Lecture 4:

Tree Algorithms & Multicast

(Some slides are from Massoud Pedram, Srini Seshan, and Ion Stoica)

Dina Katabi

nms.csail.mit.edu/~dina

Broadcast Trees

Minimum Spanning Tree (MST)

Spanning subgraph

- Subgraph of a graph G containing all the vertices of G

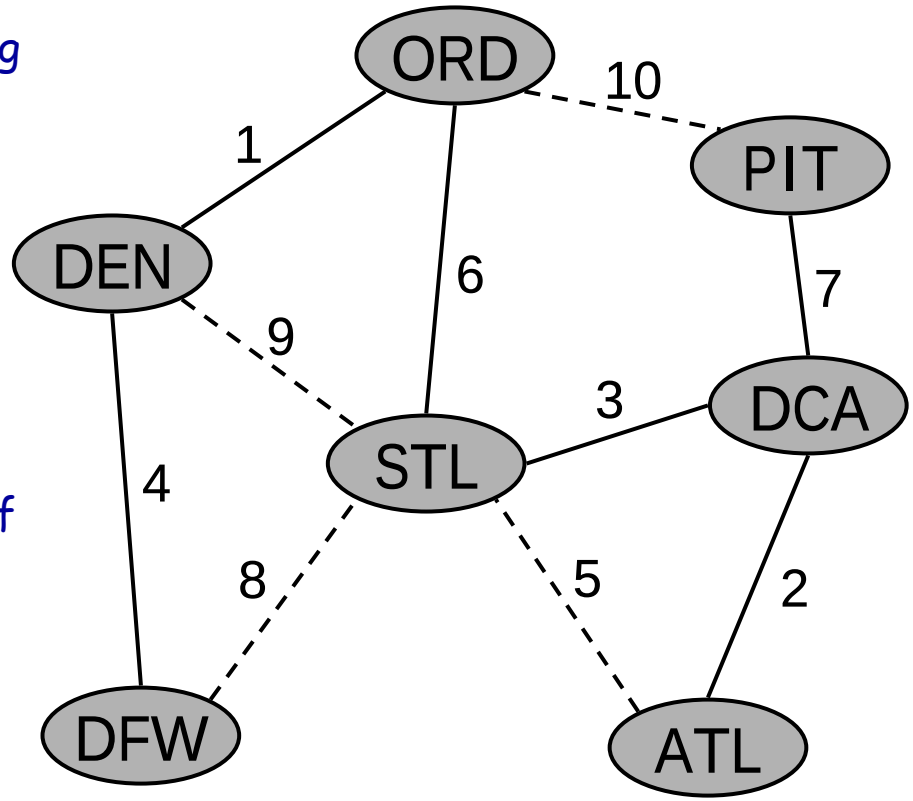
Spanning tree

- Spanning subgraph that is itself a tree (cycle free)

Minimum spanning tree (MST)

- Spanning tree of a weighted graph with minimum total edge weight (i.e., minimizes the sum of the edge weights)

- Applications
  - Communications networks
    - Broadcast
- *Is the MST unique?*



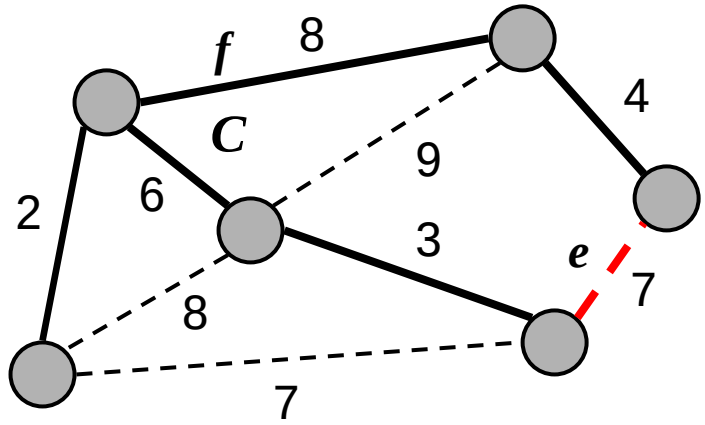
Cycle Property

Cycle Property:

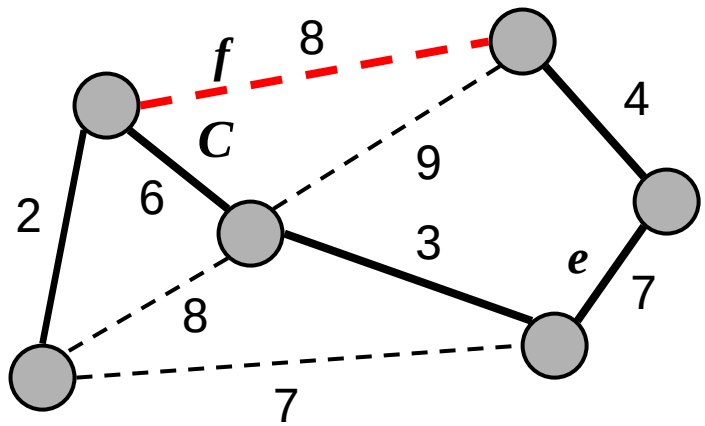
- Let T a be a minimum spanning tree of a weighted graph G
- Let e be an edge of G that is not in T and let C be the cycle formed by e with T
- For every edge f of C, $weight(f) \leq weight(e)$

Proof:

- By contradiction
- If $weight(f) > weight(e)$ we can get a spanning tree of smaller weight by replacing e with f

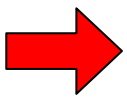


Replacing f with e yields a better spanning tree



Fragment Property

- Let F be a fragment of an MST
  - If *e* is the minimum weight edge such that F U {e} does not contain a cycle
  - Then F U {e} is a fragment of an MST
- Proof by contradiction similarly to the cycle property



Greedy works for building MST

Prim's Algorithm

- $T = \phi$
- S = {v} for an arbitrary vertex v
- Repeat until S contains all the vertices:
  - Add the lightest (i.e., minimum cost) edge $e(v_i,v_j)$ to T where $v_i \in S$ and $v_j \in (V-S)$. Add v_j to S

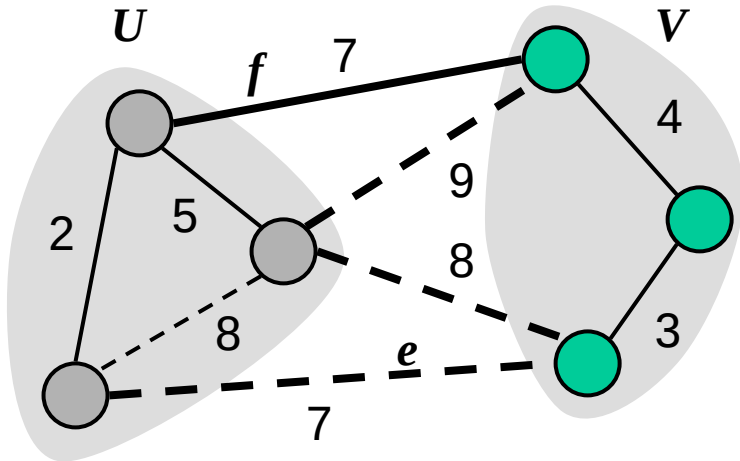
Partition Property

Partition Property:

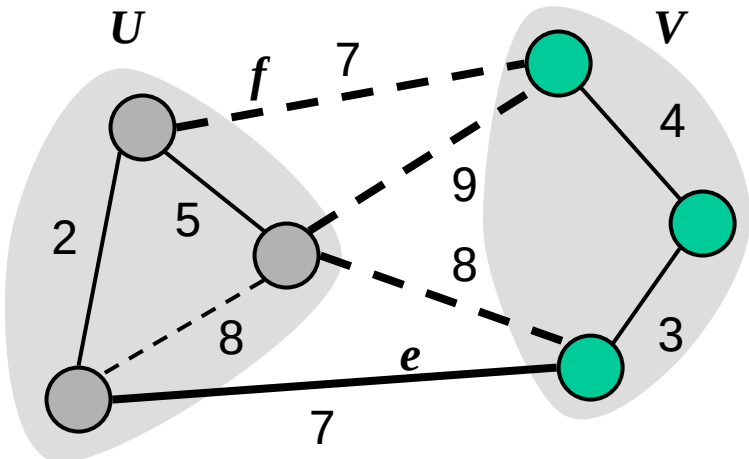
- Consider a partition of the vertices of G into subsets U and V
- Let e be an edge of minimum weight across the partition
- There is a minimum spanning tree of G containing edge e

Proof:

- Let T be an MST of G
- If T does not contain e, consider the cycle C formed by e with T and let f be an edge of C across the partition
- By the cycle property,
$$weight(f) \leq weight(e)$$
- Thus, $weight(f) = weight(e)$
- We obtain another MST by replacing f with e



