

Hierarchical Routing

Our routing study thus far - idealization

- ❖ all routers identical
 - ❖ network “flat”
- ... *not* true in practice

scale: with 200 million destinations:

- ❖ can't store all destinations in routing tables!
- ❖ routing table exchange would swamp links!

administrative autonomy

- ❖ internet = network of networks
- ❖ each network admin may want to control routing in its own network

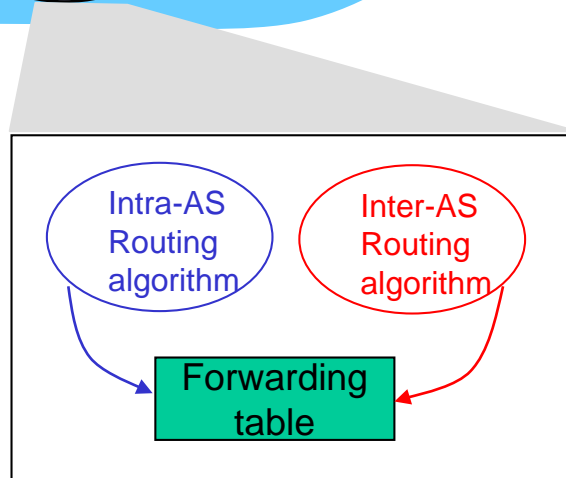
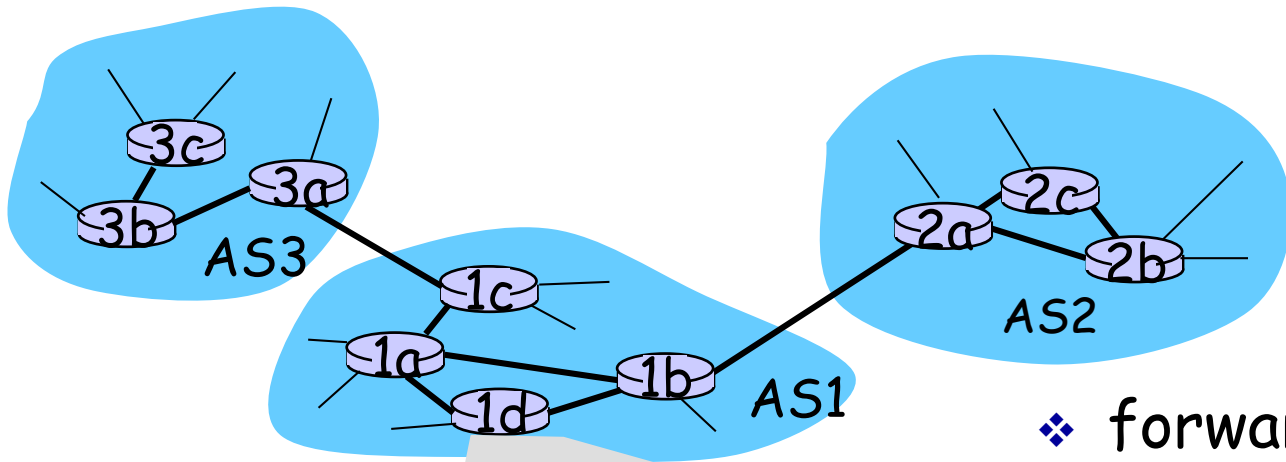
Hierarchical Routing

- ❖ aggregate routers into regions, "autonomous systems" (AS)
- ❖ routers in same AS run same routing protocol
 - "intra-AS" routing protocol
 - routers in different AS can run different intra-AS routing protocol

gateway router

- ❖ at "edge" of its own AS
- ❖ has link to router in another AS

Interconnected ASes



- ❖ forwarding table configured by both intra- and inter-AS routing algorithm
 - intra-AS sets entries for internal dests
 - inter-AS & intra-AS sets entries for external dests

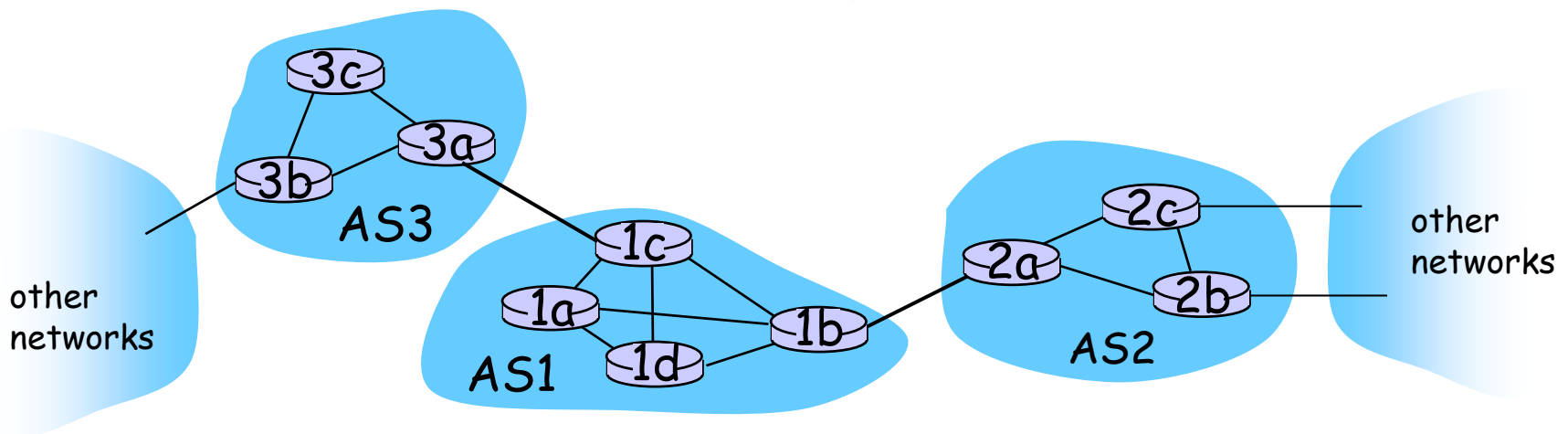
Inter-AS tasks

- ❖ suppose router in AS1 receives datagram destined outside of AS1:
 - router should forward packet to gateway router, but which one?

