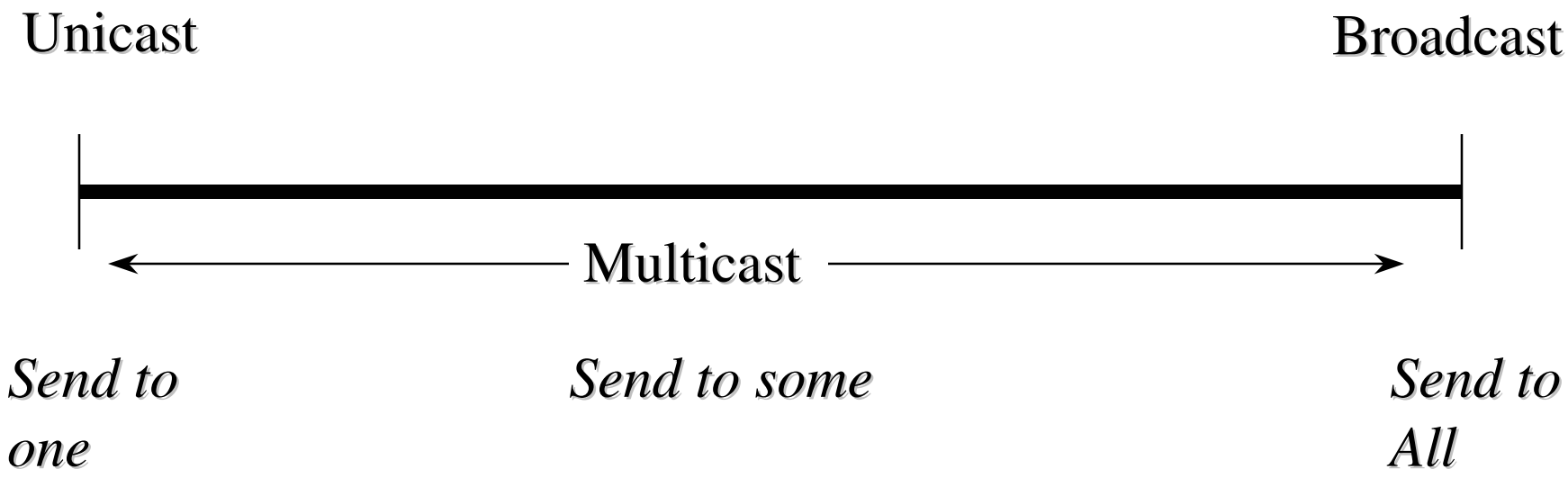
INRIA Sophia Antipolis

Definition

- **Multicast**: is the act of sending a message to multiple receivers using a *single* local "transmit" operation



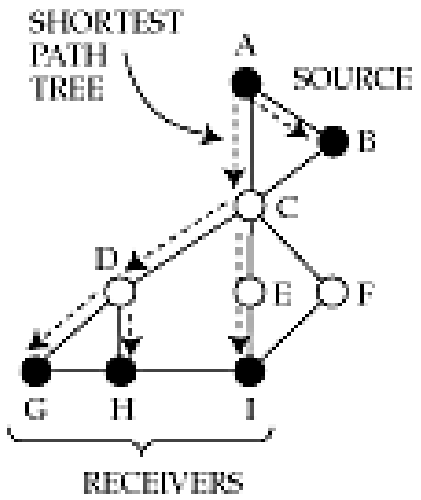
A Spectrum of Paradigms



Multicast flavors

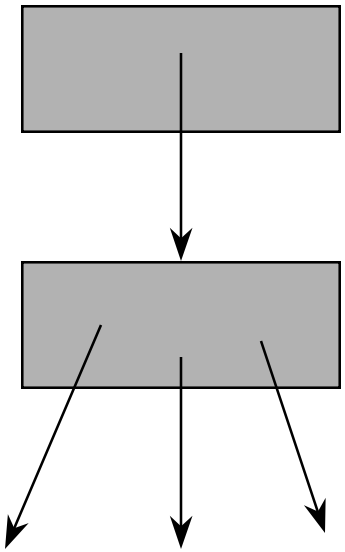
- Unicast: point to point
- Multicast:
  - point to multipoint
  - multipoint to multipoint
- Can simulate point to multipoint by a set of point to point unicasts
- Can simulate multipoint to multipoint by a set of point to multipoint multicasts
- The difference is efficiency

Multicast efficiency

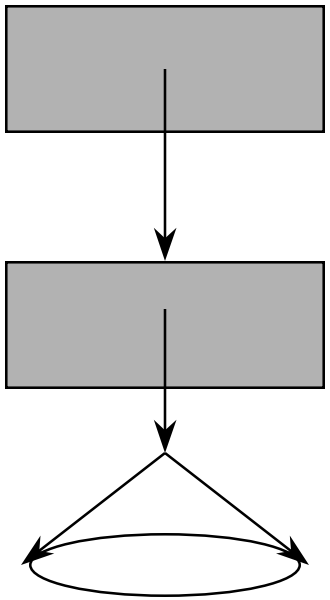


- Suppose A wants to talk to B, G, H, I, B to A, G, H, I
- With unicast, 4 messages sent from each source
  - links AC, BC carry a packet in triplicate
- With point to multipoint multicast, 1 message sent from each source
  - but requires establishment of two separate multicast “groups”
- With multipoint to multipoint multicast, 1 message sent from each source,
  - single multicast “group”

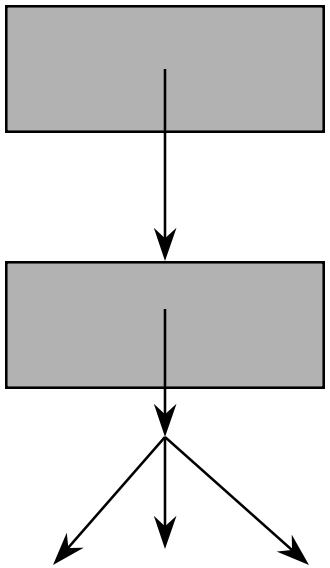
The Layering of Multicast



Multicast
by
Unicast



Multicast
by
Broadcast

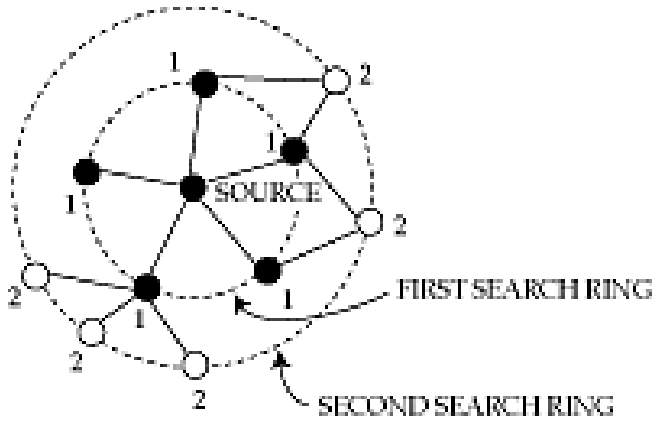


Multicast
by
Multicast

The Many Uses of Multicasting

- Teleconferencing (1-to-many) and symmetric (all to all)
- Distributed simulation (war gaming, multi-player Doom)
- Resource discovery (where's the next time server?)
- Software/File Distribution
- Video Distribution
- Network news (Usenet)
- Replicated Database Updates

Example use: expanding ring search



- A way to use multicast groups for resource discovery
- Routers decrement TTL when forwarding
- Sender sets TTL and multicasts
  - reaches all receivers <= TTL hops away
- Discovers local resources first
- Since heavily loaded servers can keep quiet, automatically distributes load

Outline

- Wide area Multicast routing
  - or the dream of « universal » communication
- Reliable multicast transport
- Multicast congestion control
- *Not covered*: enforcing reception semantics across receivers (ordering, atomicity) -- better see in “distributed computing” litterature

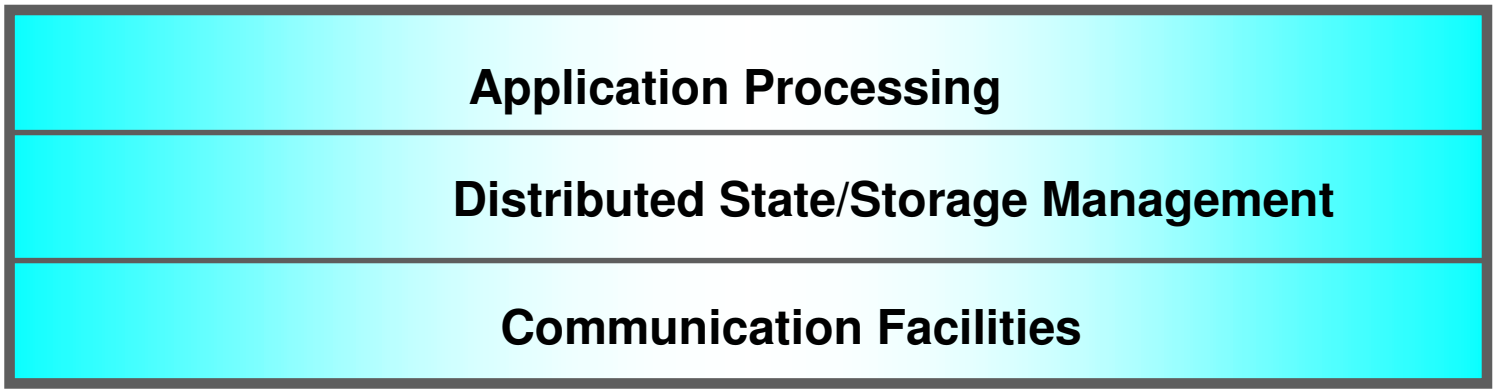
Multicast Routing

Group Communication

- What: communicate with a group of endpoints, rather than just one
  - a “natural” generalization of unicast
- basis for real-world organizations
  - groups of cooperating entities
- fantasy: group communication as the basis for “organizations of computers”
  - the “true” foundation for building distributed systems
  - it just needs to scale
- Is it easy to implement?

Distributed Systems Structure

Three primary layers



What functionality to put at each level?

Hard? multicast versus unicast

- To implement a unicast application, I can use TCP, RPC mechanisms, RMI (Remote Method Invocation), etc.
- With multicast, just basic sockets and UDP
- Where is the multicast TCP?
- Where is the multicast middleware?
- Why so limited Internet multicast deployment?

Let's look at some history ...

In the 1970's, ...

- Multi-access networks appeared
  - Ethernet, rings
- Broadcast: an accident of the technology
- We discovered uses for this "accident":
  - discovery: e.g. broadcast to locate a printer server
  - multi-point delivery: e.g. Mazewar games
- But not scalable: a single broadcast address
  - e.g. 3 Mbps experimental Ethernet broadcast address
  - Not every node involved in Mazewars, a print server

Even LAN group communication needs to be scalable

In the 1980's, ...

- 10 Mbps Ethernet:
  - 47 bits of multicast addresses
  - Lots of addresses for group communication applications

L2 group communication has become feasible

V Distributed System: early 80s

- Extended RPC-like IPC to support group operations
  - send message to “group” object; 0, 1 or more return messages
- Example uses:
    - name server group for distributed lookup
    - scheduler group to select host for remote execution
    - process manager group to act on one or more processes
    - transaction groups for distributed transactions
    - various multi-player games, distributed applications
- Experience: Average performance, fault-tolerance and ease of programming

The group model was born!