Replacing f with e yields another MST



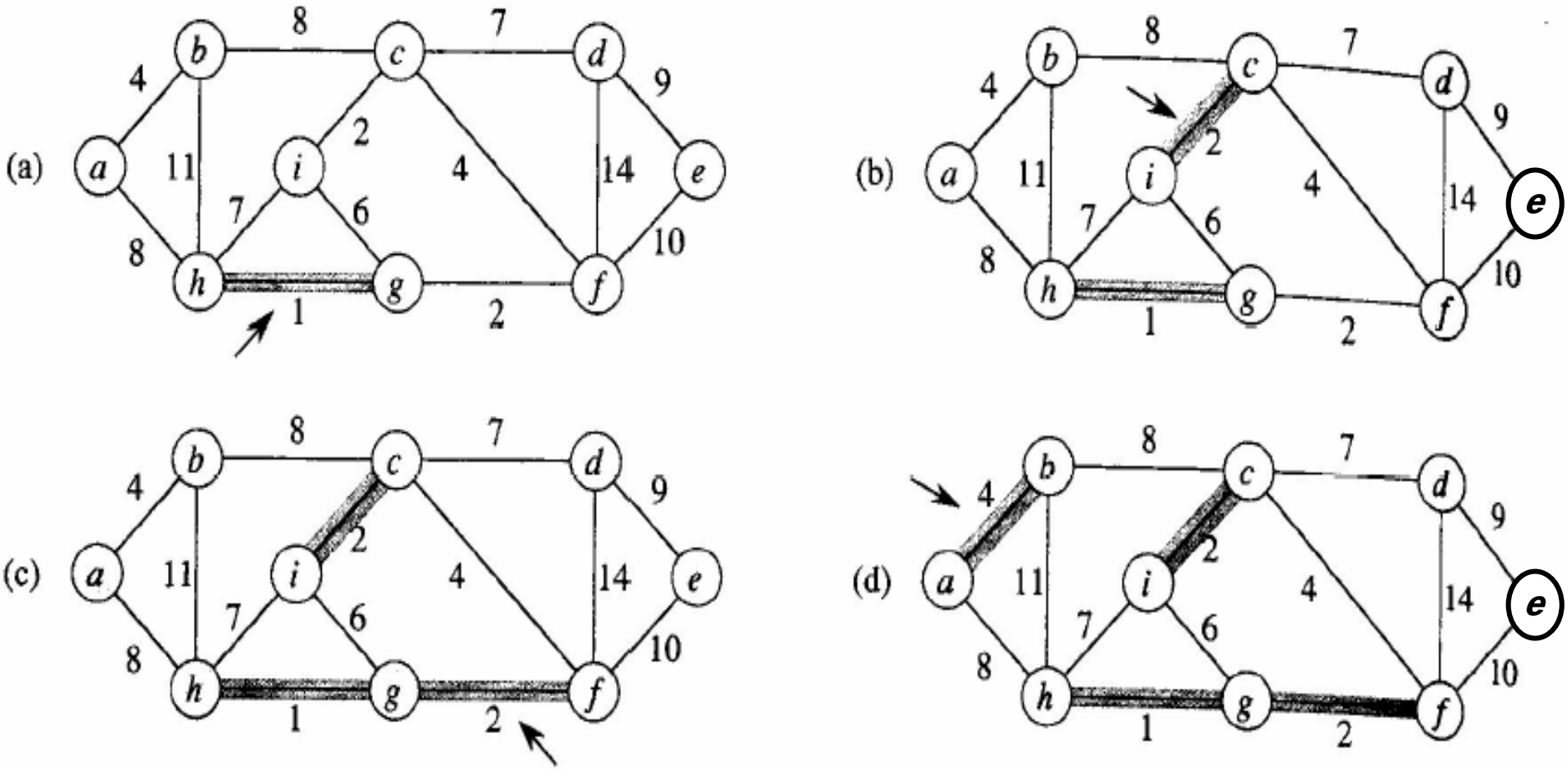
Kruskal's Algorithm

- Sort the edges
- T = ϕ
- Consider the edges in ascending order of their weights until T contains n-1 edges:
  - Add an edge e to T exactly if adding e to T does not create any cycle in T

- *Is this a centralized or distributed alg?*
- *What 's the complexity of kruskal?*

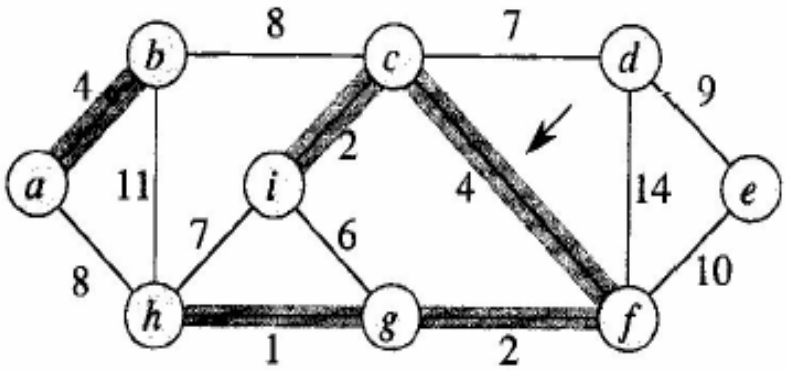
How do we find the MST of a graph?

Kruskal's Algorithm Example

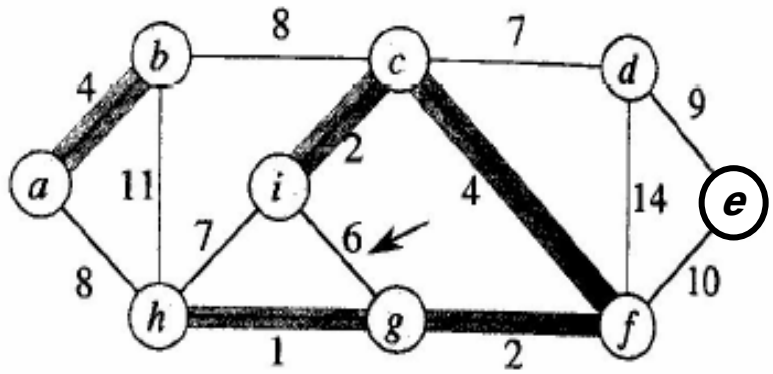


Kruskal's Algorithm Example (Cont'd)

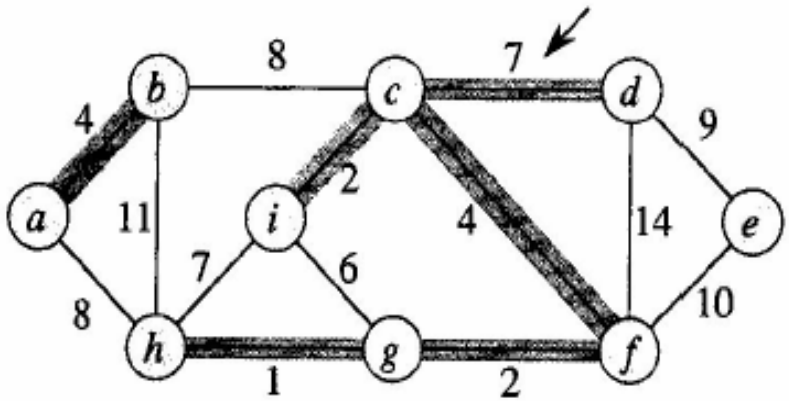
(e)



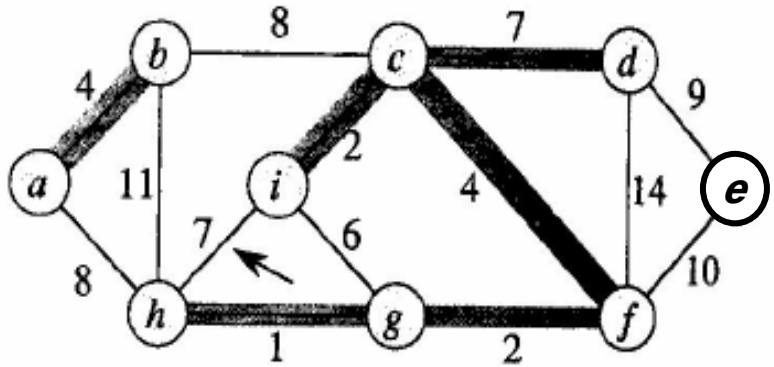
(f)



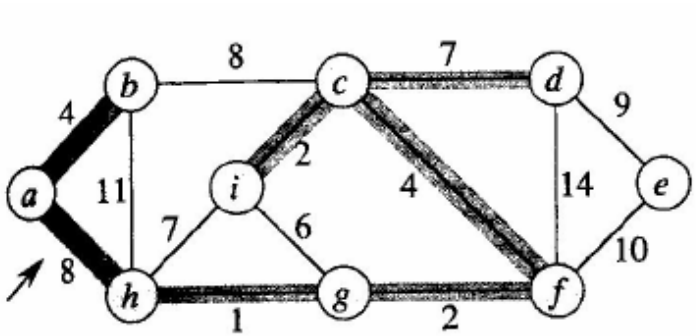
(g)



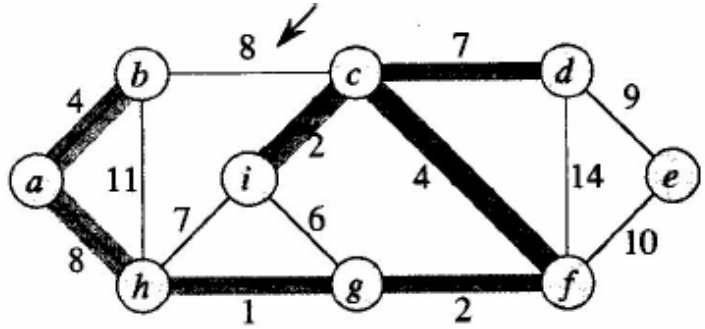
(h)

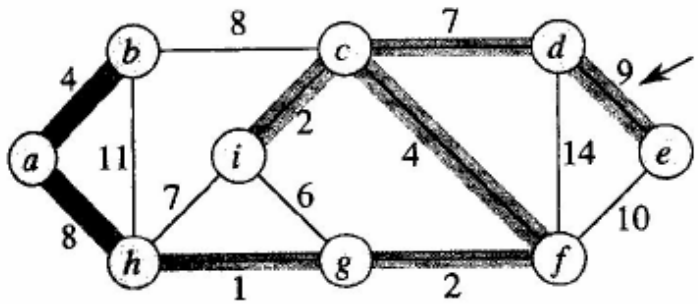


Kruskal's Algorithm Example (Cont'd)