AS1 must:

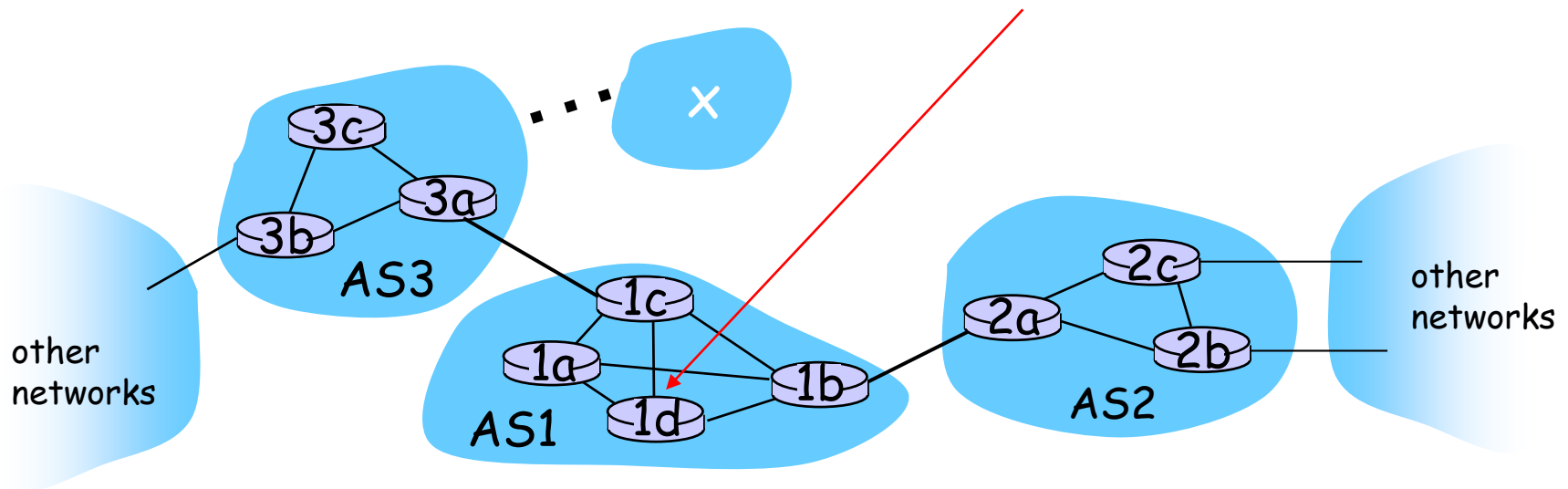
1. learn which dests are reachable through AS2, which through AS3
2. propagate this reachability info to all routers in AS1

job of inter-AS routing!



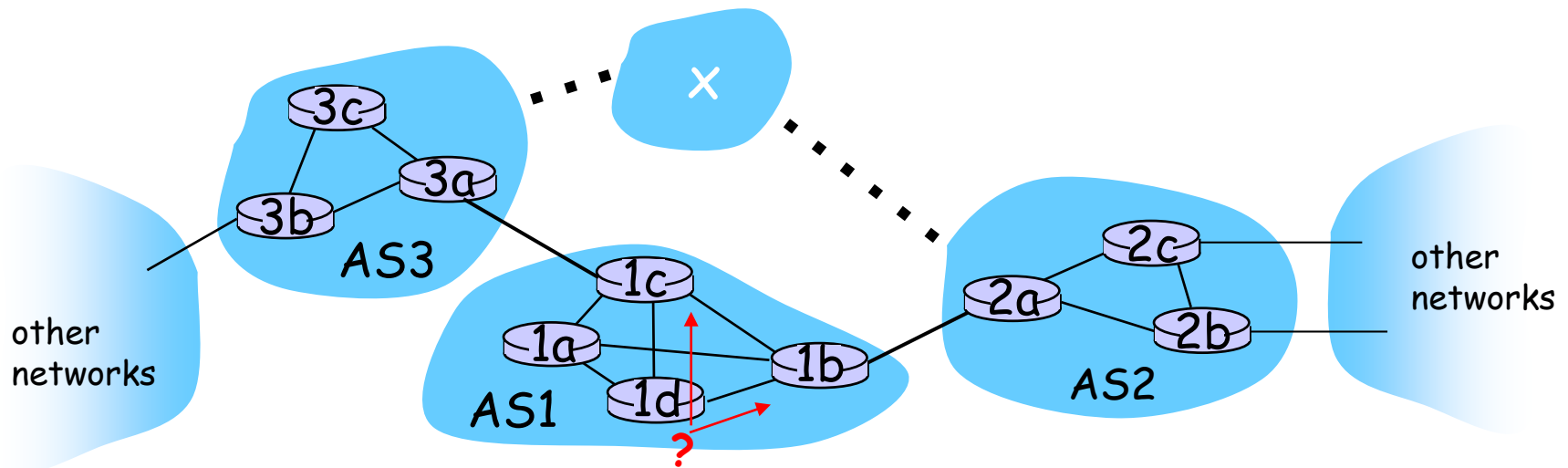
Example: Setting forwarding table in router 1d

- ❖ suppose AS1 learns (via inter-AS protocol) that subnet x reachable via AS3 (gateway 1c) but not via AS2.
 - inter-AS protocol propagates reachability info to all internal routers
- ❖ router 1d determines from intra-AS routing info that its interface I is on the least cost path to 1c.
 - installs forwarding table entry (x, I)