History of IP Multicast

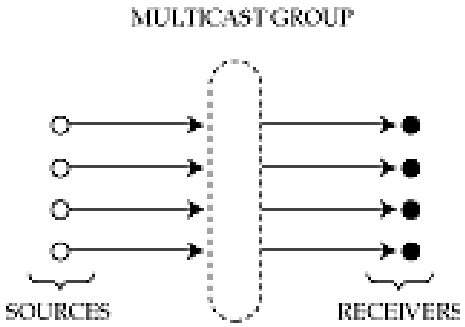
- 1983: how to scale multicast from an Ethernet to the Internet?
- IP multicast memo - april 1984
- Deering's thesis - 1991
  - host group model and IGMP
  - DVMRP - local broadcast-and-prune routing
- Focus on host group model and "local" routing
- Required underlying service
  - datagram + multicast delivery in LAN
  - (if broadcast or unicast LAN -> emulate multicast)

The host group model

Deering, 1991

- senders need not be members
- groups may be of any size
- no topological restrictions on membership
- membership dynamic and autonomous
- host groups may be transient or permanent

Multicast group



- Associates a set of senders and receivers with each other
  - but independent of them
  - created either when a sender starts sending from a group
  - or a receiver expresses interest in receiving
  - even if no one else is there!
- Sender does not need to know receivers' identities
  - *rendezvous point*

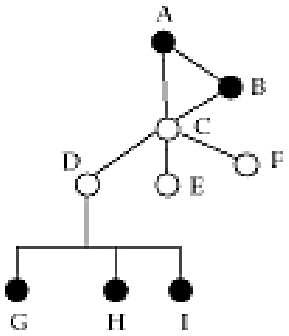
Addressing

- Multicast group in the Internet has its own Class D address
  - looks like a host address, but isn't
- Senders send to the address
- Receivers anywhere in the world request packets from that address
- "Magic" is in associating the two: *dynamic directory service*
- Four problems
  - which groups are currently active - *sdr*
  - how to express interest in joining a group - *IGMP*
  - discovering the set of receivers in a group - *Flood and prune*
  - delivering data to members of a group - *Reverse path forwarding*

Issues in wide-area multicast

Difficult because

- sources may join and leave dynamically
  - need to dynamically update shortest-path tree
- leaves of tree are often members of broadcast LAN
  - would like to exploit LAN broadcast capability



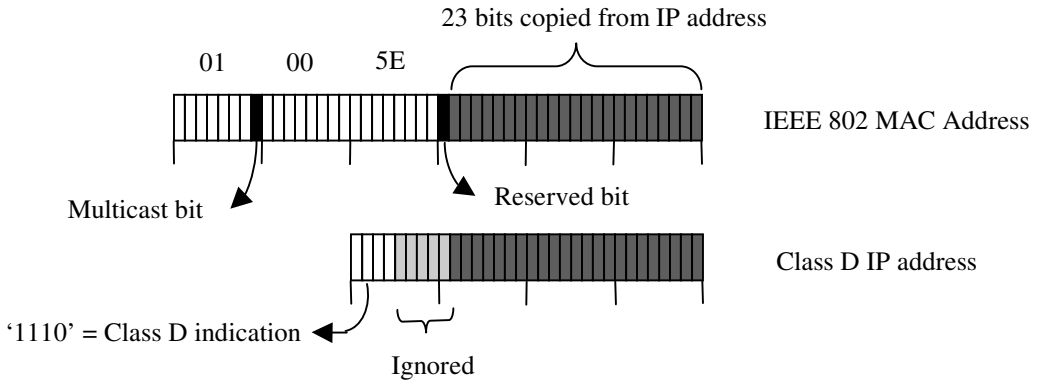
- would like a receiver to join or leave without explicitly notifying sender
  - otherwise it will not scale

Multicast in a broadcast LAN

- Wide area multicast can exploit a LAN's broadcast capability
- E.g. Ethernet will multicast all packets with multicast bit set on destination address

- Two problems:
  - what multicast MAC address corresponds to a given Class D IP address?
  - does the LAN have contain any members for a given group (why do we need to know this?)

Class D to MAC translation



- Multiple Class D addresses map to the same MAC address
  - a host may receive MAC-layer mcast for groups to which it does not belong -> dropped by IP
- Well-known translation algorithm => no need for a translation table

Internet Group Management Protocol

- Detects if a LAN has any members for a particular group
  - If no members, then we can *prune* the shortest path tree for that group by telling parent
- Router periodically broadcasts a *query* message
- Hosts reply with the list of groups they are interested in
- To suppress traffic
  - reply after random timeout
  - broadcast reply
  - if someone else has expressed interest in a group, drop out
- To receive multicast packets:
  - translate from class D to MAC and configure adapter

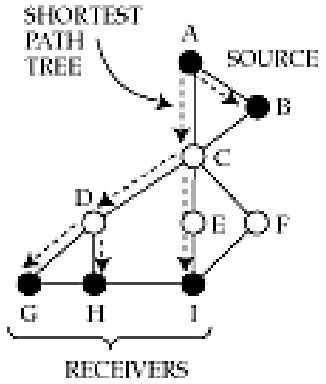
Wide area multicast

- Assume
  - each endpoint is a router
  - a router can use IGMP to discover all the members in its LAN that want to subscribe to each multicast group
- Goal
  - distribute packets coming from any sender directed to a given group to all routers on the path to a group member

Simplest solution

- Flood packets from a source to entire network
- If a router has not seen a packet before, forward it to all interfaces except the incoming one
- Pros
  - simple
  - always works!
- Cons
  - routers receive duplicate packets
  - detecting that a packet is a duplicate requires storage, which can be expensive for long multicast sessions

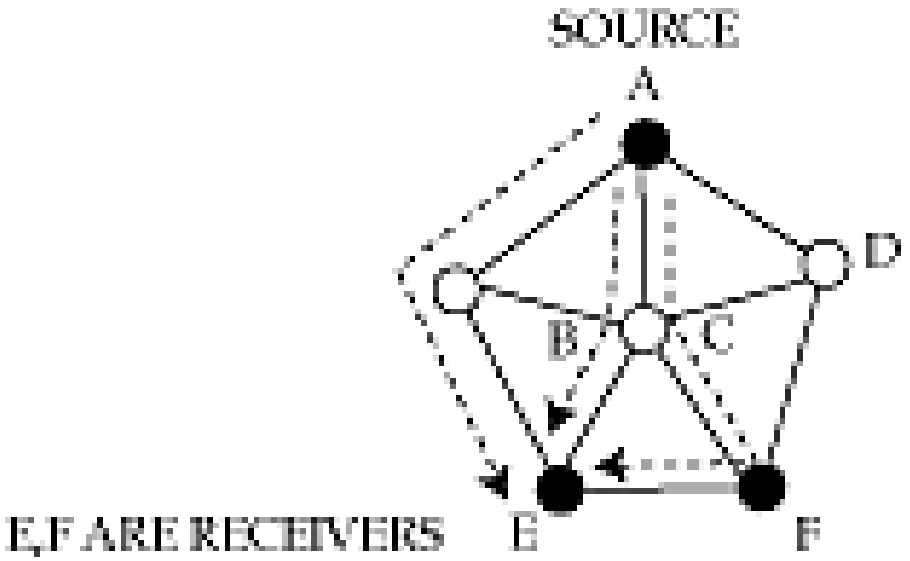
Shortest path tree



- Ideally, want to send exactly one multicast packet per link to reach all interested destinations
  - forms a *multicast tree* rooted at sender
- Optimal multicast tree provides *shortest* path from sender to every receiver
  - *shortest-path* tree rooted at sender

A clever solution

- *Reverse path forwarding*
- Rule
  - forward packet from S to all interfaces if and only if packet arrives on the interface that corresponds to the shortest path *to* S
  - no need to remember past packets
  - C need not forward packet received from D