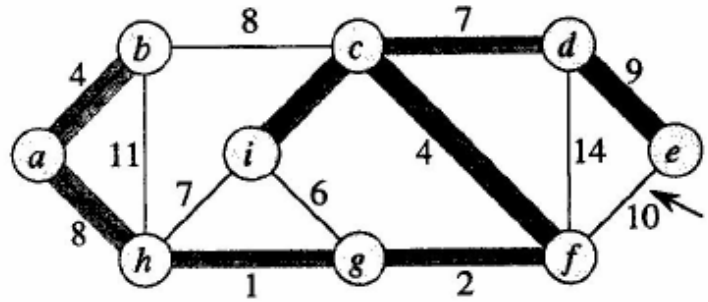


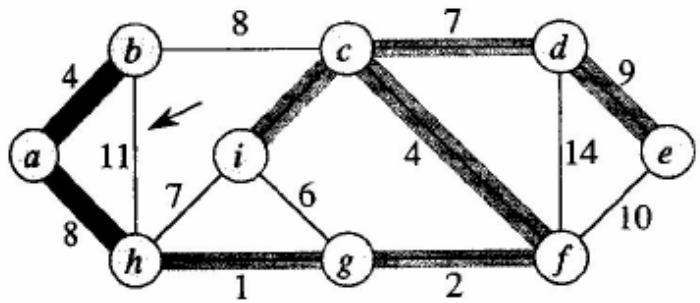
(j)



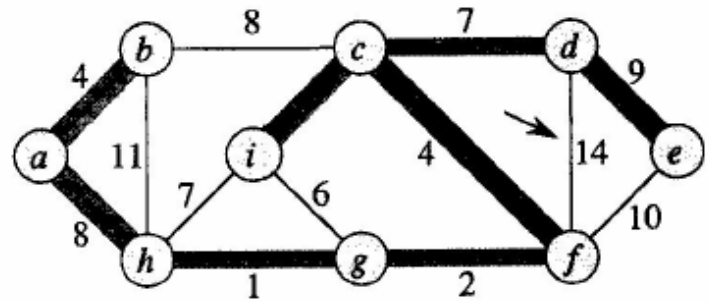


(l)





(n)

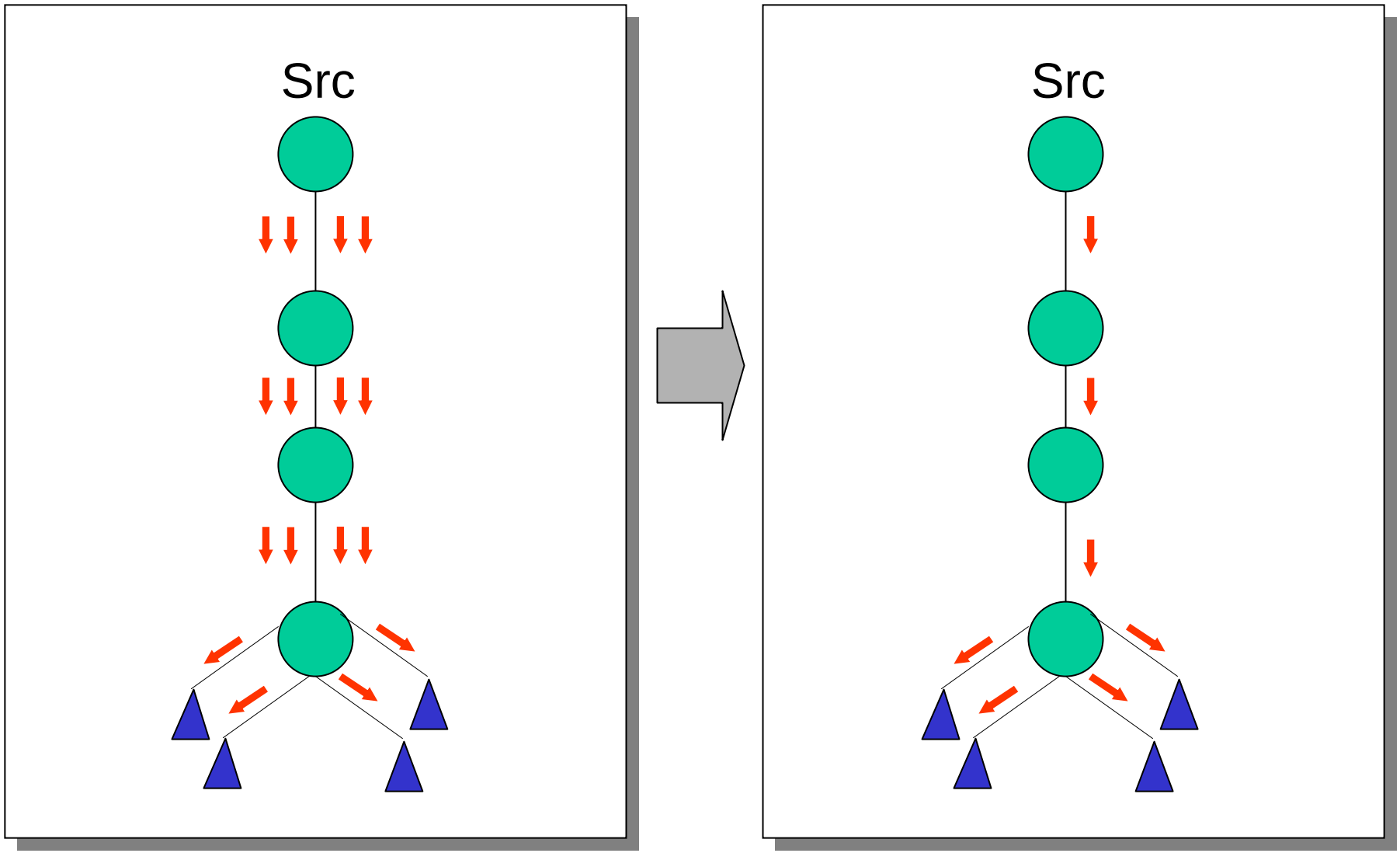


Multicast

Multicast

- Unicast is one-to-one
- Multicast is one-to-many, or many-to-many
- Applications of Multicast
  - Single sender to many receivers
    - Online TV
    - Publish-subscribe
    - Web-cache updates
  - Many senders to many receivers
    - Interactive learning
    - Teleconferencing

Why do we need multicast routing?



IP Multicast

- Multicast Addressing: we need to identify the intended receivers of a multicast, which we call the multicast group
  - Each group has an ID
    - an IP address with a multicast prefix
  - Note that the group is location-independent (i.e., an IP multicast address is a name not an address)
- Multicast Routing: allows routers to learn how to deliver multicast packets and where to duplicate them

IP Multicast Semantics

- Analogy:
  - Each multicast address is like a radio frequency, on which anyone can transmit, and to which anyone can tune-in.
- Sender sends to the multicast IP address
- Receivers can join or leave the multicast group at will
- Routers deliver packets from sender to receivers

Multicast Routing

- Source-based Multicast Tree
- Shared Multicast Tree

Source-Based Multicast

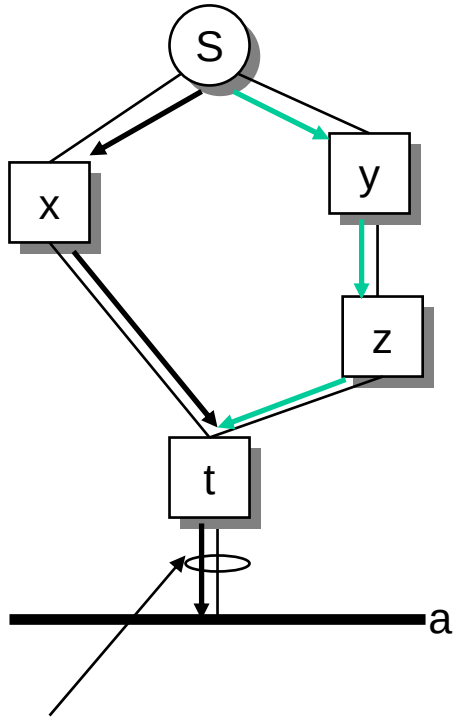
- Consider a single sender
- Routing builds shortest path trees rooted at sender
  - Delivers packets to each receiver along the shortest path
- What if multiple sources are sending to the same group of receivers?
  - Builds multiple source-based trees → inefficient
  - Build one shared tree (will talk about it later)

A protocol that builds source-based trees: Distance-Vector Multicast Routing

- DVMRP is an extension to DV unicast routing
  - Works on top of a conventional distance-vector unicast routing protocol (like RIP)
  - Recall that DV builds a shortest path spanning tree routed at destination
- DVMRP router forwards a multicast packet if
  - The packet arrived from the link used to reach the source of the packet (reverse path forwarding check – RPF)
  - If downstream links have not pruned the tree

Reverse Path Flooding (RPF)

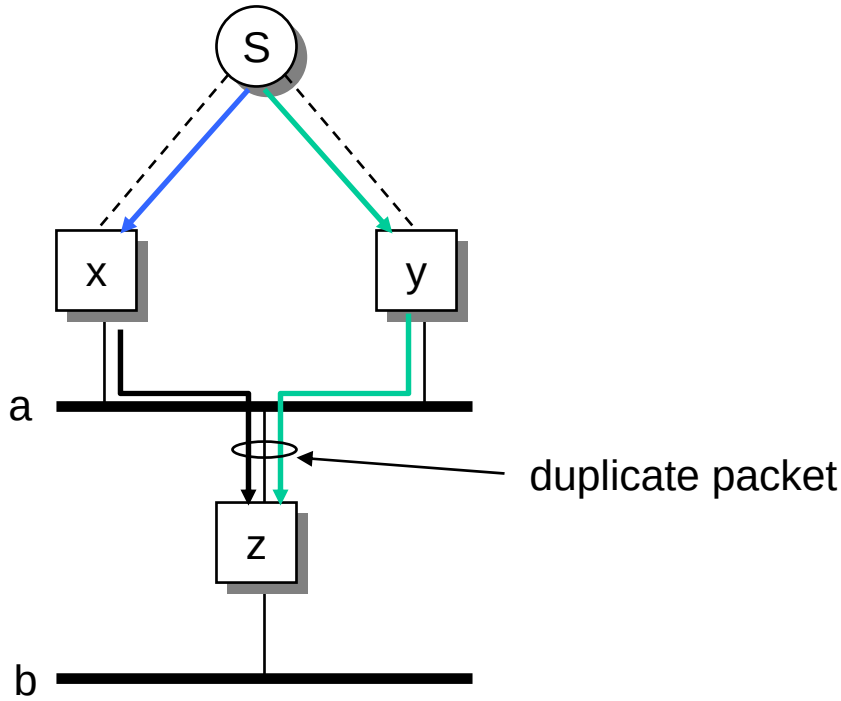
- A router forwards a multicast packet from source (S) iff it arrives via the shortest path from the router back to S
- Packet is replicated out all but the incoming interface
- Reverse shortest paths easy to compute → just use info in DV routing tables
  - DV gives shortest reverse paths
  - Works if costs are symmetric