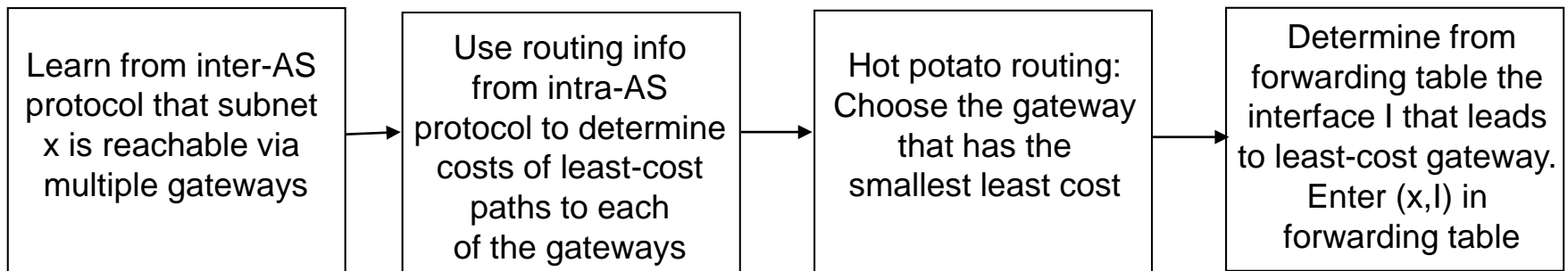
Example: Choosing among multiple ASes

- ❖ now suppose AS1 learns from inter-AS protocol that subnet **x** is reachable from AS3 and from AS2.
- ❖ to configure forwarding table, router 1d must determine which gateway it should forward packets towards for dest **x**
 - this is also job of inter-AS routing protocol!



Example: Choosing among multiple ASes

- ❖ now suppose AS1 learns from inter-AS protocol that subnet x is reachable from AS3 and from AS2.
- ❖ to configure forwarding table, router 1d must determine towards which gateway it should forward packets for dest x .
 - this is also job of inter-AS routing protocol!
- ❖ **hot potato routing**: send packet towards closest of two routers.



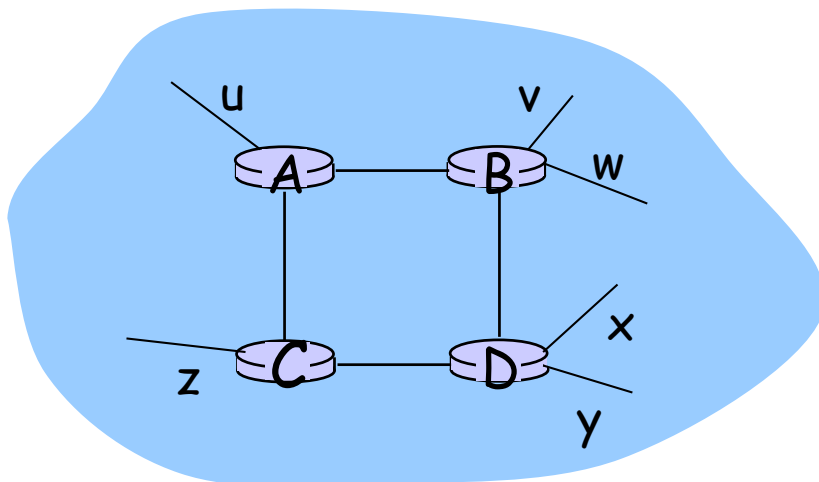
Intra-AS Routing

- ❖ also known as **Interior Gateway Protocols (IGP)**
- ❖ most common Intra-AS routing protocols:
 - RIP: Routing Information Protocol
 - OSPF: Open Shortest Path First
 - IGRP: Interior Gateway Routing Protocol (Cisco proprietary)

RIP (Routing Information Protocol)

❖ distance vector algorithm

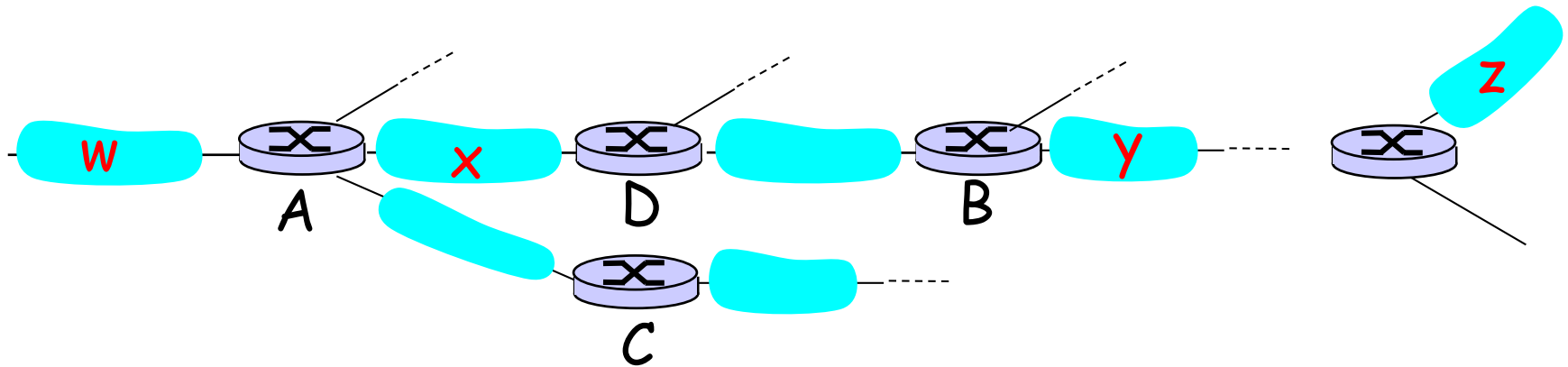
- distance metric: # hops (max = 15 hops), each link has cost 1
- DVs exchanged with neighbors every 30 sec in response message (aka **advertisement**)
- each advertisement: list of up to 25 destination **subnets** (in IP addressing sense)



from router A to destination **subnets**:

<u>subnet</u>	<u>hops</u>
u	1
v	2
w	2
x	3
y	3
z	2

RIP: Example



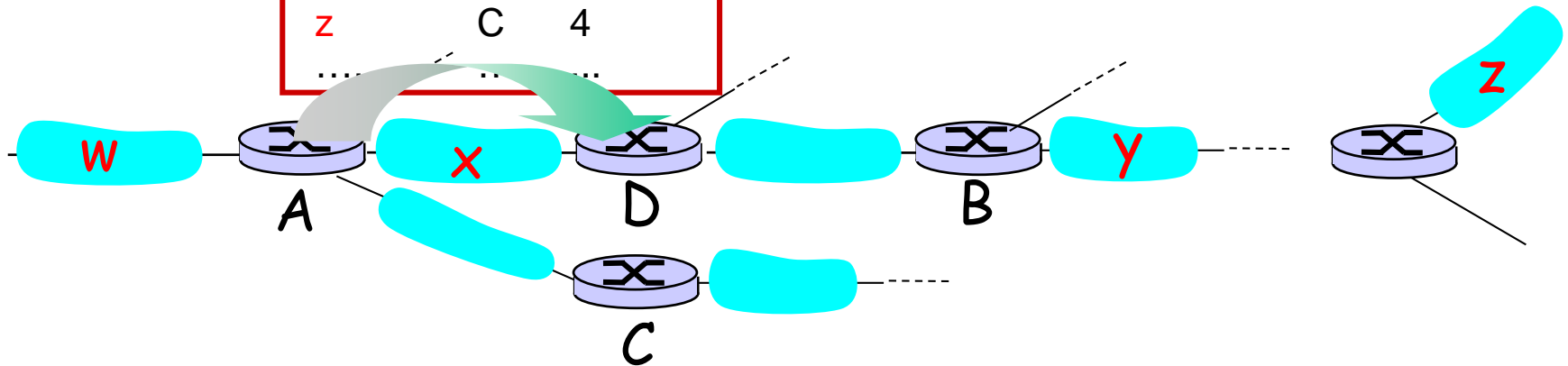
routing table in router D

destination subnet	next router	# hops to dest
W	A	2
y	B	2
Z	B	7
X	--	1
....

RIP: Example

A-to-D advertisement

dest	next	hops
W	-	1
X	-	1
Z	C	4
....



routing table in router D

destination subnet	next router	# hops to dest
W	A	2
y	B	2
Z	B → A	7 → 5
X	--	1
....

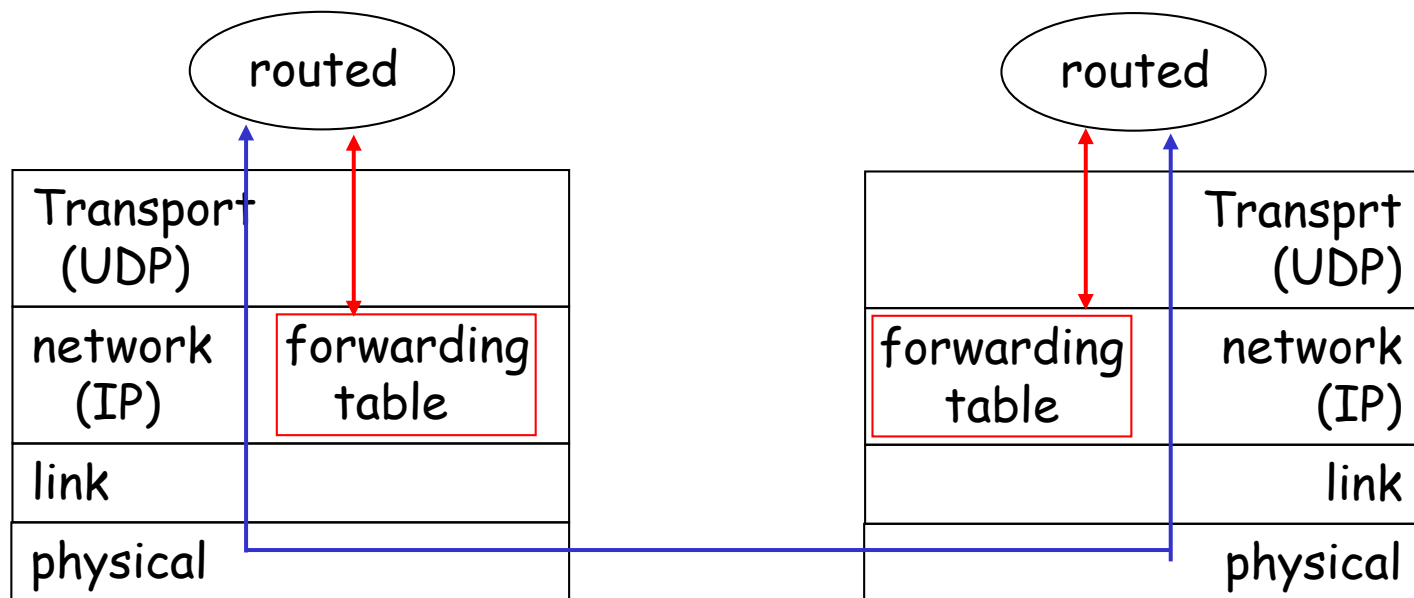
RIP: Link Failure and Recovery

If no advertisement heard after 180 sec -->
neighbor/link declared dead

- routes via neighbor invalidated
- new advertisements sent to neighbors
- neighbors in turn send out new advertisements (if tables changed)
- link failure info quickly (?) propagates to entire net
- *poison reverse* used to prevent ping-pong loops (infinite distance = 16 hops)

RIP Table processing

- ❖ RIP routing tables managed by **application-level** process called route-d (daemon)
- ❖ advertisements sent in UDP packets, periodically repeated



OSPF (Open Shortest Path First)

- ❖ “open”: publicly available
- ❖ uses Link State algorithm
 - LS packet dissemination
 - topology map at each node
 - route computation using Dijkstra's algorithm
- ❖ OSPF advertisement carries one entry per neighbor router
- ❖ advertisements disseminated to **entire** AS (via flooding)
 - carried in OSPF messages directly over IP (rather than TCP or UDP)

OSPF "advanced" features (not in RIP)