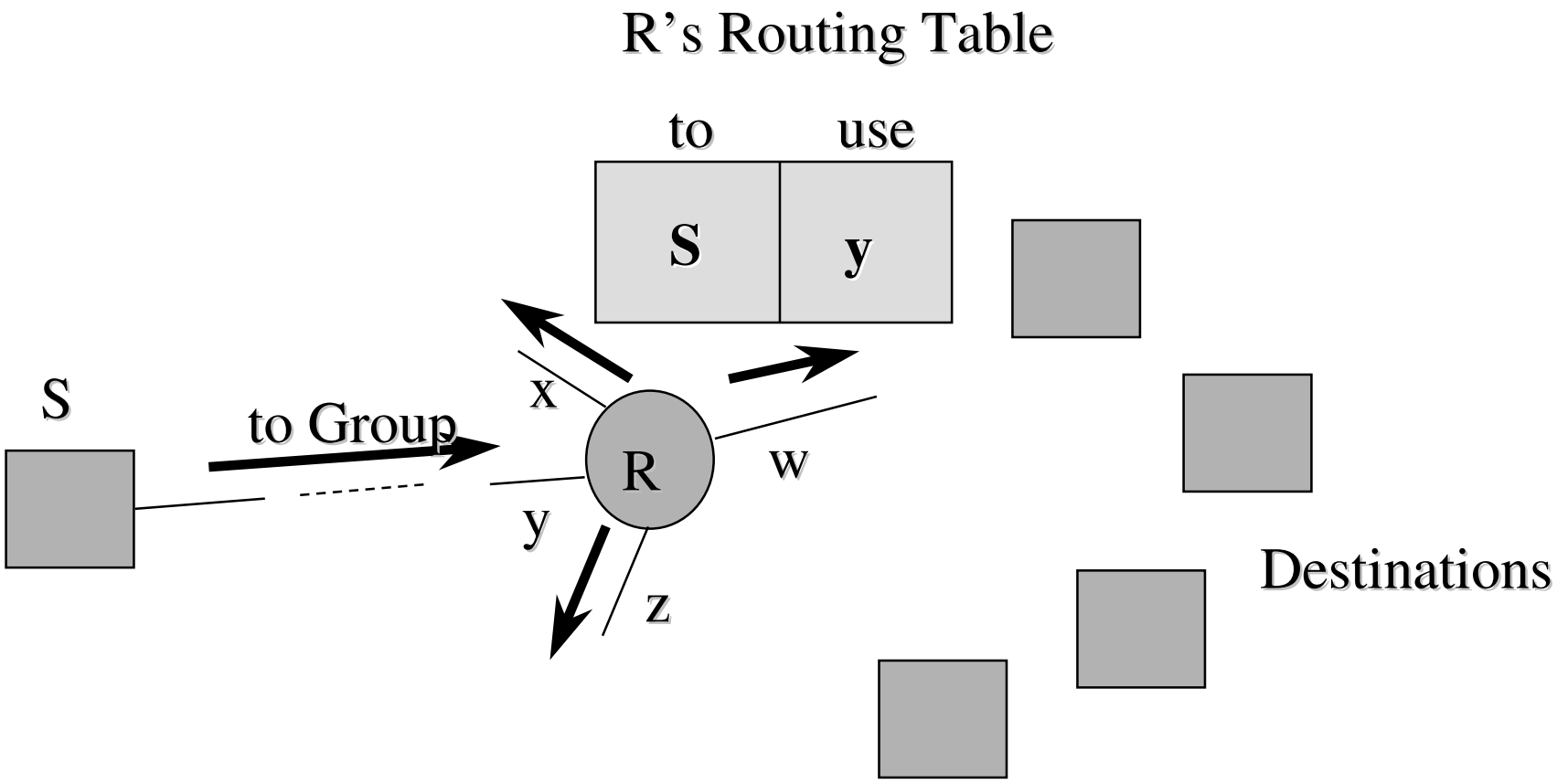


Reverse Path Forwarding

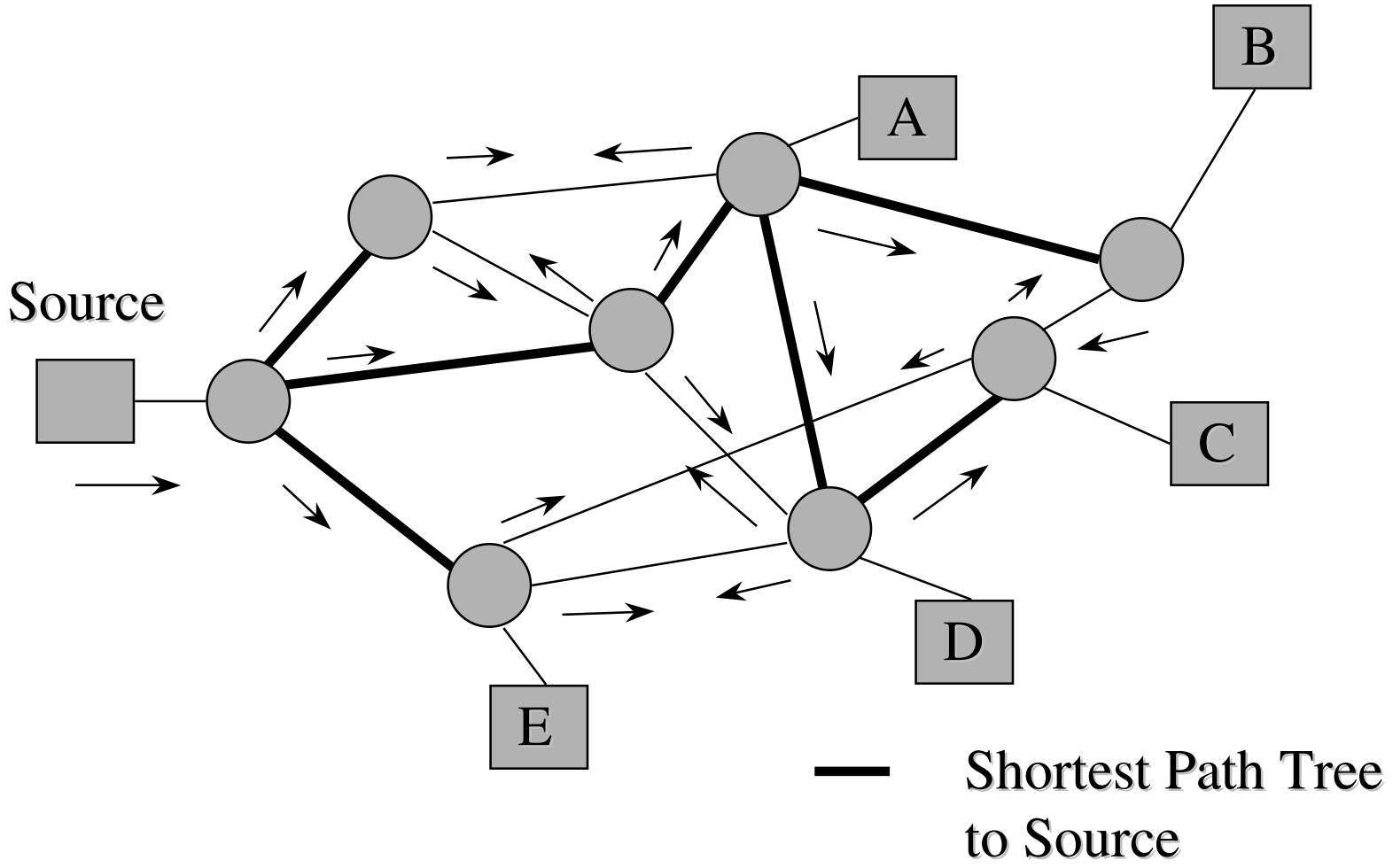
Detailed description

- Routers forward based on *source* of multicast packet
- Flood on all outgoing interfaces if packet arrives from Source on link that router would use to send packets to source
- Otherwise Discard
- Rule avoids flooding loops
- Uses Shortest Path Tree from destinations to source (reverse tree)

Reverse Path Forwarding: router action

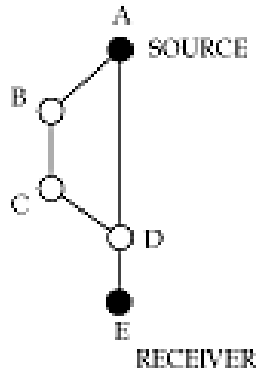


Reverse Path Forwarding



Cleverer

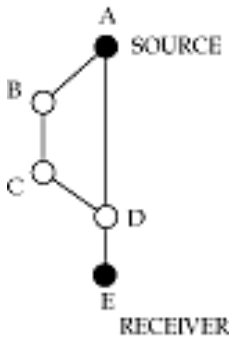
- Don’t send a packet downstream if you are not on the shortest path from the downstream router to the source
- C need not forward packet from A to E



- Potential confusion if downstream router has a choice of shortest paths to source (nodes B and C in figure on slide 28)

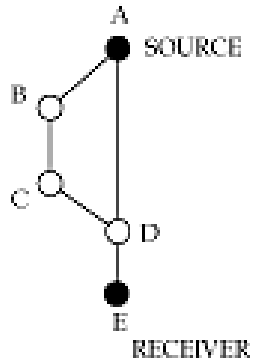
Pruning

- RPF does not completely eliminate unnecessary transmissions



- B and C get packets even though they do not need it
- Pruning => router tells parent in tree to stop forwarding
- Can be associated either with a multicast group or with a source *and* group
  - trades selectivity for router memory

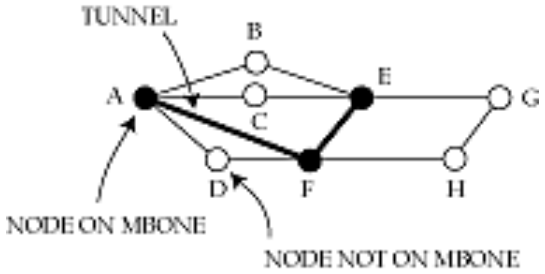
Rejoining



- What if host on C's LAN wants to receive messages from A after a previous prune by C?
  - IGMP lets C know of host's interest
  - C can send a *join(group, A)* message to B, which propagates it to A
  - or, periodically flood a message; C refrains from pruning

A problem

- Reverse path forwarding requires a router to know shortest path to a source
  - known from routing table
- seems straightforward!
- But not all routers do support multicast
  - *virtual links* between multicast-capable routers
  - shortest path to A from E is not C, but F

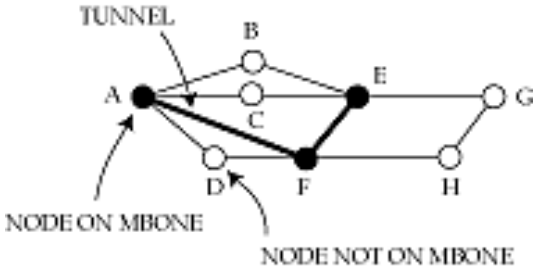


A problem (contd.)

- Two problems
  - how to build virtual links
  - how to construct routing table for a network with virtual links

Tunnels

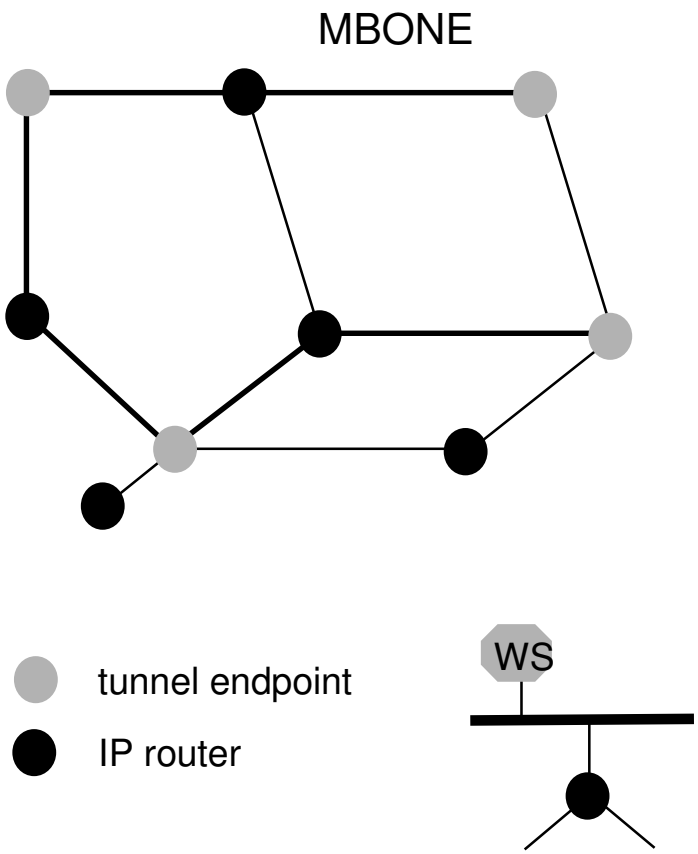
- Why do we need them?