Forward packets that arrives on shortest path from "t" to "S" (assume symmetric routes)

Problem

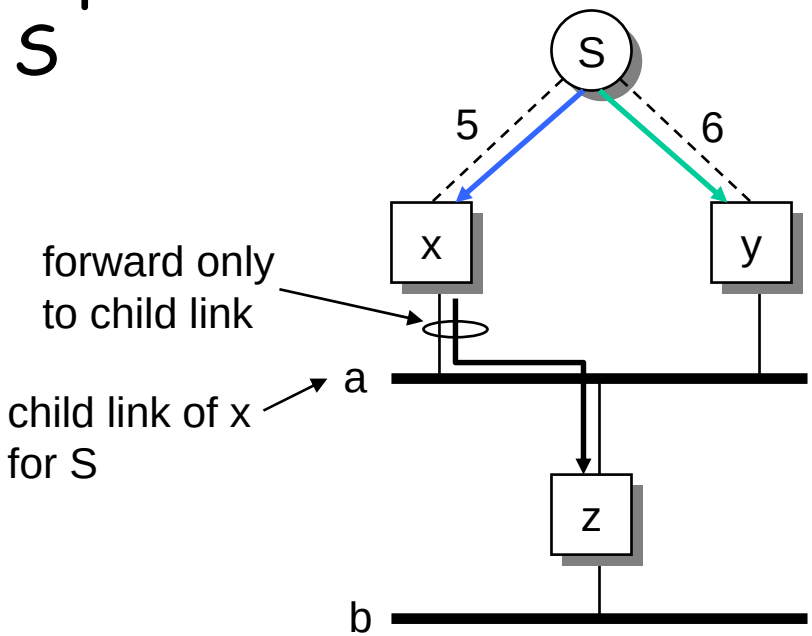
- Flooding can cause a given packet to be sent multiple times over the same link



- Solution: Reverse Path Broadcasting

Reverse Path Broadcasting (RPB)

- Basic idea: forward a packet from S only on child links for S
- Child link of router R for source S: link that has R as parent on the shortest path from the link to S



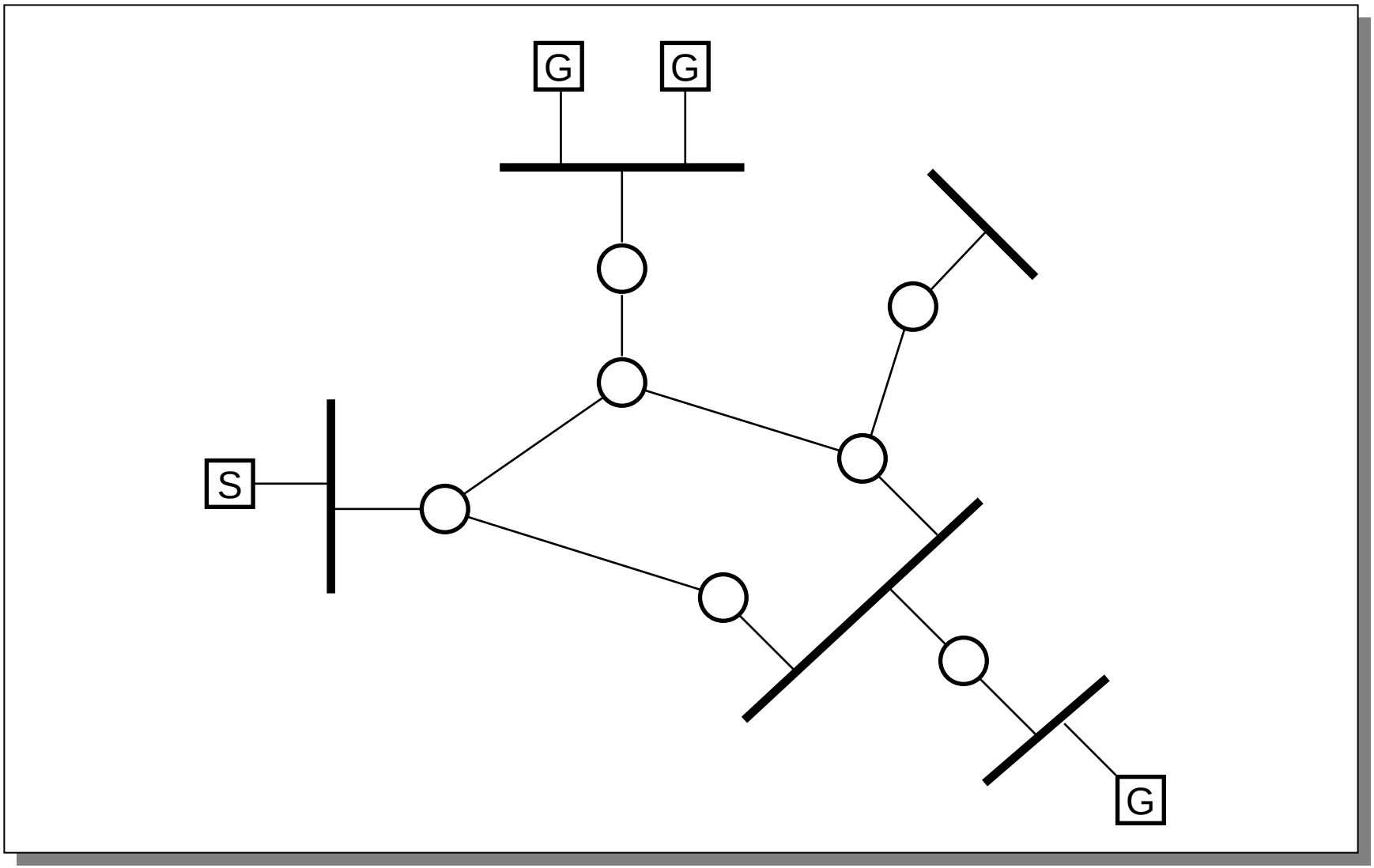
Identify Child Links

- Routing updates identify parent
- Since distances are known (from the unicast routing protocol), each router can easily figure out if it is the parent for a given link
- In case of tie, lower address wins

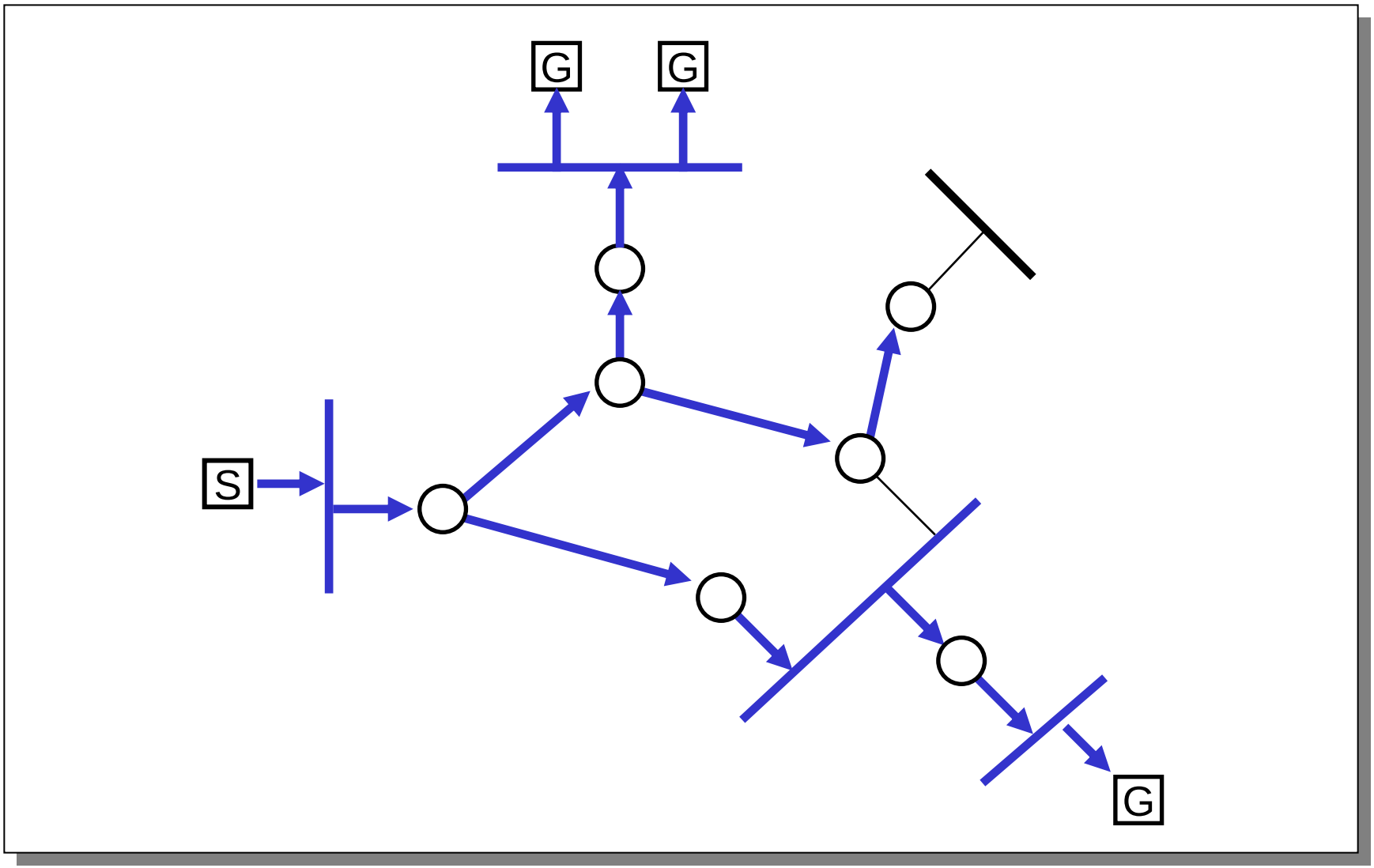
Truncated RBP

- We still need to prevent traffic from flooding the whole network and reaching non-interested hosts
- Don't forward traffic onto network with no receivers
  1. Identify leaves
  2. Detect group membership in leaf
  3. Prune back transmission so that only absolutely necessary links carry traffic