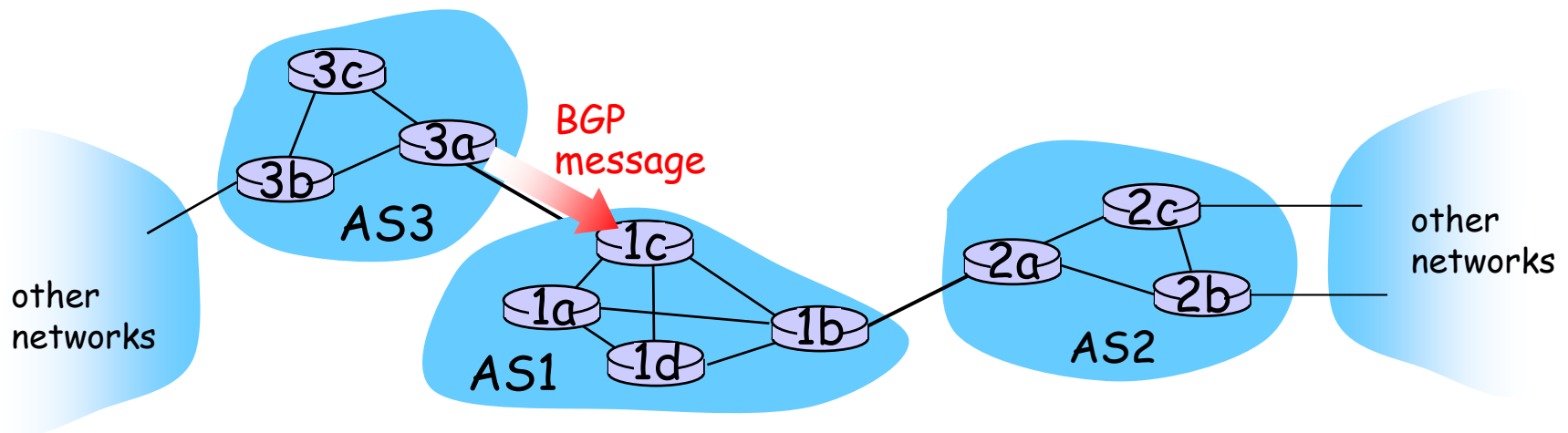
- ❖ **security**: all OSPF messages authenticated (to prevent malicious intrusion)
- ❖ **multiple** same-cost **paths** allowed (only one path in RIP)
- ❖ for each link, multiple cost metrics for different **TOS (Type of Service)** (e.g., satellite link cost set "low" for best effort ToS; high for real time ToS)
- ❖ integrated uni- and **multicast** support:
 - Multicast OSPF (MOSPF) uses same topology data base as OSPF

Internet inter-AS routing: BGP

- ❖ **BGP (Border Gateway Protocol):** the de facto inter-domain routing protocol
 - “glue that holds the Internet together”
- ❖ BGP provides each AS a means to:
 - **eBGP:** obtain subnet reachability information from neighboring ASs.
 - **iBGP:** propagate reachability information to all AS-internal routers.
 - determine “good” routes to other networks based on reachability information and policy.
- ❖ allows subnet to advertise its existence to rest of Internet: *“I am here”*

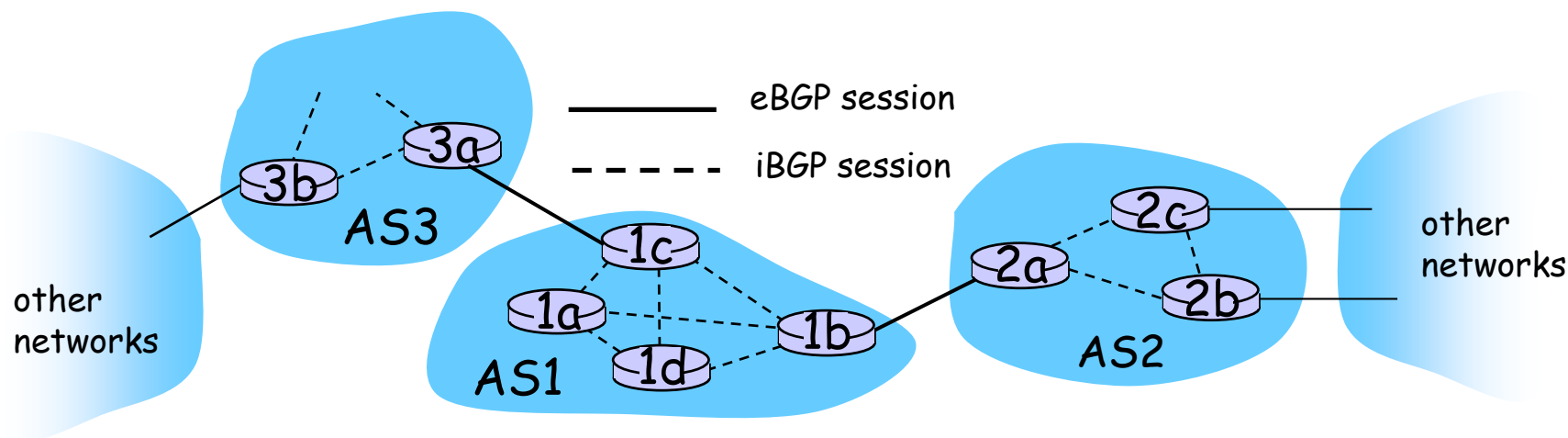
BGP basics

- ❖ **BGP session:** two BGP routers (“peers”) exchange BGP messages:
 - advertising **paths** to different destination network prefixes (“path vector” protocol)
 - exchanged over semi-permanent TCP connections
- ❖ when AS3 advertises a prefix to AS1:
 - AS3 **promises** it will forward datagrams towards that prefix
 - AS3 can aggregate prefixes in its advertisement



BGP basics: distributing path information

- ❖ using eBGP session between 3a and 1c, AS3 sends prefix reachability info to AS1.
 - 1c can then use iBGP to distribute new prefix info to all routers in AS1
 - 1b can then re-advertise new reachability info to AS2 over 1b-to-2a eBGP session
- ❖ when router learns of new prefix, it creates entry for prefix in its forwarding table.