- Consider packet sent from A to F via multicast-incapable D
- If packet's destination is Class D, D drops it
- If destination is F's address, F doesn't know multicast address!
- So, put packet destination as F, but carry multicast address internally
- Encapsulate IP in IP => set protocol type to IP-in-IP

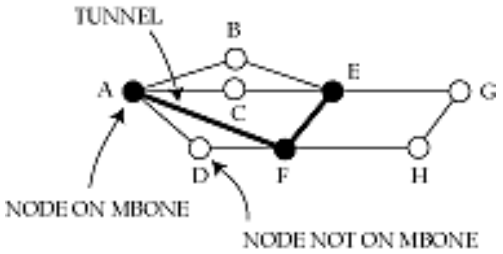
Mbone

- Mbone = multicast backbone
- virtual network overlaying Internet
- needed until mcast capable routers deployed
- IP in IP encapsulation
- limited capacity, resilience



Multicast routing protocol

- Interface on “shortest path” to source depends on whether path is real or virtual



- Shortest path from E to A is not through C, but F
  - so packets from F will be flooded, but not from C
- Need to discover shortest paths only taking multicast-capable routers into account
  - DVMRP

DVMRP

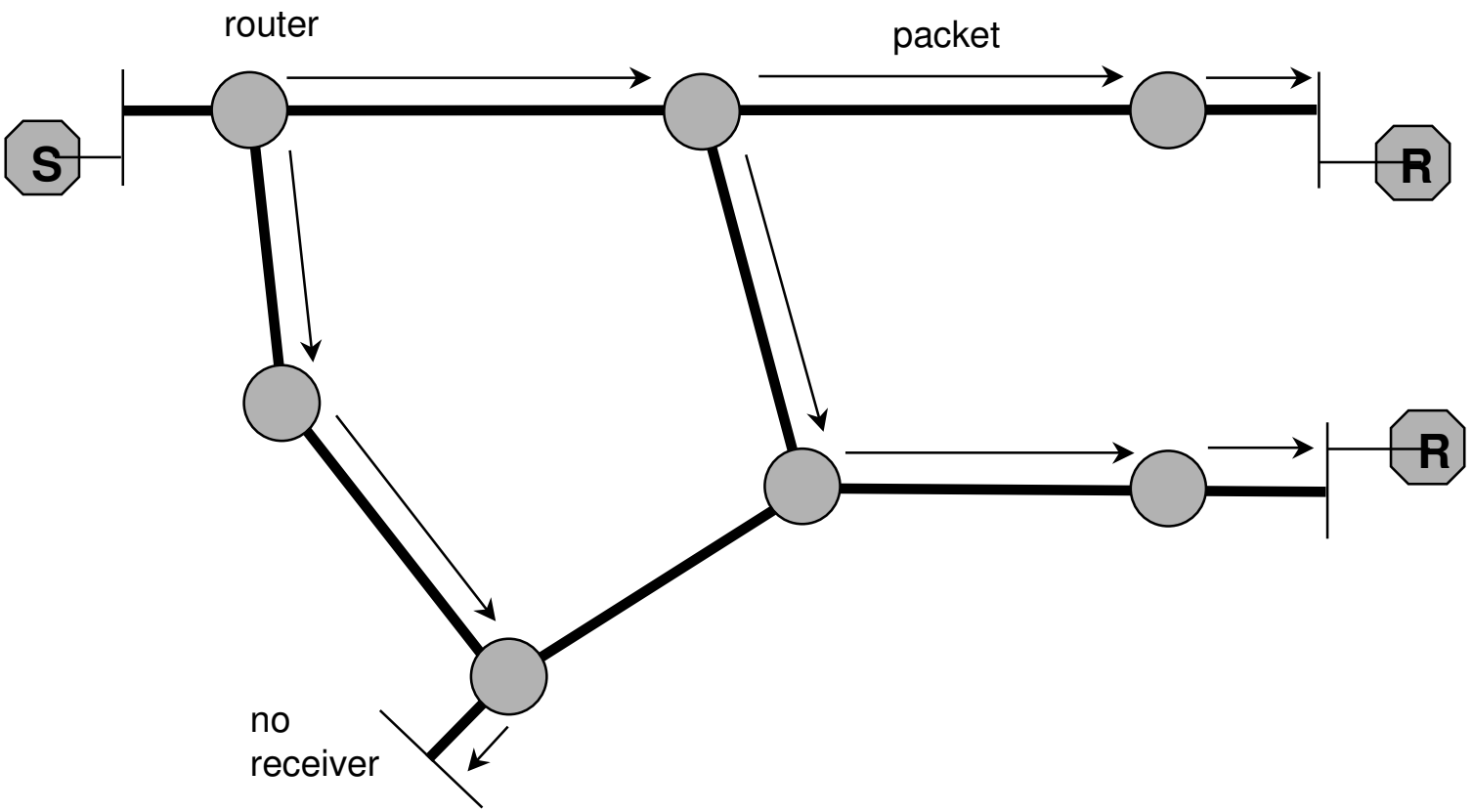
- Distance-vector Multicast routing protocol
- Very similar to RIP
  - distance vector
  - hop count metric
- Used in conjunction with
  - flood-and-prune (to determine memberships)
    - prunes store per-source and per-group information
  - reverse-path forwarding (to decide where to forward a packet)
  - explicit join/graft messages to reduce join latency (but no source info, so still need flooding)

Multicast Forwarding in DVMRP

1. check incoming interface: discard if not on shortest path to source
2. forward to all outgoing interfaces
3. don't forward if interface has been *pruned*
4. prunes time out every two minutes to remove state information in routers

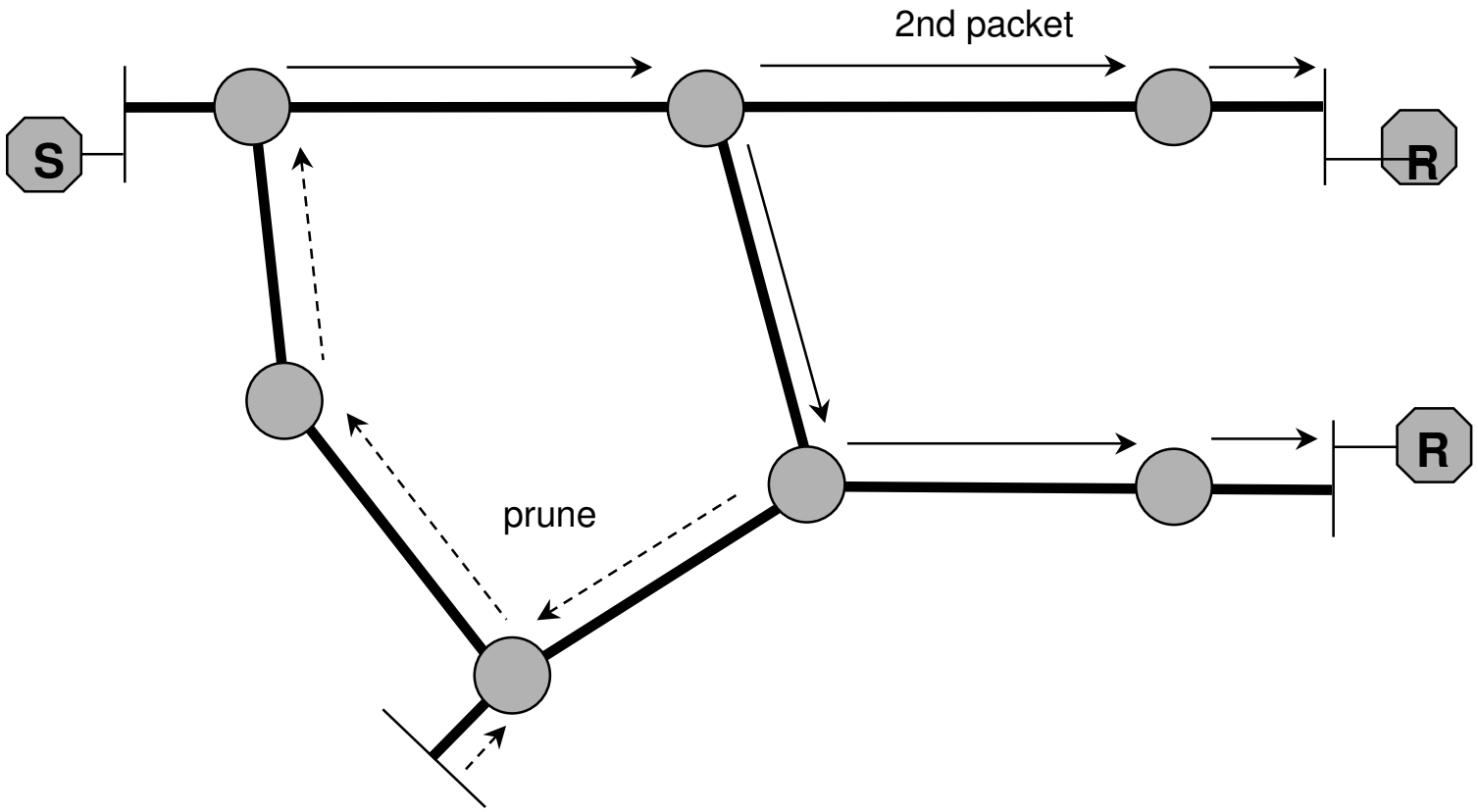
DVMRP Forwarding (cont.)

Basic idea is to flood and prune



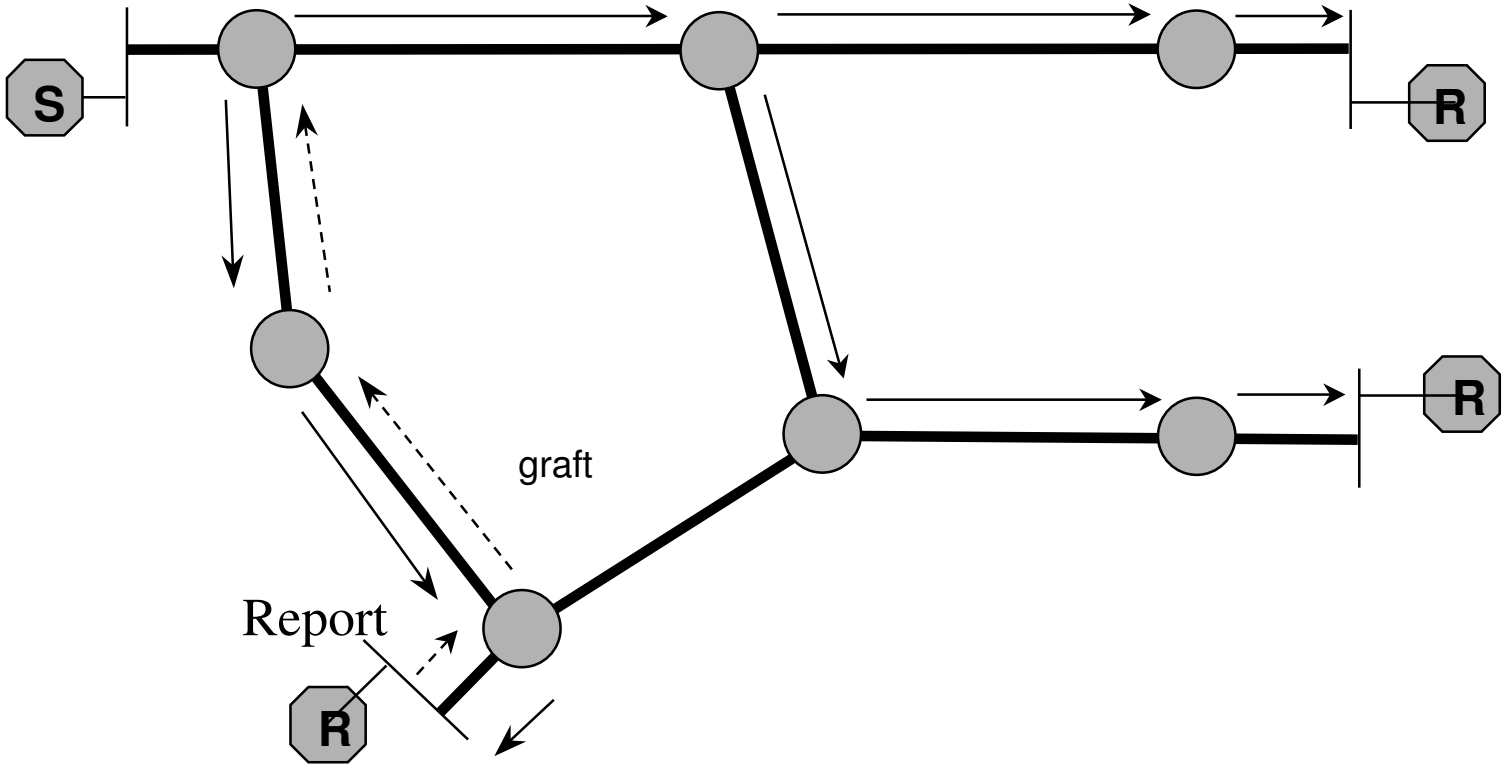
DVMRP Forwarding (cont.)