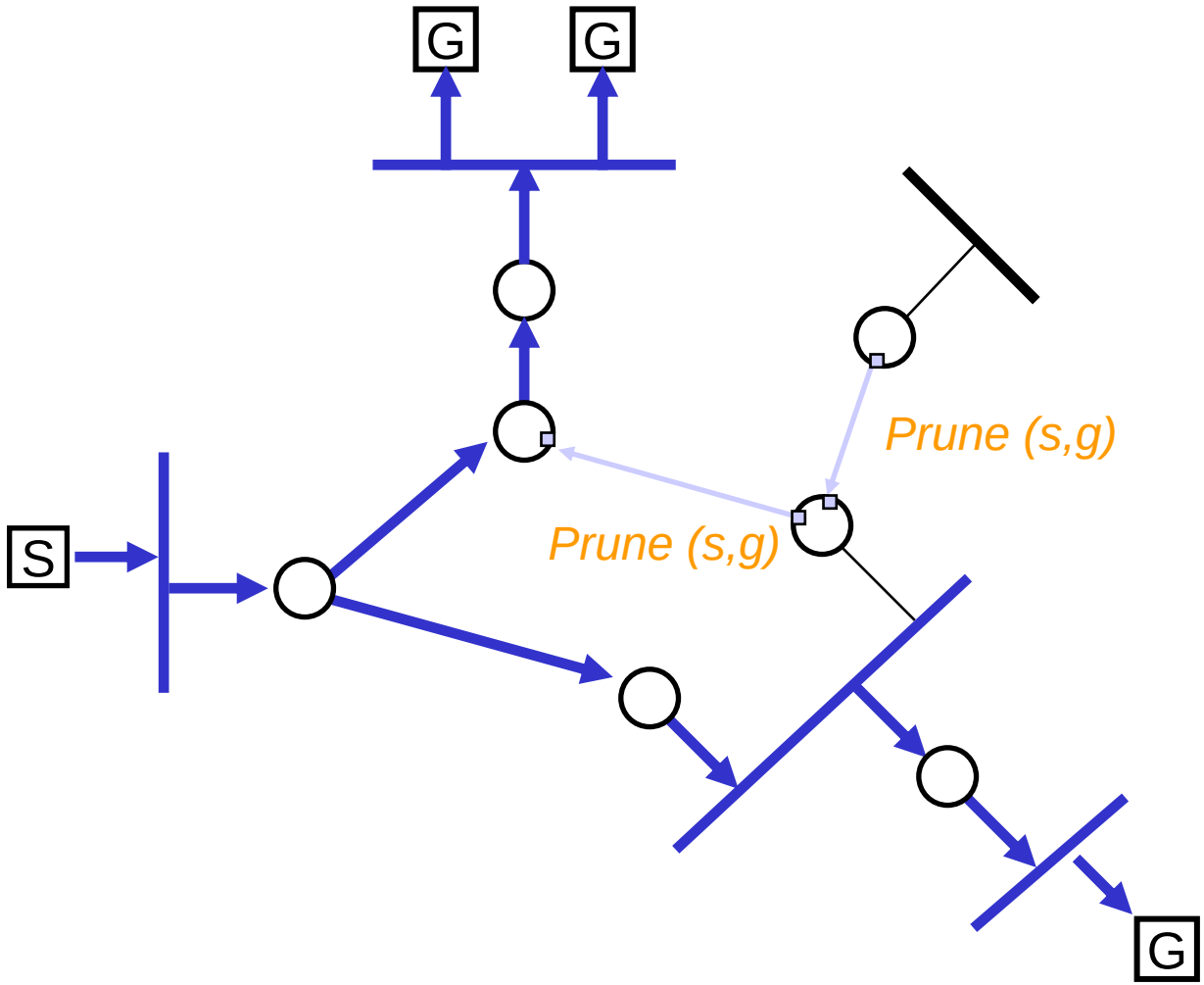
Example Topology



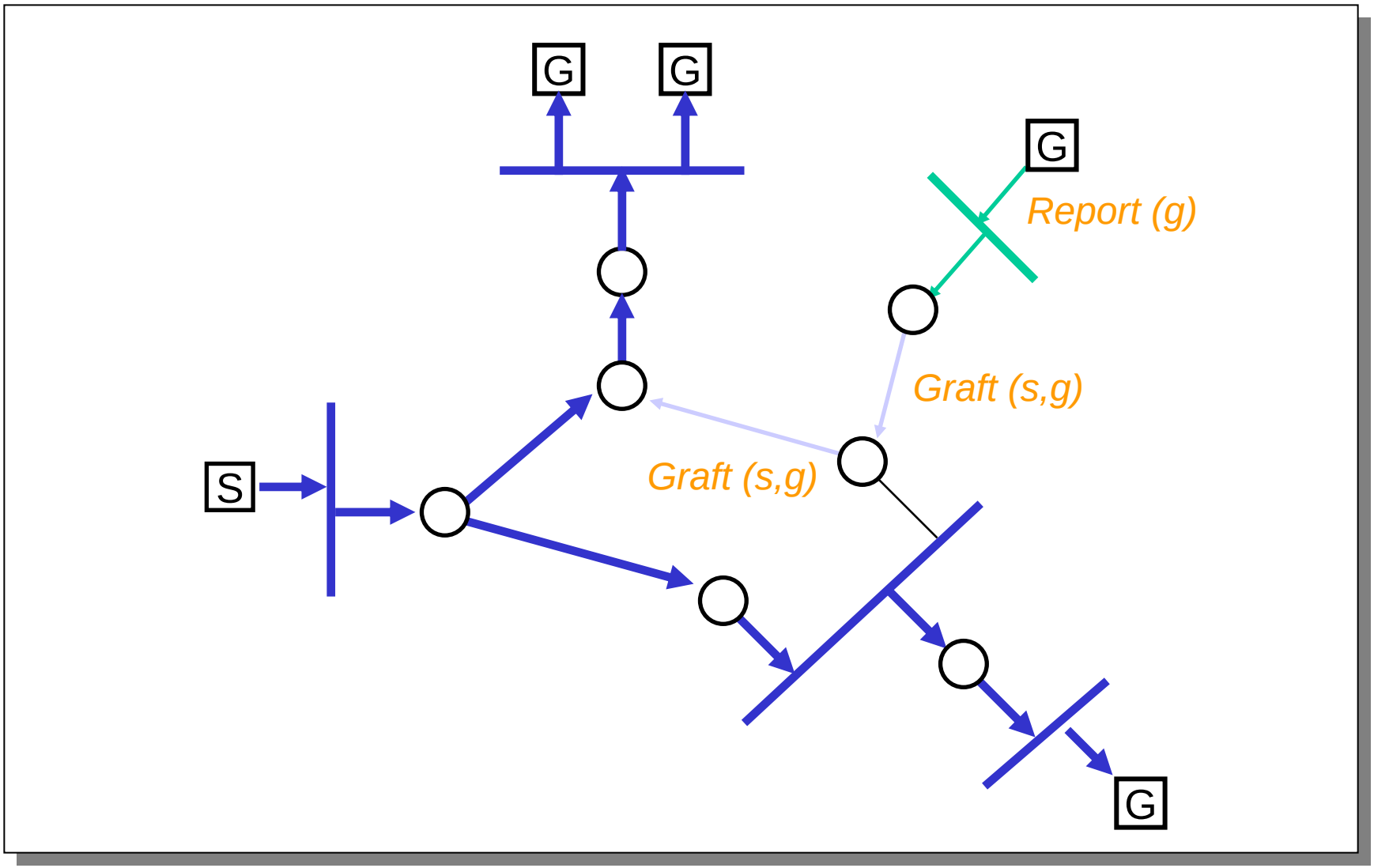
Broadcast with Truncation



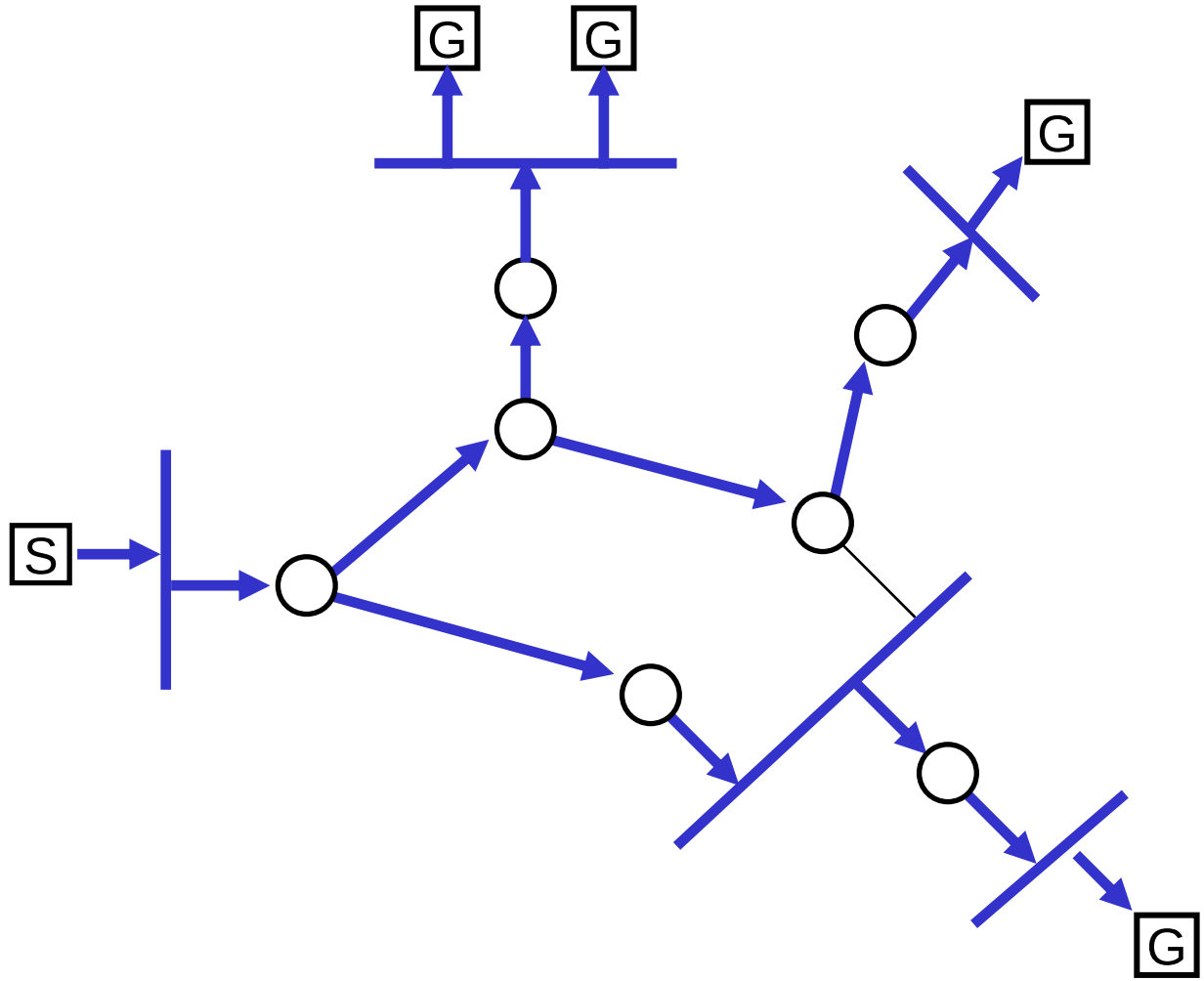
Prune



Graft



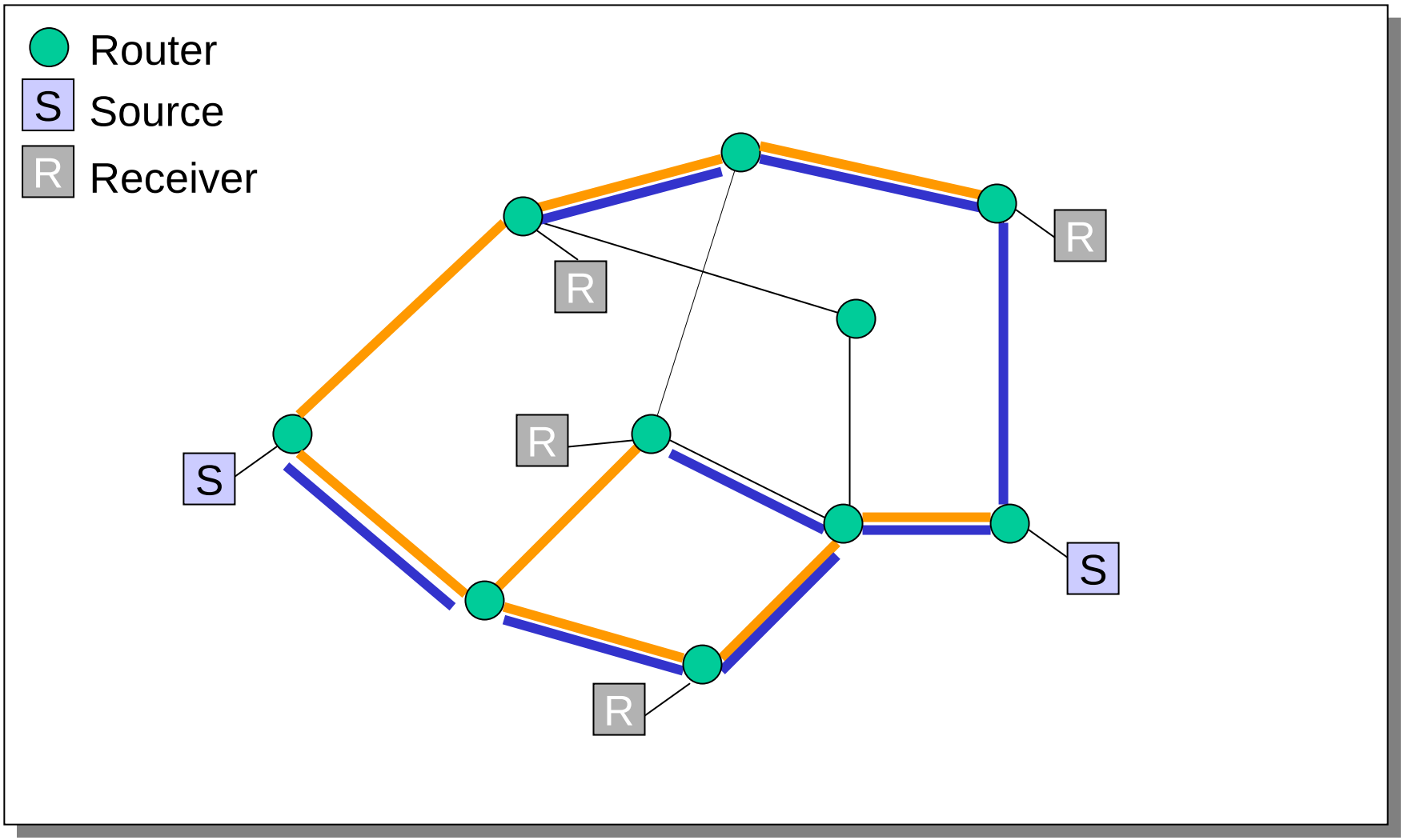
Steady State



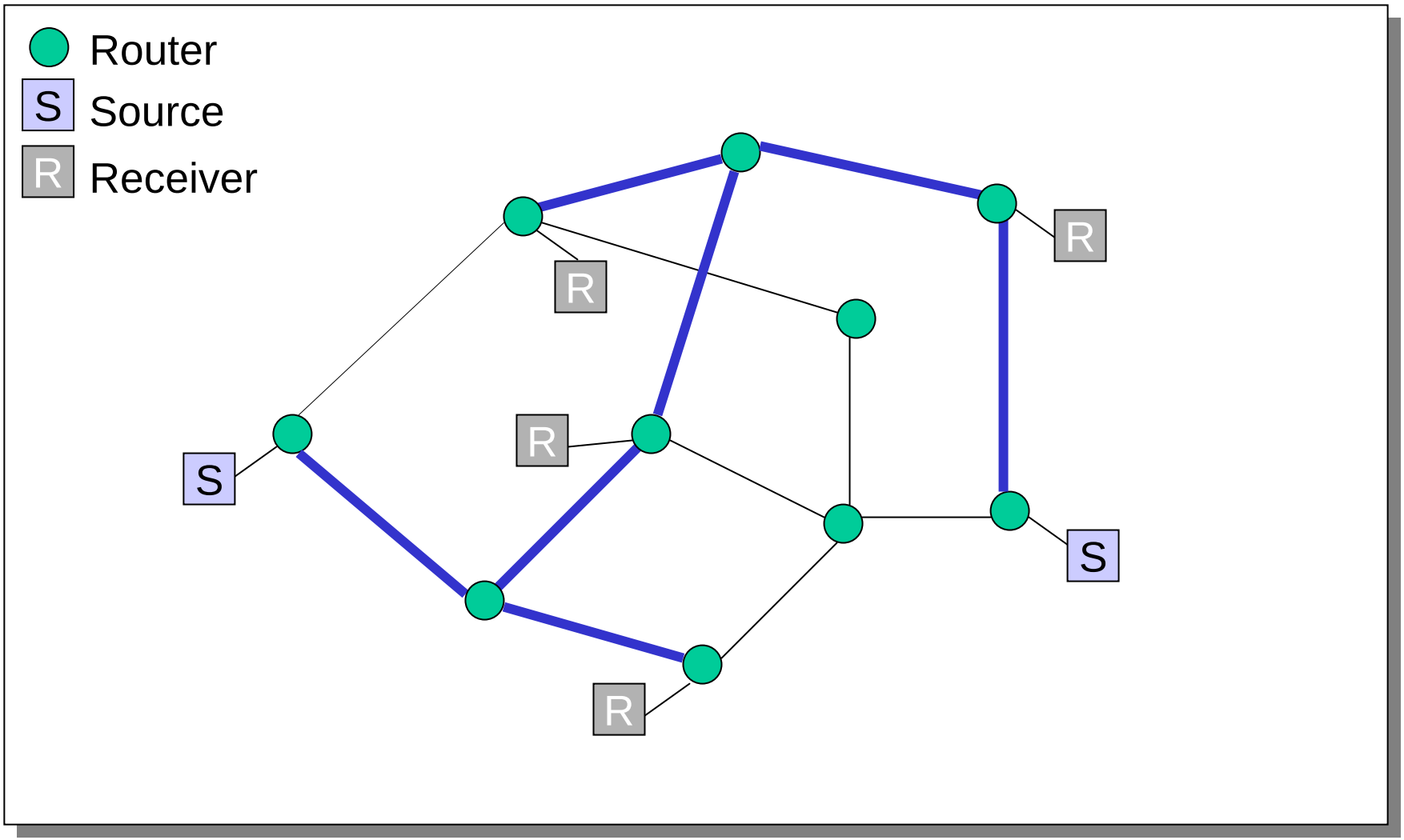
Source vs. Shared Trees

- What if there are many senders who are sending to the same group?
  - Unscalable, routers store per-group*per-sender state
- Can reduce the state stored at routers by building a shared tree that contains all receivers and is used by all senders to deliver traffic to the group