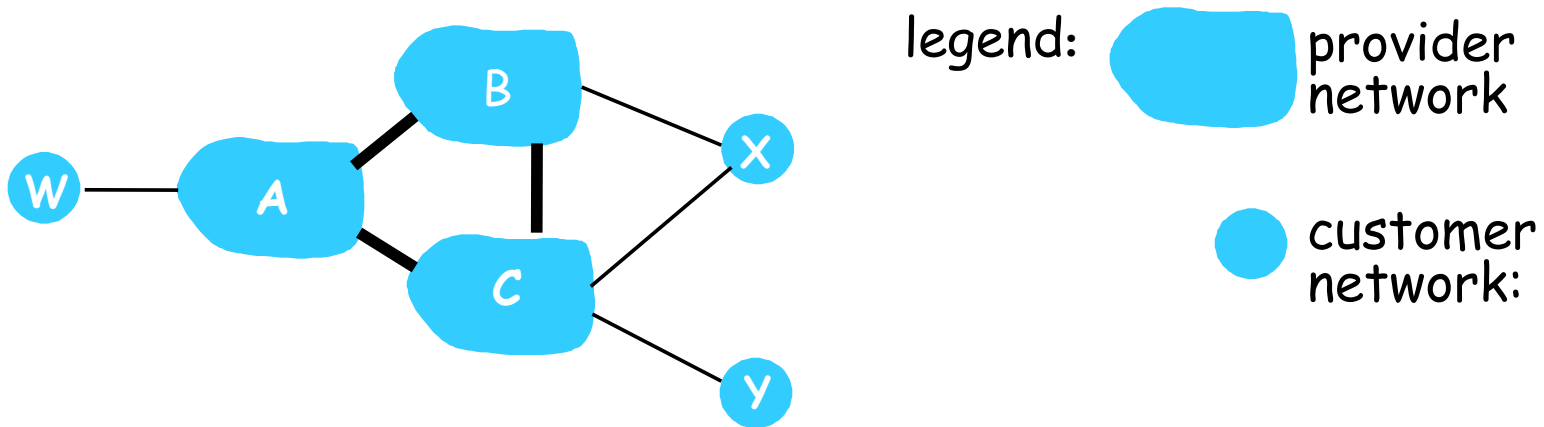
Path attributes & BGP routes

- ❖ advertised prefix includes BGP attributes
 - prefix + attributes = "route"
- ❖ two important attributes:
 - **AS-PATH**: contains ASs through which prefix advertisement has passed: e.g., AS 67, AS 17
 - **NEXT-HOP**: indicates specific internal-AS router to next-hop AS. (may be multiple links from current AS to next-hop-AS)
- ❖ gateway router receiving route advertisement uses **import policy** to accept/decline
 - e.g., never route through AS x
 - **policy-based** routing

BGP route selection

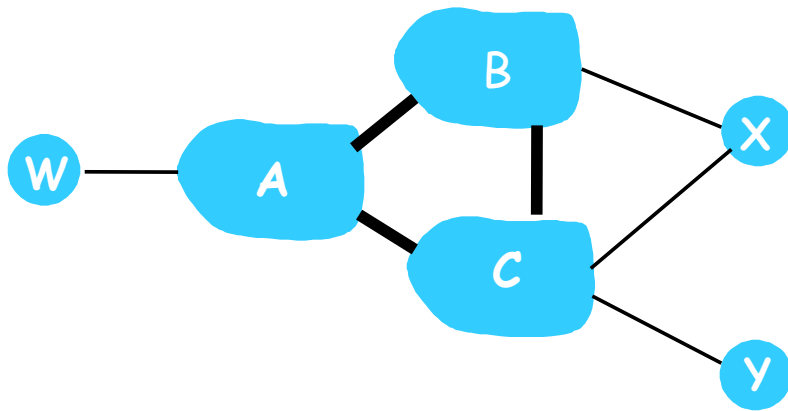
- ❖ router may learn about more than one route to destination AS, selects route based on:
 1. local preference value attribute: policy decision
 2. shortest AS-PATH
 3. closest NEXT-HOP router: hot potato routing
 4. additional criteria



BGP routing policy



- ❖ A,B,C are **provider networks**
- ❖ X,W,Y are customer (of provider networks)
- ❖ X is **dual-homed**: attached to two networks
 - X does not want to route from B via X to C
 - .. so X will not advertise to B a route to C

BGP routing policy (2)



legend:  provider network
 customer network:

- ❖ A advertises path AW to B
- ❖ B advertises path BAW to X
- ❖ Should B advertise path BAW to C?
 - No way! B gets no "revenue" for routing $CBAW$ since neither W nor C are B's customers
 - B wants to force C to route to w via A
 - B wants to route *only* to/from its customers!

Why different Intra- and Inter-AS routing ?

Policy:

- ❖ Inter-AS: admin wants control over how its traffic routed, who routes through its net.
- ❖ Intra-AS: single admin, so no policy decisions needed

Scale:

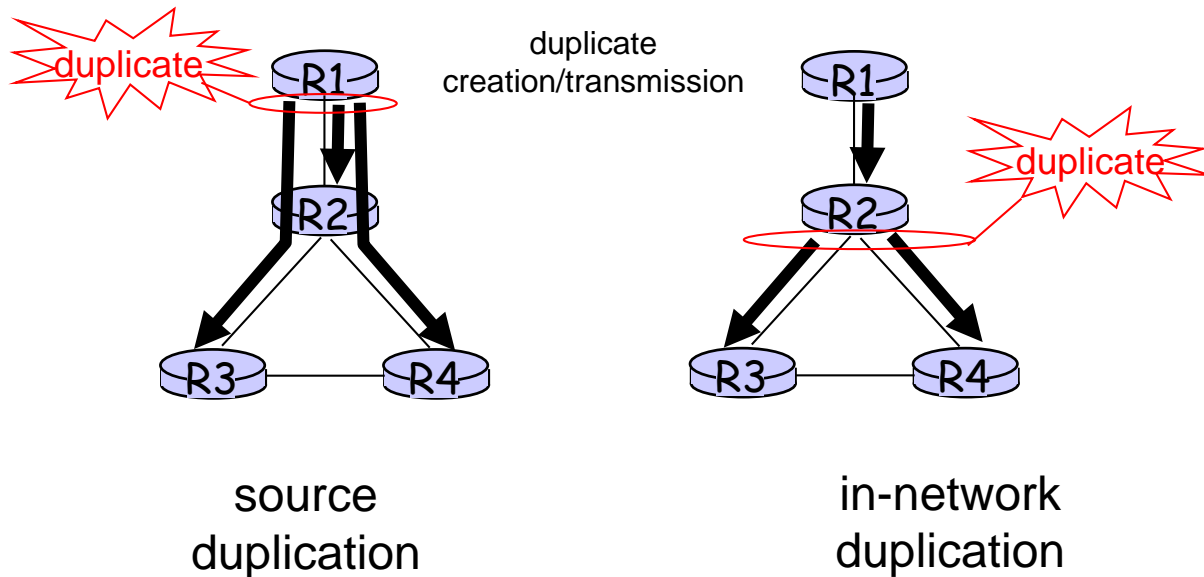
- ❖ hierarchical routing saves table size, reduced update traffic