Prune branches where no members and branches not on shortest paths



DVMRP Forwarding (cont.)

Add new user via grafting; departure via pruning



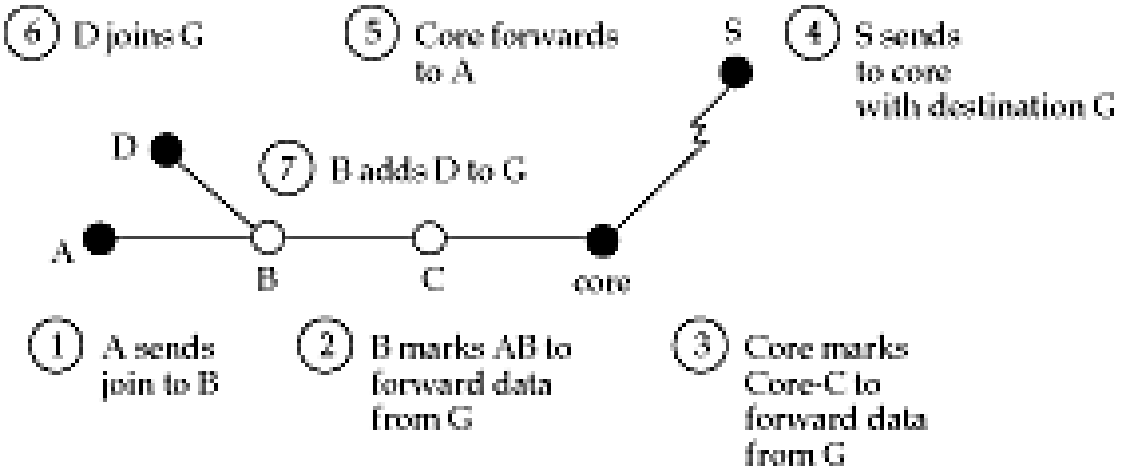
MOSPF

- Multicast extension to OSPF
- Routers flood group membership information with LSPs
- Each router independently computes shortest-path tree that only includes multicast-capable routers
  - no need to flood and prune
- Complex
  - interactions with external and summary records
  - need storage per group per link
  - need to compute shortest path tree per source and group

Core-based trees

- Problems with DVMRP-oriented approach
  - need to periodically flood and prune to determine group members
  - need to store per-source and per-group prune records at each router
- Key idea with core-based tree
  - coordinate multicast with a *core* router
  - host sends a join request to core router
  - routers along path mark incoming interface for forwarding

Example



- Pros
  - routers not part of a group are not involved in pruning
  - explicit join/leave makes membership changes faster
  - router needs to store only one record per group
- Cons
  - all multicast traffic traverses core, which is a bottleneck
  - traffic travels on non-optimal paths

Protocol independent multicast (PIM)

- Tries to bring together best aspects of CBT and DVMRP
- Choose different strategies depending on whether multicast tree is *dense* or *sparse*
  - flood and prune good for dense groups
    - only need a few prunes
    - CBT needs explicit join per source/group
  - CBT good for sparse groups
- Dense mode PIM == DVMRP
- Sparse mode PIM is similar to CBT
  - but receivers can switch from CBT to a shortest-path tree

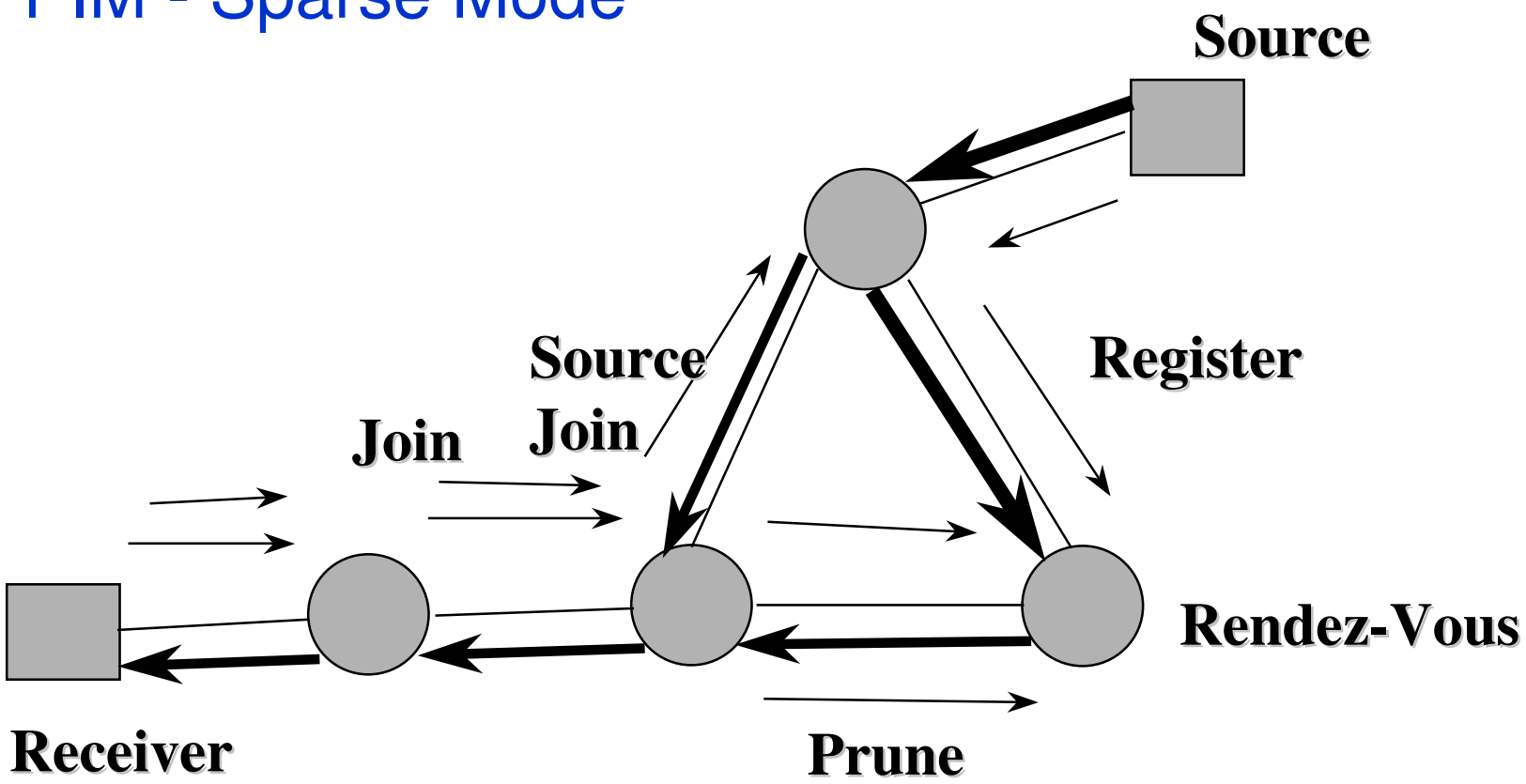
PIM- Dense Mode

- Independent from underlying unicast routing
- Slight efficiency cost
- Contains protocol mechanisms to:
  - detect leaf routers
  - avoid packet duplicates

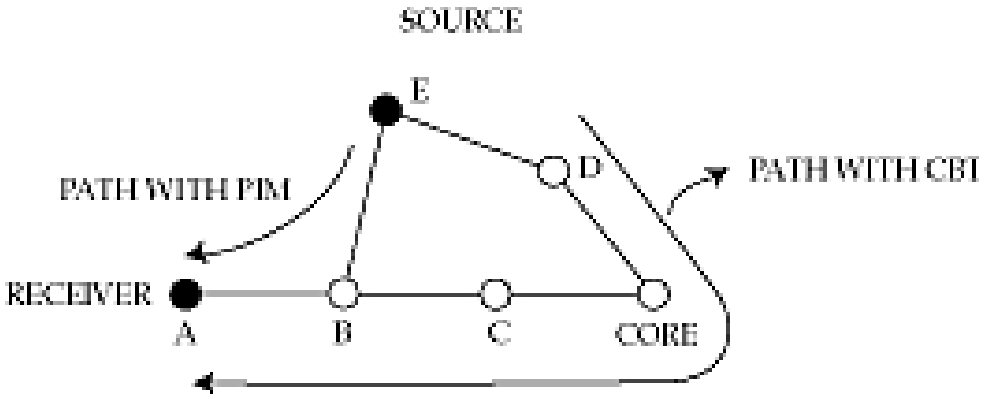
PIM - Sparse Mode

- Rendezvous Point (Core): Receivers Meet Sources
- Reception through RP connection = Shared Tree
- Establish Path to Source = Source-Based Tree

PIM - Sparse Mode



PIM (contd.)



- In CBT, E must send to core
- In PIM, B discovers shorter path to E (by looking at unicast routing table)
  - sends join message directly to E
  - sends prune message towards core
- Core no longer bottleneck
- Survives failure of core

More on core

- Renamed a *rendezvous point*
  - because it no longer carries all the traffic like a CBT core
- Rendezvous points periodically send “I am alive” messages downstream
- Leaf routers set timer on receipt
- If timer goes off, send a join request to alternative rendezvous point
- Problems
  - how to decide whether to use dense or sparse mode?
  - how to determine “best” rendezvous point?

Inter-domain multicast

- Need complex protocols for
  - Intra-domain address allocation
  - Inter-domain address allocation
  - Intra-domain multicast routing (DVMRP/MOSPF/PIM)
  - Build inter-domain multicast tree

The Multi-source Multicast Problem

- Far harder than we thought!

- The rendezvous problem: How does sender in Afghanistan find receivers in Argentina?
  - A highly dynamic global directory at the IP-level?
    - If you can solve this ...
- How to deny this sender if unwanted?
  - still uses network resources unless widely denied
- A global address space of 28-bits, with dynamic allocation by applications
  - not enough bits to allocate

Mission impossible, and for what applications?