Source-based Trees with 2 Senders



A Shared Tree



Shared Trees

- What shared tree should we build?
- The optimal shared tree is a Steiner tree

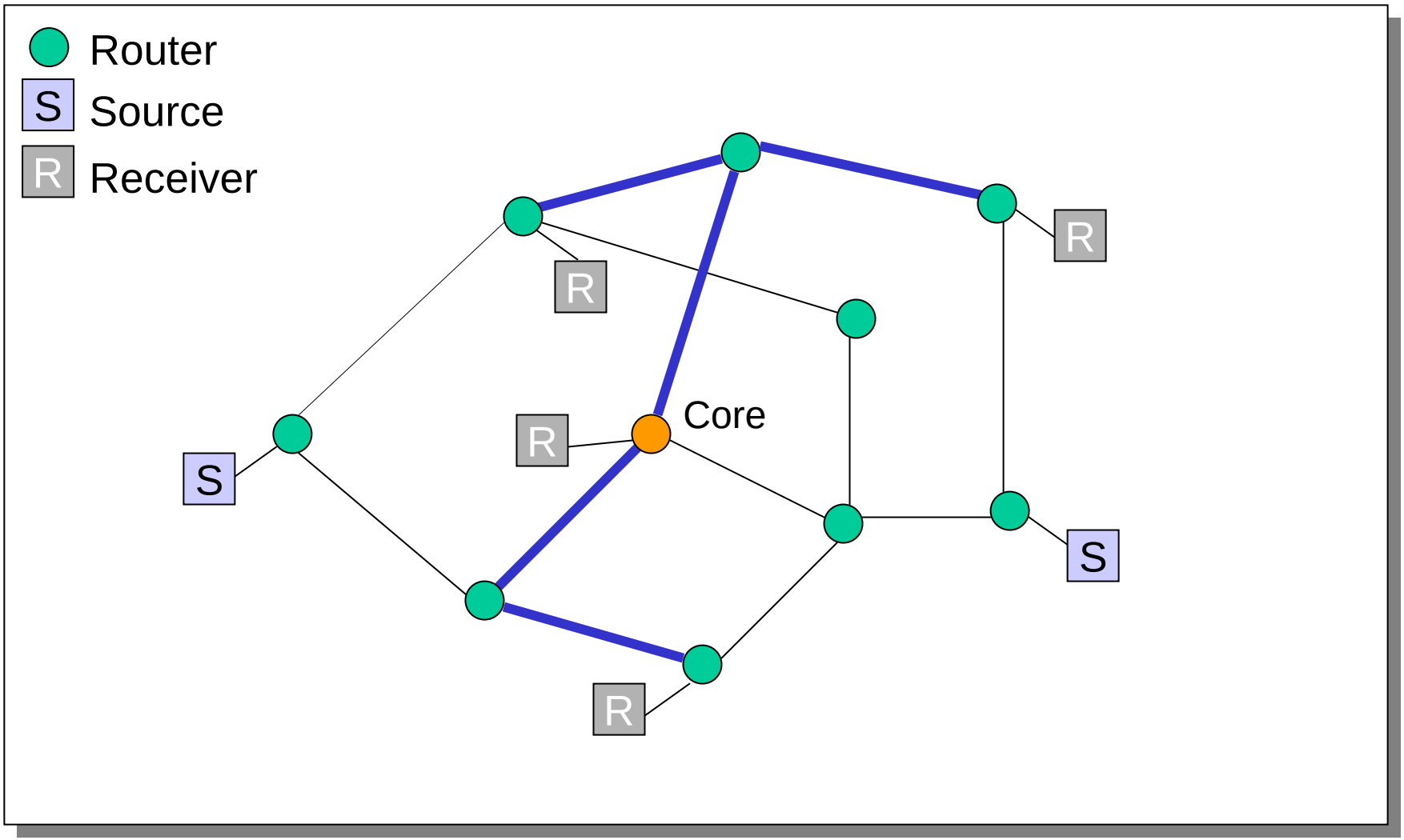
Steiner Tree

- G=(V, E) is undirected graph. A positive, real cost assigned to each edge in E. Given a subset of nodes S in V, the Steiner tree is the minimum cost tree that connects all nodes in S
- Proved to be NP-complete (no polynomial time solution)
  - Excellent heuristics
- Not used in practice because of complexity

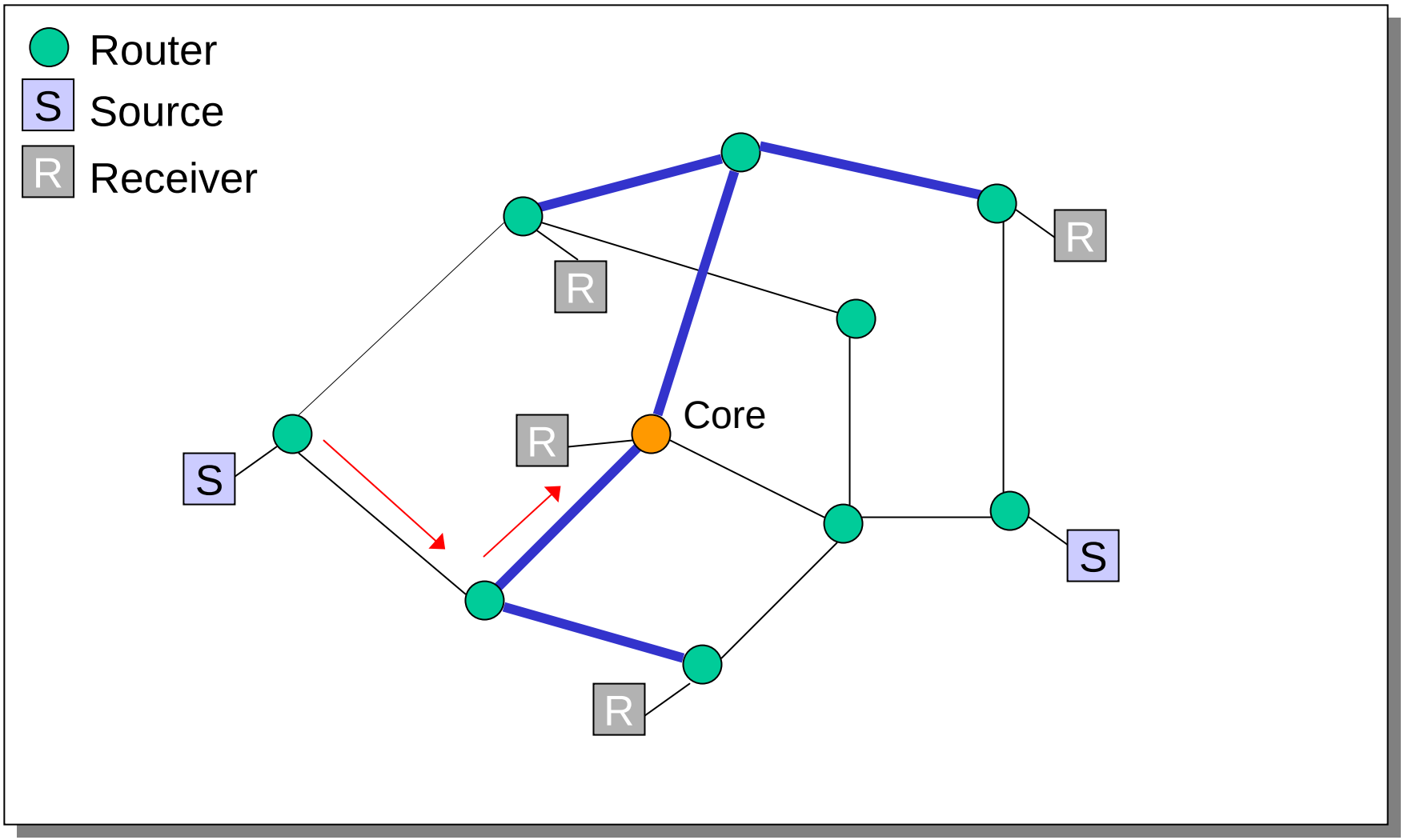
One Approach to Shared Trees: Core-Based Shared Multicast Trees (CBT)

- Build a shortest path tree routed at a core router
- Unicast packets to core and bounce them back to the multicast group
- Tree construction is receiver-based
  - One tree per group
  - Only nodes on tree involved
- Reduce routing table state from O(S x G) to O(G)