Performance:

- ❖ Intra-AS: can focus on performance
- ❖ Inter-AS: policy may dominate over performance

Broadcast Routing

- ❖ deliver packets from source to all other nodes
- ❖ source duplication is inefficient:



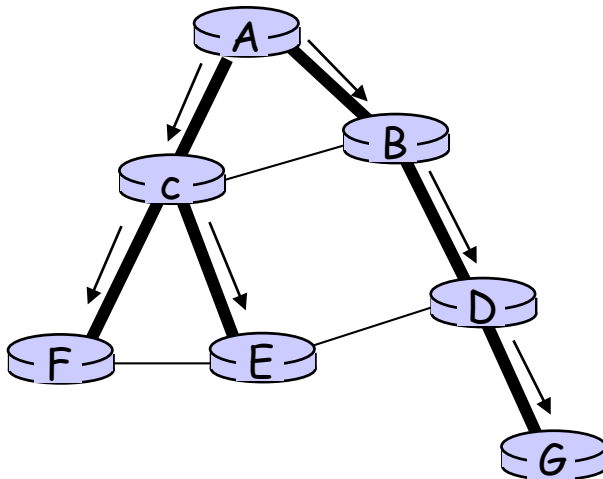
- ❖ source duplication: how does source determine recipient addresses?

In-network duplication

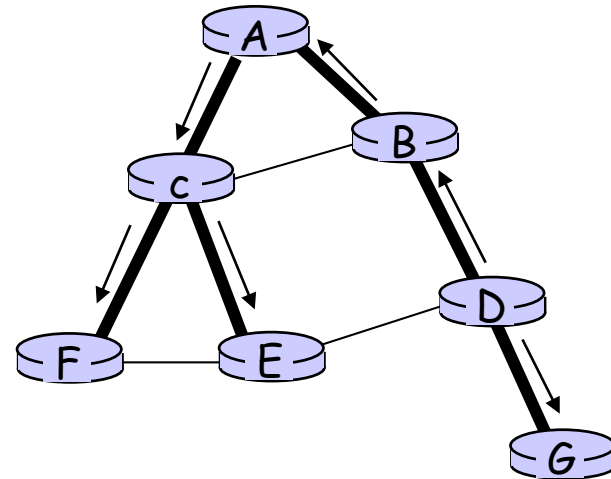
- ❖ flooding: when node receives broadcast packet, sends copy to all neighbors
 - problems: cycles & broadcast storm
- ❖ controlled flooding: node only broadcasts packet if it hasn't broadcast same packet before
 - node keeps track of packet its already broadcasted
 - or reverse path forwarding (RPF): only forward packet if it arrived on shortest path between node and source
- ❖ spanning tree
 - No redundant packets received by any node

Spanning Tree

- ❖ First construct a spanning tree
- ❖ Nodes forward copies only along spanning tree



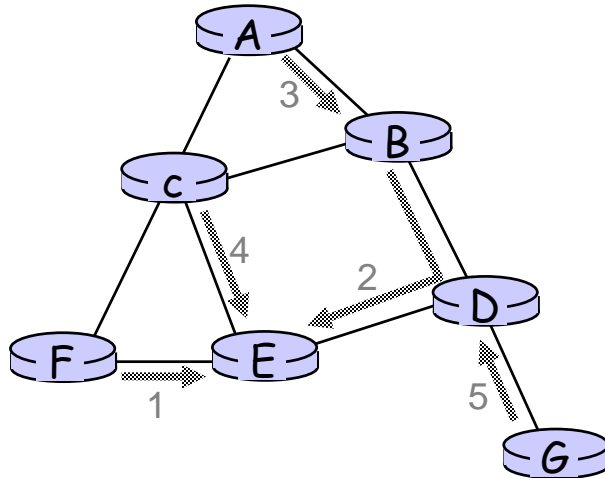
(a) Broadcast initiated at A



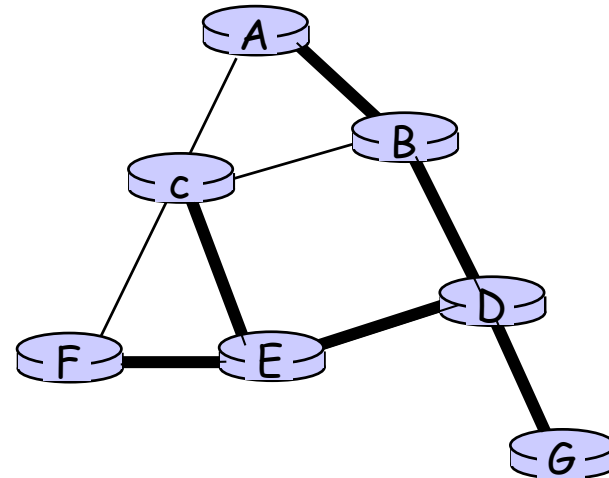
(b) Broadcast initiated at D

Spanning Tree: Creation

- ❖ center node
- ❖ each node sends unicast join message to center node
 - message forwarded until it arrives at a node already belonging to spanning tree