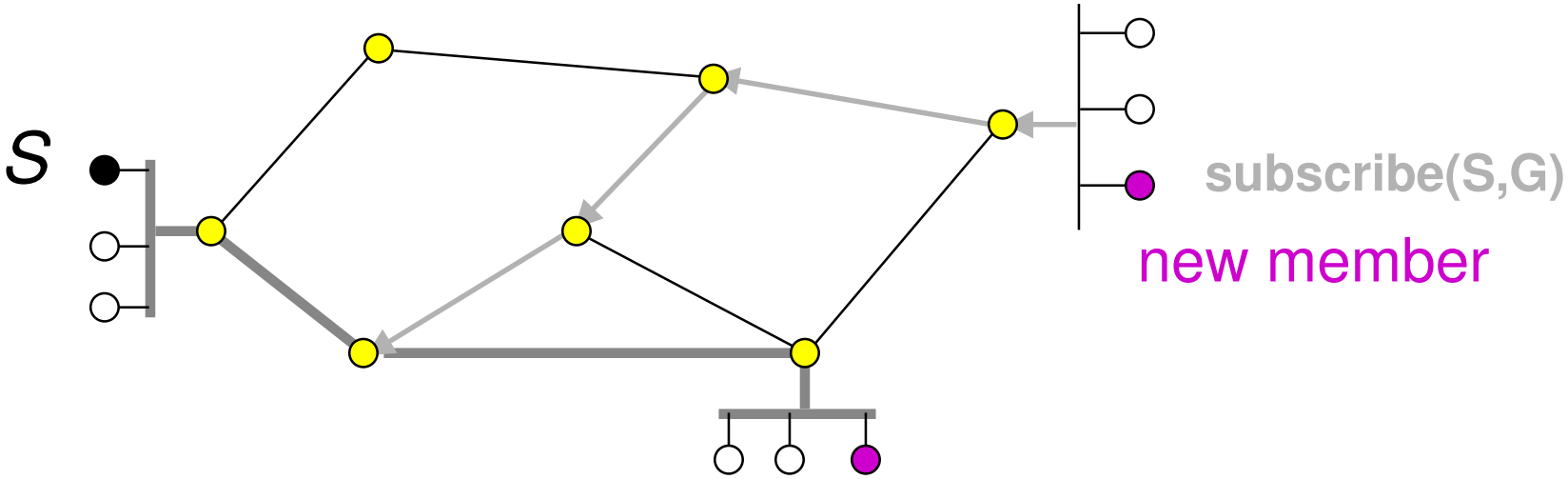
Scalable Multicast Applications?

- Small-scale multicast can just use unicast
  - not pretty but it works, except for discovery
- Large-scale discovery, expanding ring search?
  - 100 million hosts, each occasionally multicasts to the 100 million - how occasional? How about never!
- Most large-scale applications are single-source
  - e.g. Multicast file transfer, Internet TV/radio, web cache update/invalidation

Multi-source multicast is hard and not needed!

Solution: Single-source multicast

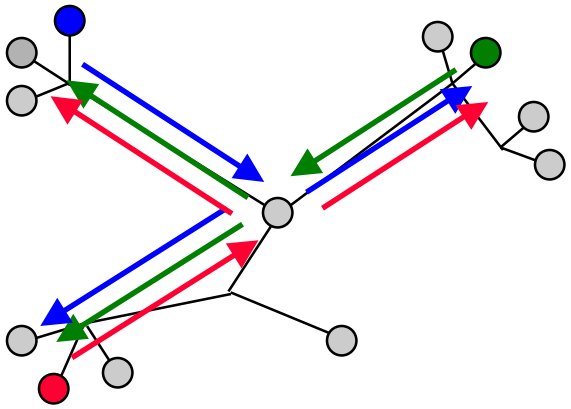
- Key (embarrassingly simple) idea:
  - identify multicast distribution by (S,G)



- Subscription to (S,G) follows unicast path to S
  - relies only on unicast routing information

Scalable multicast routing becomes trivial

Video Conferencing



- 1 channel per participant, if small
  - same cost as for PIM-SM if all are active
- Large video conference - too many channels
- But, floor control and access control is needed
  - “rendevous through an application-level moderator
- Same (or less) cost as PIM-SM at the network layer

SSM is fine with video conferences

Multi-source Group Applications?

- Small groups: unicast or channel per member
- Large groups without structure are mobs
  - Application/middleware-level relays or reflectors needed to provide structure
- Same as a PIM rendezvous point but:
  - can do floor control, access control
  - application can select RP
  - application can select redundancy, fail-over strategy
  - dramatically simplifies the network layer
- Network “middle” boxes can provide relay service

SSM + relays is superior for multi-source apps

Benefits

- Solves the scalable multicast routing problem:
  - join protocol, building on proven unicast routing
- Solves the access control problem:
  - only source can send (duh!)
- Solves multicast address allocation problem
  - thousands of multicast addresses per-source
- Simplifies the network layer

15 years to realize we were putting too much at the network level

The Internet deployment delayed as a result

Some Lessons

- We need to ask hard questions about real compelling applications
  - The service model should be a slave to these applications
- We had the wrong service model:
  - SSM channel model versus the group model
- End-to-end (again): do not put functionality at the lower level unless there is a real win,
  - because it can be a real lose!
- Group communication is hard

Dimensionality of Group Comm.

- Size - how many
- Reliability - of message delivery
- Timing - synchronized with receivers
- Proximity - near by, far away
- Similarity - homo. vs. hetero. nodes
- Network connectivity - fast/slow, errors, etc.

Group comm. Is far more diverse than unicast; no, far, far far more diverse

Size of groups

- With unicast, its one - the other end
- With group communication, it could be:
  - 10
  - 100
  - 1000
  - 10,000
  - 100,000
  - 1,000,000

So, how many solutions to group problems are there? Many, many?

Reliability

- With unicast, the other end needs to receive the packet
- With group communication, it could be we need:
  - one of the group to receives it
  - a few of the group to receive it
  - Most ...
  - all?

Again, not just one single group problem,

but many!

Timing

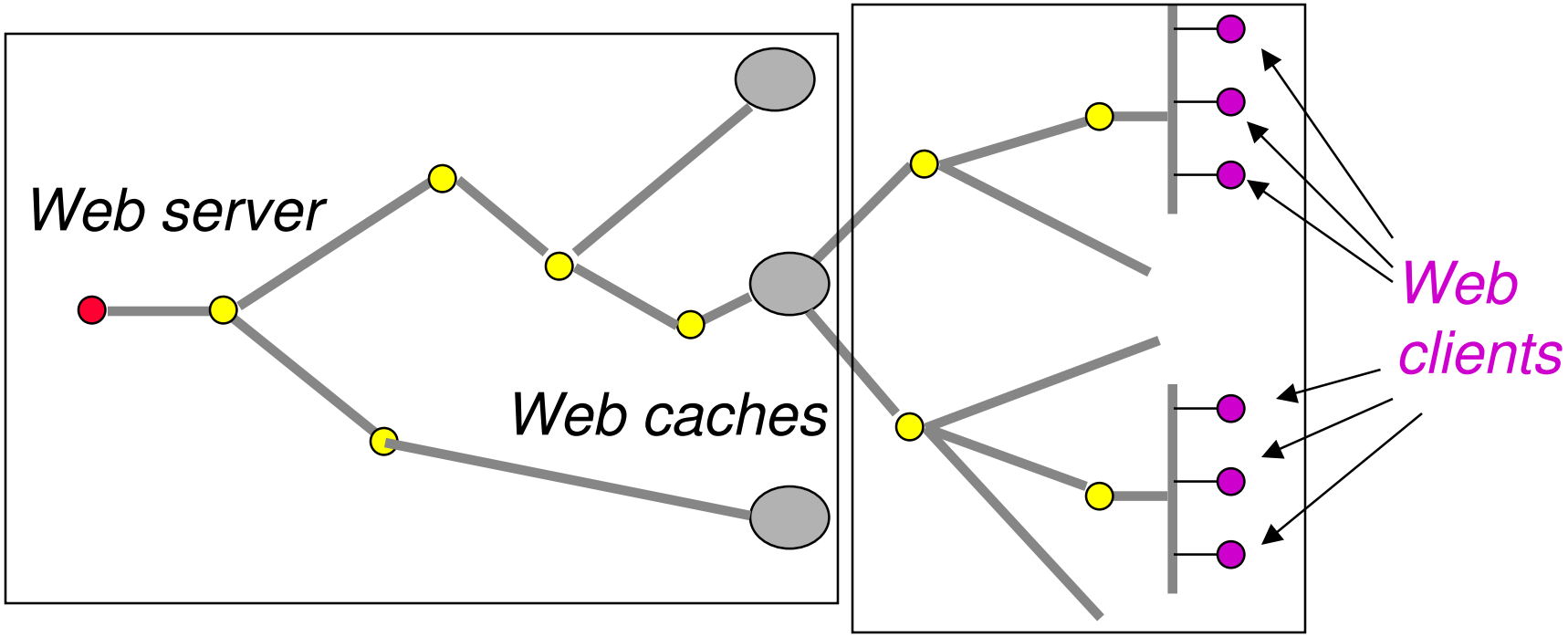
- With unicast, the other end is ready to receive
- With group communication, it could be:
  - one of the group is ready for the message
  - a few of the groups are ready
  - several ...
  - Most
  - all?

Not just one group problem, but many?

Group Comm. is not Scalable!

- Any middleware has to deal with the cross-product of (some of) these dimensions
  - seems hopeless
- Within one application:
  - the larger the group, the more the time skew
  - handling time skew implies longer-term storage
  - long-term storage (disk) not part of the real-time store-and-forward communication layer.
- Alternative: file-and-forward networking

Example: Web Multi-point Delivery



- A group distribution tree but ...
- Web cache as a “file-and-forward” node
- Limited scale groups from caches to clients
- Limited scale groups from server to caches

A combination of comm. and storage nodes

Summary

- Multicast Routing is well researched problem.
- However, challenge now is
  - deployment
  - inter-operability
  - management

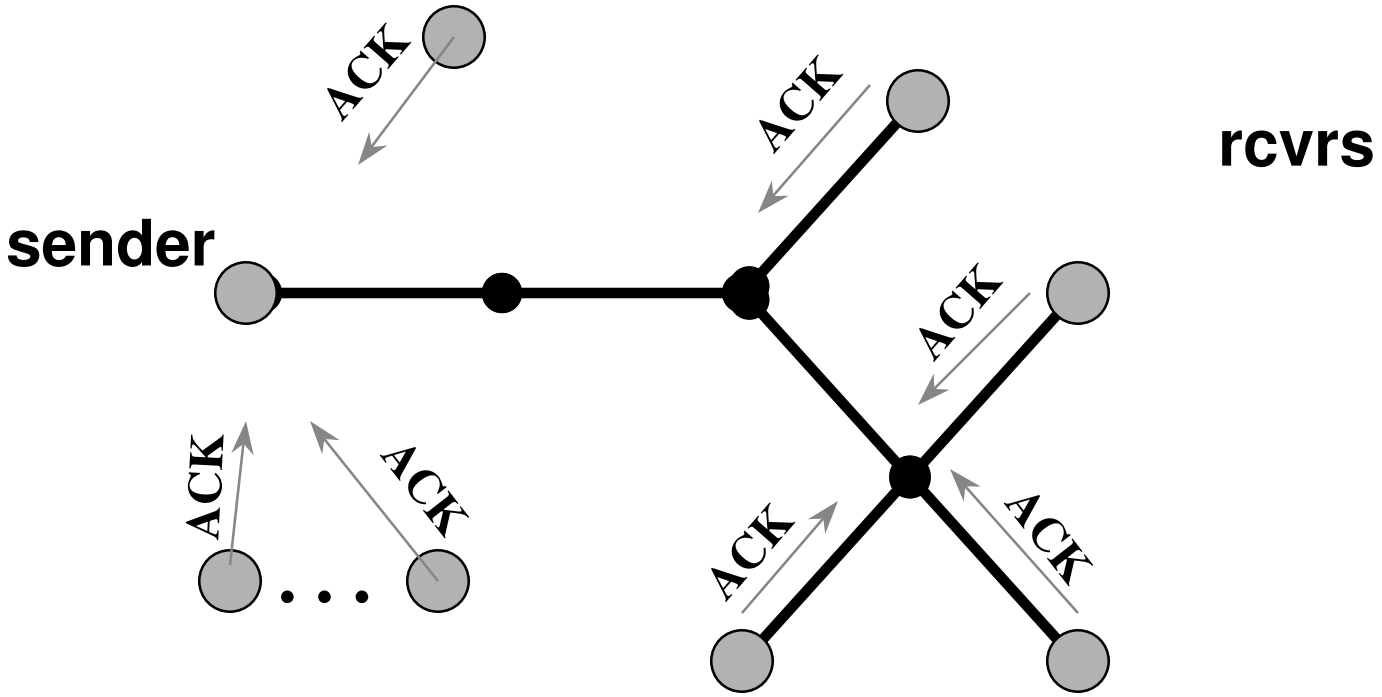
Reliable Multicast

Problem

How to transfer data reliably from source to R receivers

- scalability: 10s -- 100s -- 1000s -- 10000s -- 100000s of receivers
- heterogeneity
- feedback implosion problem