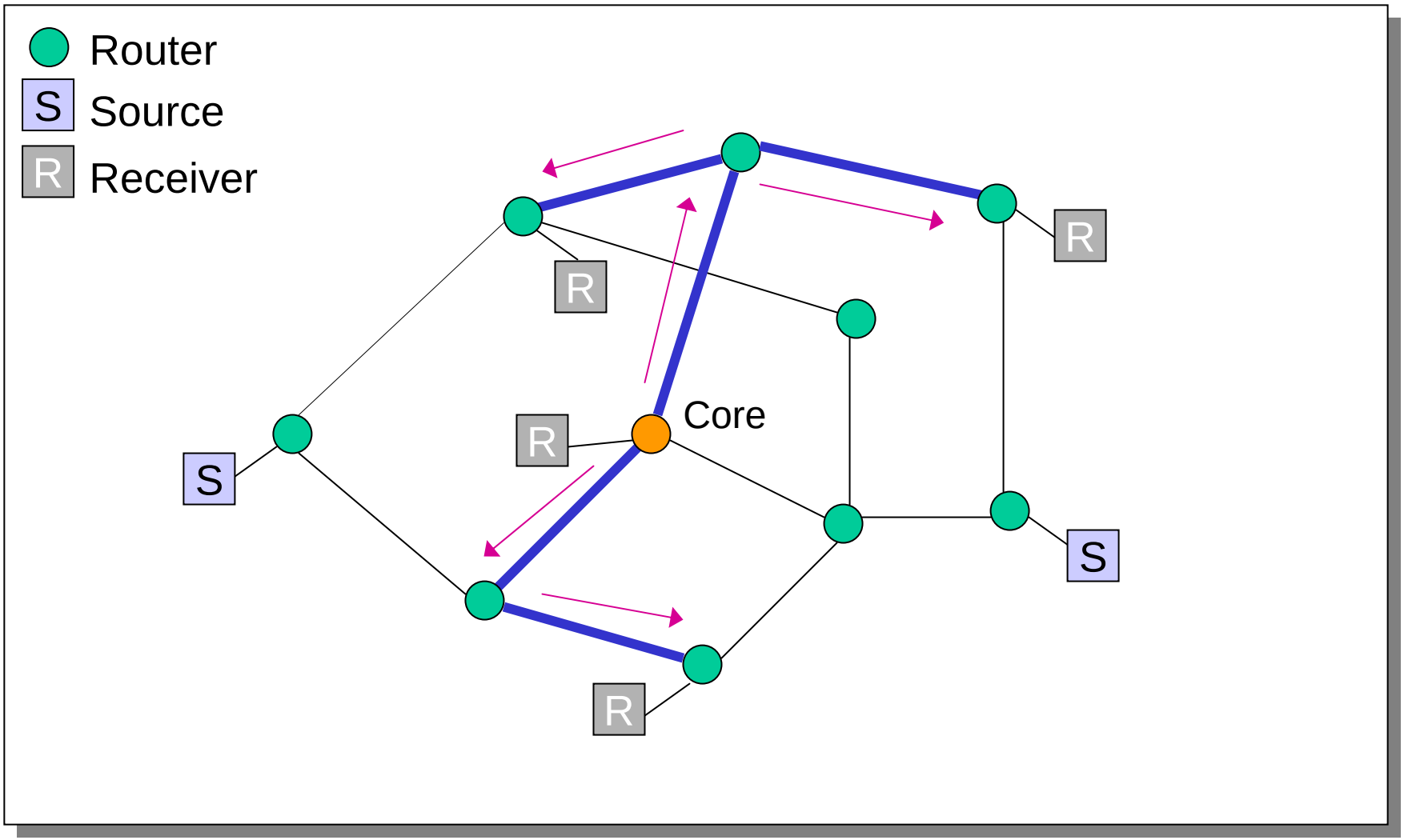
A Core-based Shared Tree



A Shared Tree: Sender encapsulates and unicasts packets to core



A Shared Tree: Core decapsulates and multicasts packets on the tree



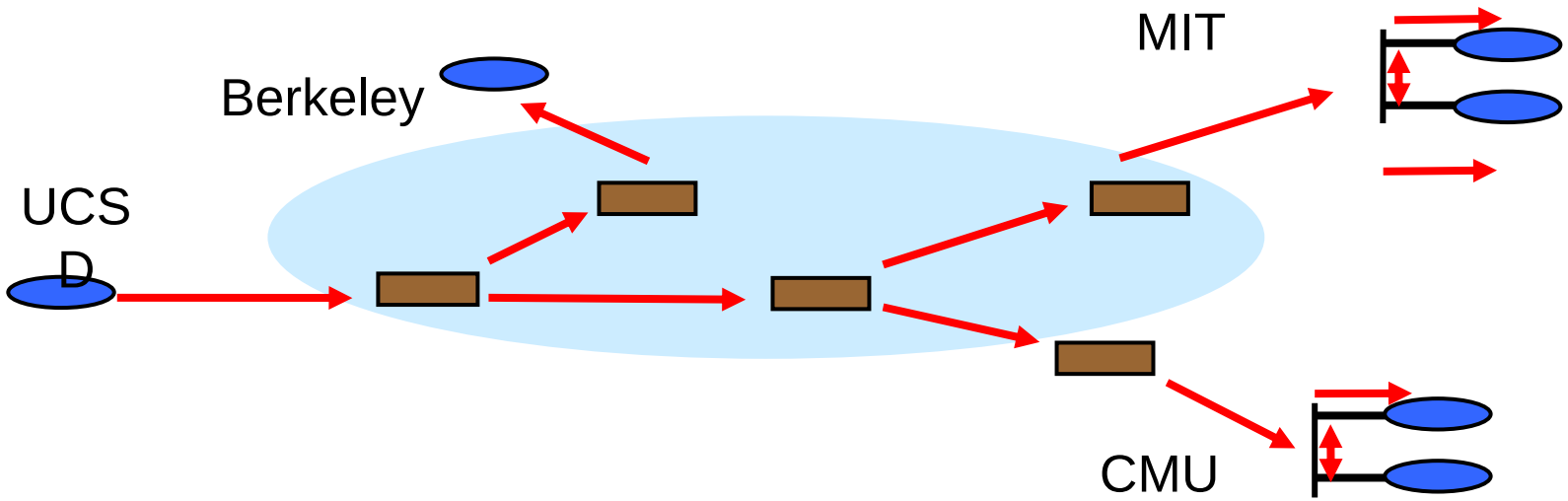
Disadvantages

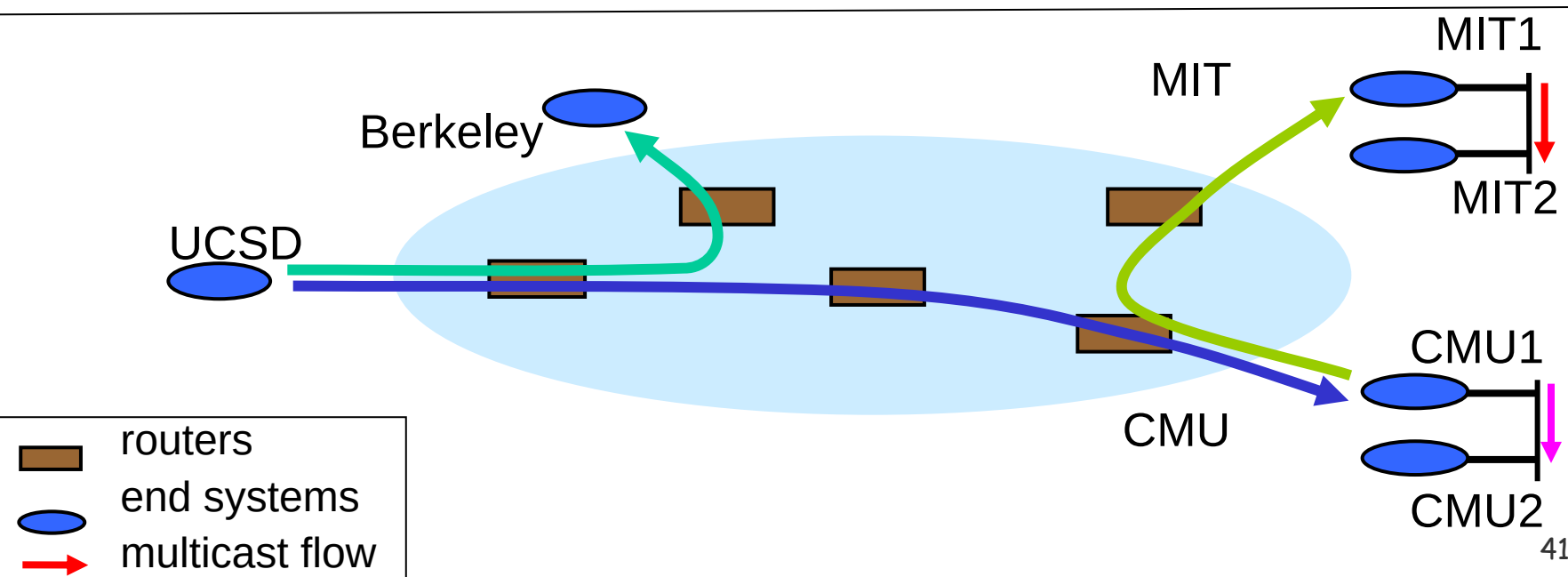
- Sub-optimal delay
- Single point of failure
  - Core goes out → everything is lost until error recovery elects a new core
- Small, local groups with non-local core
  - Need good core selection
  - Optimal choice of core (computing topological center) is NP complete

Status of IP Multicast

- IP multicast is not deployed (very limited deployment only in academic networks)
- Why? Possible explanations
  - Violates ISP input-rate-based billing model
    - No incentive for ISPs to enable multicast!
  - Not really scalable
  - Allows end user to insert state in routers
    - Not secure
    - Does not make ISP happy

Application Level Multicast





Application-Level Multicast

- Create a multicast tree using an overlay
  - How to organize the multicast tree?
- Advantage
  - Quick deployment
  - All multicast state in end systems
  - Computation at forwarding points simplifies support for higher level functionality
- Disadvantages
  - Performance concerns compared to IP Multicast
    - Increase in delay
    - Bandwidth inefficiency (packet duplication)