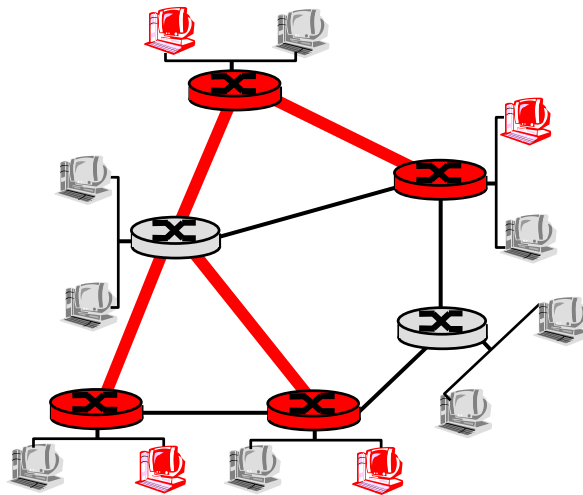
(a) Stepwise construction of spanning tree



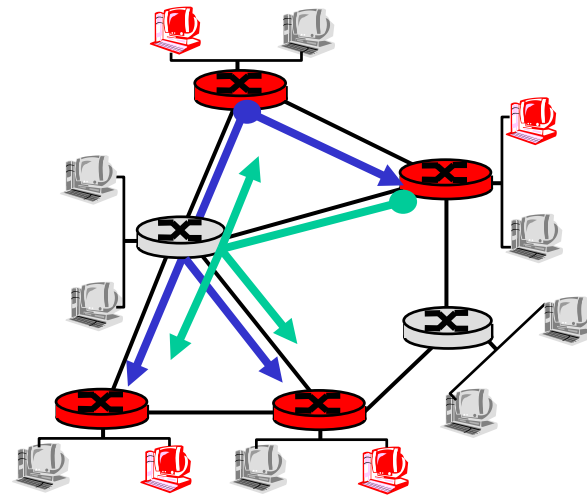
(b) Constructed spanning tree

Multicast Routing: Problem Statement

- ❖ **Goal:** find a tree (or trees) connecting routers having local multicast group members
 - **tree:** not all paths between routers used
 - **source-based:** different tree from each sender to receivers
 - **shared-tree:** same tree used by all group members



Shared tree



Source-based trees



Approaches for building multicast trees

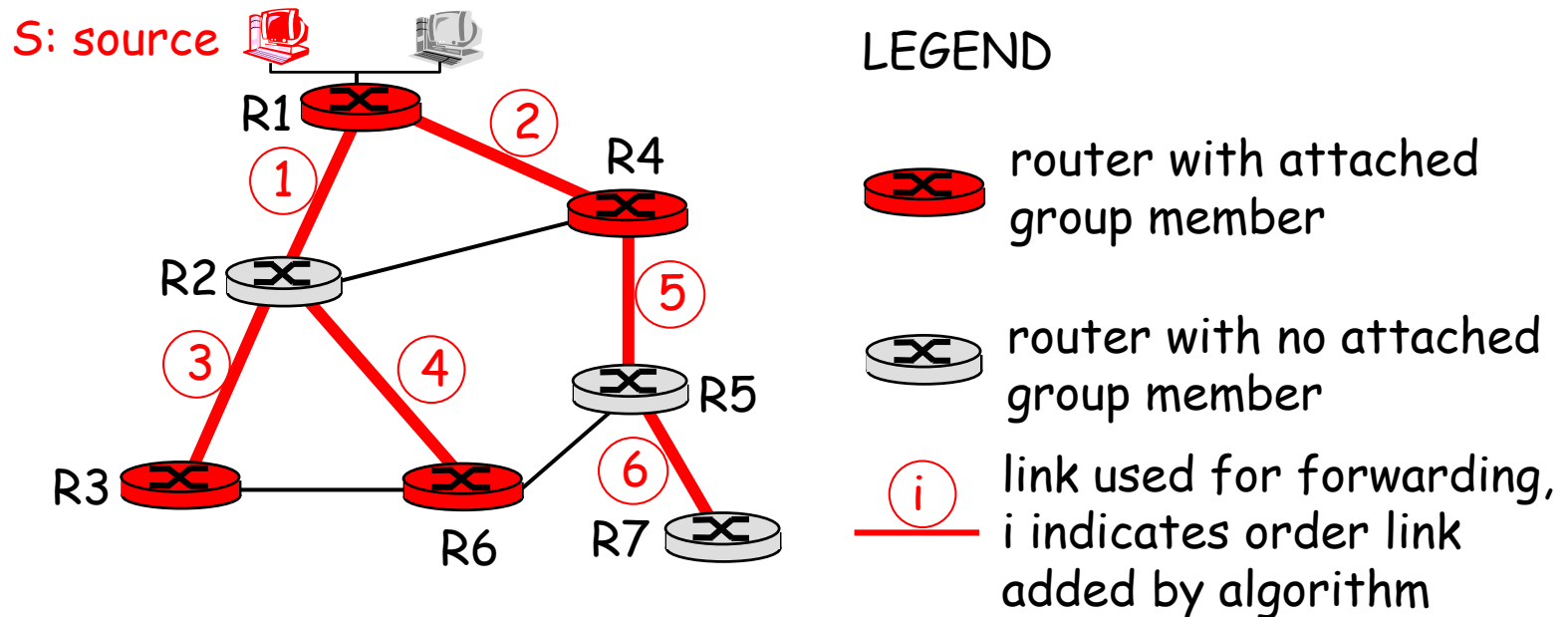
Approaches:

- ❖ **source-based tree:** one tree per source
 - shortest path trees
 - reverse path forwarding
- ❖ **group-shared tree:** group uses one tree
 - minimal spanning (Steiner)
 - center-based trees

...we first look at basic approaches, then specific protocols adopting these approaches

Shortest Path Tree

- ❖ multicast forwarding tree: tree of shortest path routes from source to all receivers
 - Dijkstra's algorithm





Reverse Path Forwarding

- ❖ rely on router's knowledge of unicast shortest path from it to sender
- ❖ each router has simple forwarding behavior:

if (multicast datagram received on incoming link on shortest path back to center)
then flood datagram onto all outgoing links
else ignore datagram