Feedback Implosion Problem



Issues

- level of reliability
  - full reliability
  - semi-reliability
- ordering
  - no ordering
  - ordering per sender
  - full ordering (distributed computing)

Applications

- application requirements
  - file transfer, finite duration
  - streaming applications (billing, etc.), infinite duration
  - low latency (DIS, teleconferencing)
- application characteristics
  - one-many: one sender, all other participants receivers (streaming appl. teleconferencing)
  - many-many: all participants send and receive (DIS)

Approaches

- shift responsibilities to receivers
- feedback suppression
- server-based recovery
- local recovery
- forward error correction (FEC)

Sender Oriented Reliable Mcast

Sender: mcasts all (re)transmissions

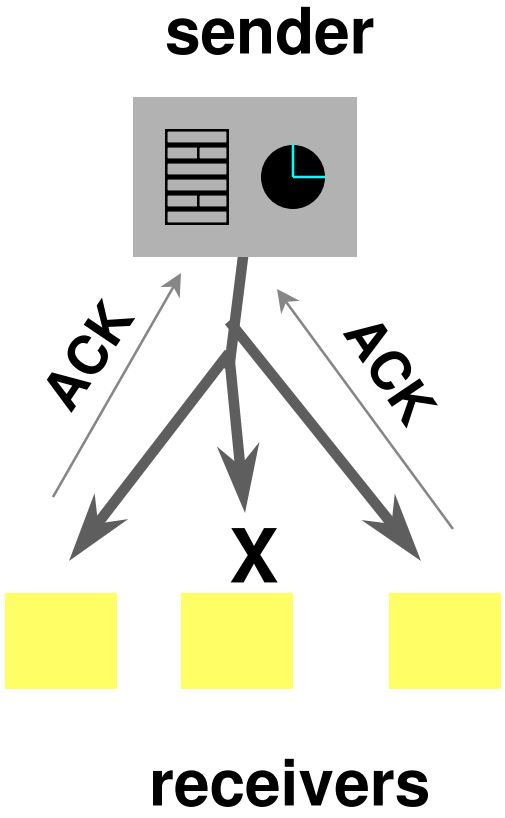
selective repeat

use of timeouts for loss detection

ACK table

Rcvr: ACKs received pkts

Note: group membership important



Vanilla Rcvr Oriented Reliable Mcast

Sender: mcasts (re)transmissions

selective repeat

responds to NAKs

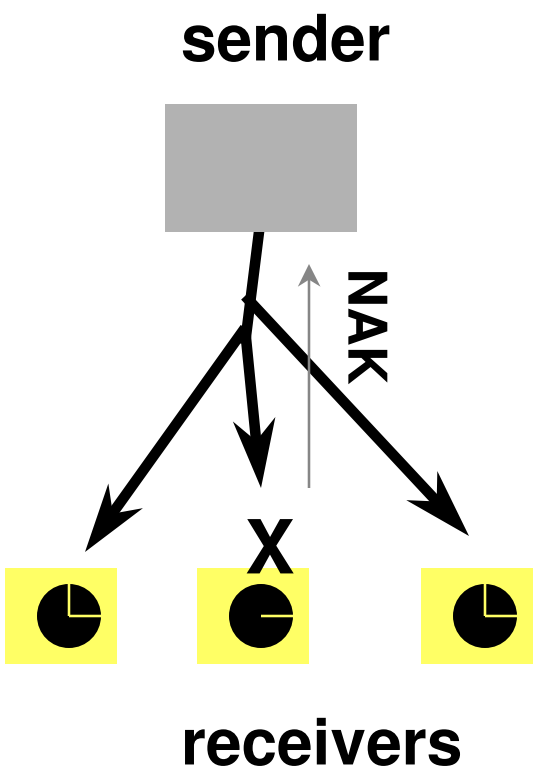
Rcvr: upon detecting pkt loss

sends pt-pt NAK

timers to detect lost retransmission

Note: easy to allow joins/leaves

Significant performance improvement
shifting burden to receivers for 1-many; not as great for many-many

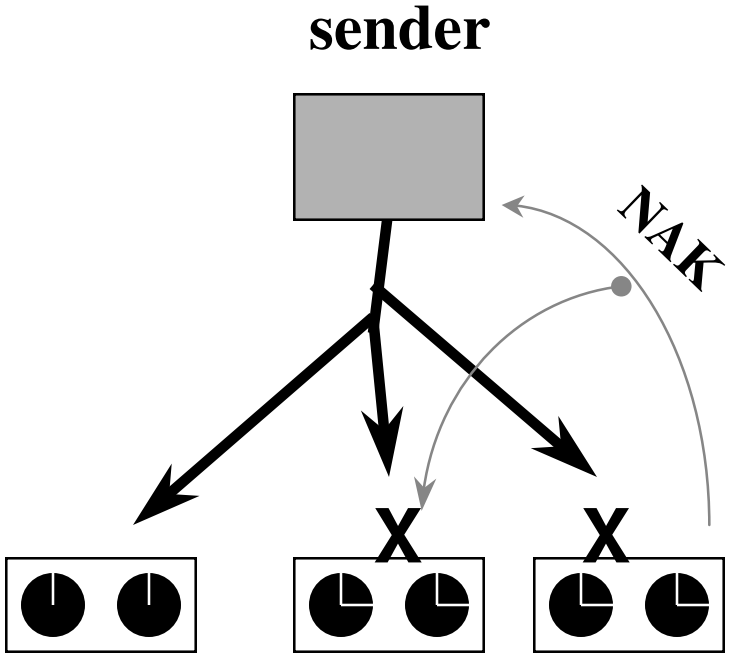


Feedback Suppression

- randomly delay NAKs
- multicast to all receivers

+ reduce bandwidth

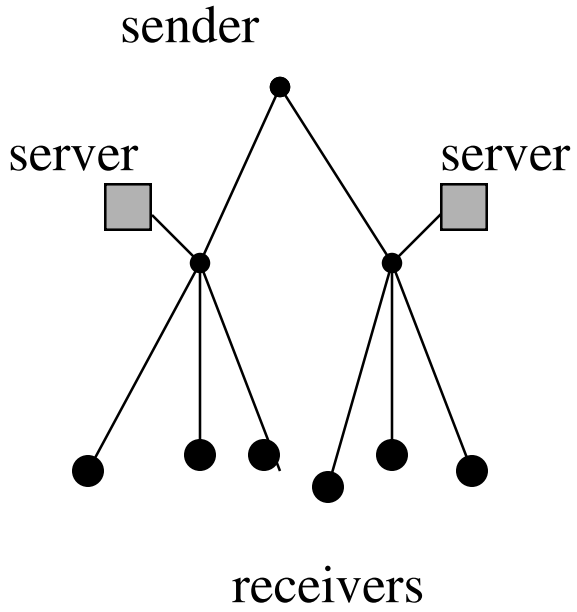
- additional complexity at receivers (timers, etc)



Server-based Reliable Multicast

- first transmisions mcast to all receivers and servers
- each receiver assigned to server
- servers perform loss recovery
- can have more than 2 levels

LBRM (Cheriton)



Local Recovery

Lost packets recovered from nearby receivers

- deterministic methods
  - impose tree structure on rcvrs with sender as root
  - rcvr goes to upstream node on tree

    RMTP (Lucent)
- self-organizing methods
  - rcvrs elect nearby rcvr to act as retransmitter using scoped multicast and random delays (SRM)
- hybrid methods

RMTP (Lucent)

Reliable Multicast Transport Protocol

- imposes a tree structure on rcvrs corresponding to multicast routing tree
- nodes inside tree
  - aggregate ACKs/NAKs
  - provide repairs to downstream nodes
- late-joins supported thru 2-level cache
- rate- and window-based flow control

SRM (LBL)

Scalable Reliable Multicast

- rcvr-oriented using NAK suppression and self-organizing local recovery
- supports late-joins and leaves
- as built in wb, uses rate-based flow control
- has been used with 100s of participants over the Internet

SRM detailed operation

- NACKs and retransmissions are multicast
  - suppress duplicates, random delay before replying
- Each host estimates the « delay » with all other hosts
  - schedule an action after a randomization delay
  - chose a smaller delay for NACKs/retransmissions if closer to source/requesting host
- If other request/repair received
  - cancel action, double interval
- If timer expired
  - do action, double interval

Forward Error Correction (FEC)

Add redundancy in order to reduce need to recover from losses (e.g., Reed Solomon codes)

(k,n) code

- for every k data pkts, construct n-k parity pkts
- can recover all data pkts if no more than n-k losses

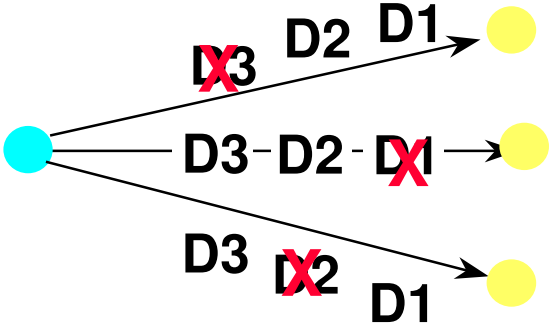
+ reduce loss probability

- greater overheads at end-hosts

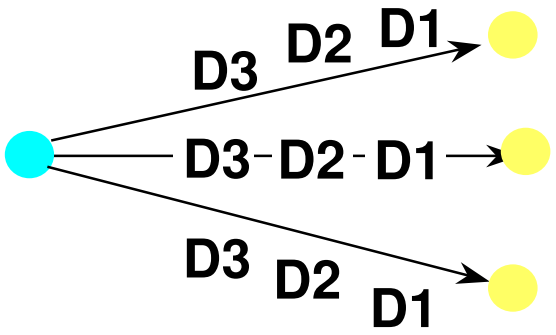
Q: can FEC reduce network resource utilization?

Potential Benefits of FEC

Initial Transmission

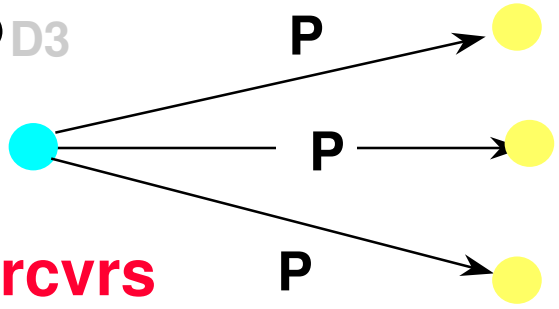


Data Retransmission



Parity Retransmission

$P = D1 \otimes D2 \otimes D3$



One parity pkt can recover different data pkts at different rcvrs

Summary

- reliable mcast is a hot research topic
- unresolved issues
  - proper integration of different ideas wrt different applications
  - integration with flow/congestion control
  - interaction with group membership
  - notion of semi-reliability

Multicast congestion control

Problem

- Match transmission rates to
  - Network capacity
  - Receiver “consumption” rates

Multicast Flow Control Challenges

- Accommodating *heterogeneity* among receivers and paths leading to them
- Preserving *fairness* among
  - receivers of same flow
  - distinct flows
- *Scalability* of feedback

Multicast Flow Control Solutions

- Loss-Tolerant Applications (e.g., Video)
  - Information content per unit time can be preserved at lower data rates
- Applications demanding data integrity
  - lower data rates => lower information content per unit time

- *Goal*: Co-Existence with TCP?

Single rate congestion control

- Scalable Feedback Control
  - Receivers measure loss rates
  - Randomly generated feedback
  - Source estimates receivers' state and adjusts video rate by changing compression parameters
- Source adapts to "slowest" receiver
  - or another "single" rate
- Problem: fairness among receivers

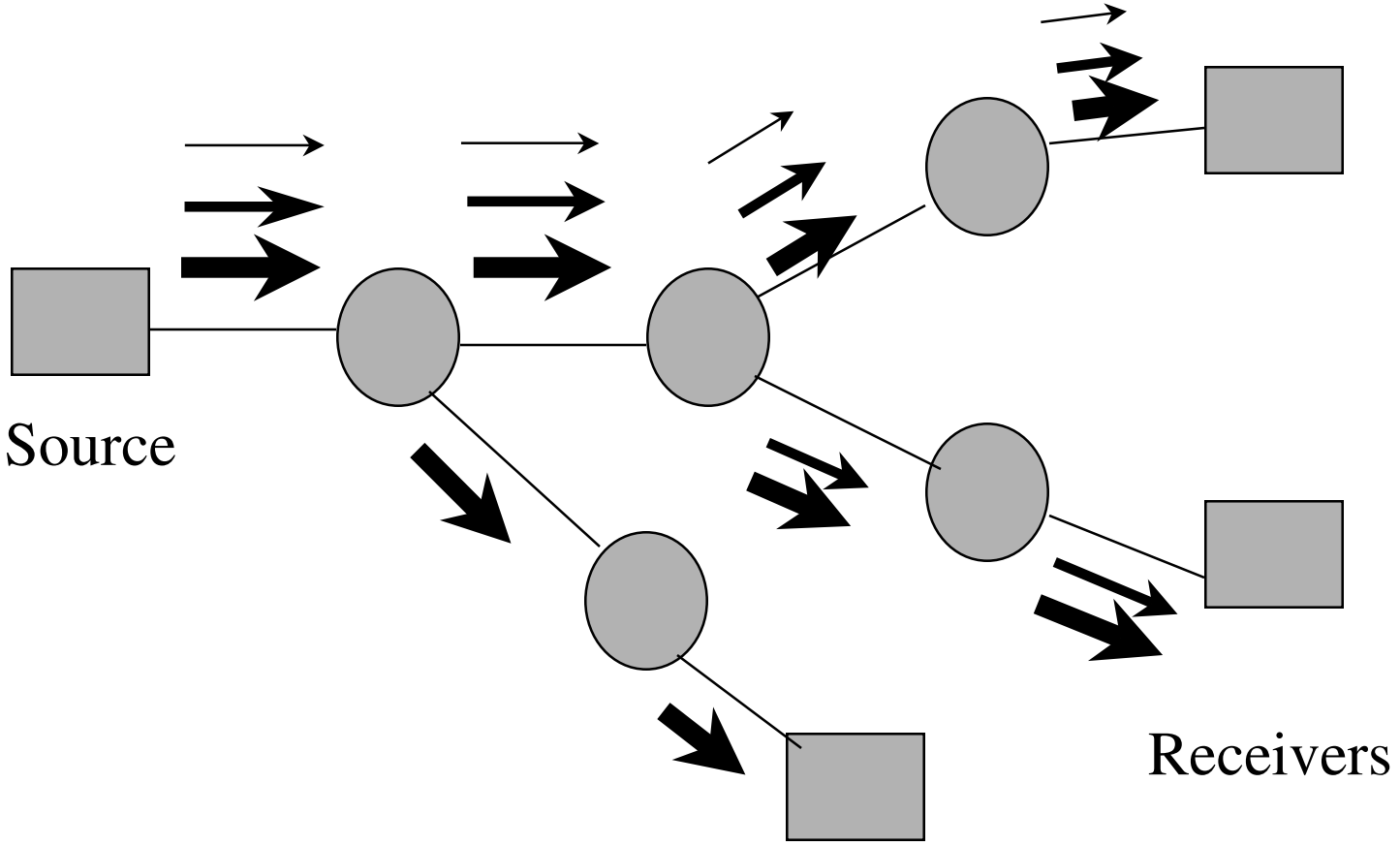
Simulcast

- Improving *fairness* using
- Send replicated video streams at different rates
  - Receivers can control rate of each stream within limits
  - Receivers can move among streams
- Fairness at the expense of increased bandwidth consumption

Receiver-driver Layered Multicast

- Single video stream subdivided into layers
- Receivers add and drop layers depending on congestion
- Challenge: Distributed Consensus, Layer Synchronization

Receiver-driven Layered Multicast



Receiver-Driven Layered Multicast

- Drop Layer:
  - indicated by loss
- Add Layer:
  - No such indication
- Use join experiments with shared learning
  - Reluctance to join layers that failed
  - Inform others via multicast of failed experiments

Flow Control for Reliable Multicast

- Less Understood/Mature Area
- Some Possibilities:
  - Window flow control (a la TCP)
    - Not Scalable, Not Fair (across receivers)
  - Multiple Multicast Groups

Multiple Multicast Groups

- Simulcasting or Destination Set Splitting
- Cumulative Layering

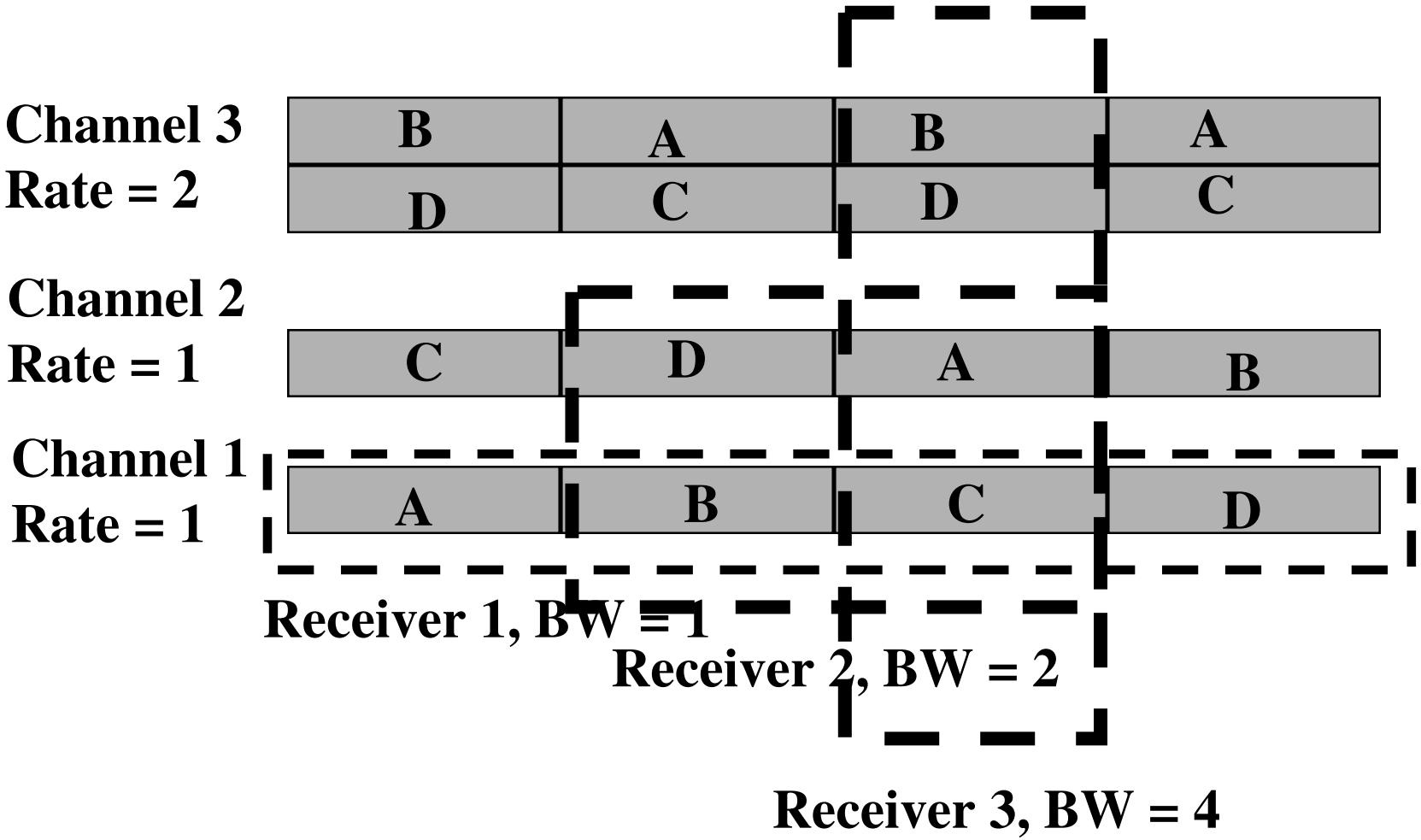
Simulcasting

- Similar to the Video case
- Send multiple (*uncoordinated* streams) at different rates
- Each stream carries all data
- Receivers join appropriate stream - *one at a time*

Cumulative Layering

- Multiple data streams at different rates
- Each stream contains entire data
- Receivers join asynchronously -- Streams transmit continuously
- Schedule to minimize reception time
- (Scheme allows for FEC encoding)

Cumulative layering



Cumulative Layering

- Can achieve minimum reception time with asynchronous receivers
- Schedulability requires some parameter relationships
- Synchronization among channels needed

Summary

- Flow control for multicast communication is a **hard** problem:
  - Scalability
  - Heterogeneity
  - Added dynamic dimension (receivers and their join behavior)
  - Plenty of room for innovation!