CSC358 *Intro. to Computer Networks*
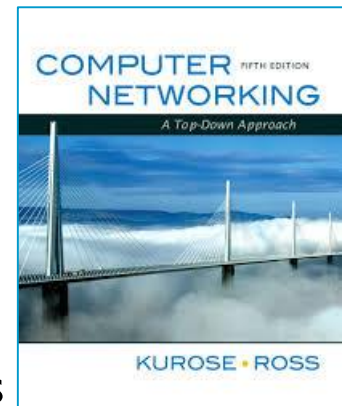
Lecture 9: *DV, Routing in the Internet*

Amir H. Chinaei, Winter 2016

ahchinaei@cs.toronto.edu
http://www.cs.toronto.edu/~ahchinaei/

Many slides are (inspired/adapted) from the above source
© all material copyright; all rights reserved for the authors

Office Hours: T 17:00–18:00 R 9:00–10:00 BA4222

TA Office Hours: W 16:00-17:00 BA3201 R 10:00-11:00 BA7172
csc358ta@cdf.toronto.edu
http://www.cs.toronto.edu/~ahchinaei/teaching/2016jan/csc358/

Distance vector algorithm

Bellman-Ford equation (dynamic programming)

let

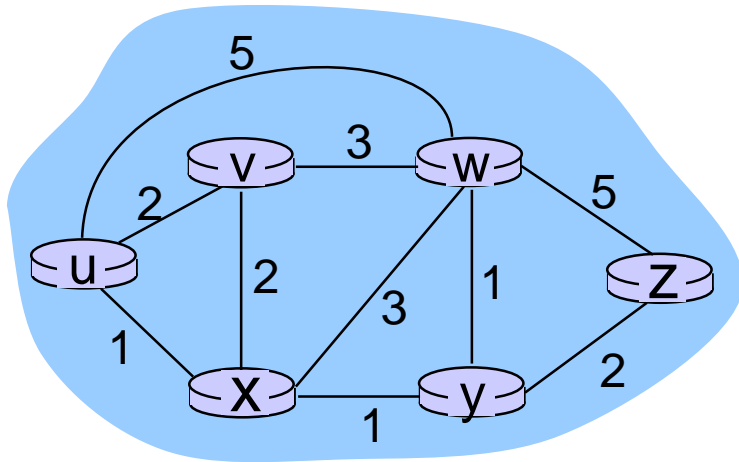$d_x(y) :=$ cost of least-cost path from x to y

then

$$d_x(y) = \underset{v}{min} \{c(x,v) + d_v(y) \}$$

cost from neighbor v to destination y

cost to neighbor v

min taken over all neighbors v of x

Bellman-Ford example

clearly, $d_v(z) = 5$, $d_x(z) = 3$, $d_w(z) = 3$

B-F equation says:

$$d_u(z) = \min \{ c(u,v) + d_v(z),$$
$$c(u,x) + d_x(z),$$
$$c(u,w) + d_w(z) \}$$
$$= \min \{2 + 5,$$
$$1 + 3,$$
$$5 + 3\} = 4$$

node achieving minimum is next
hop in shortest path, used in forwarding table

Distance vector algorithm

❖ $D_x(y)$ = estimate of least cost from x to y
  ▪ x maintains distance vector $\mathbf{D_x}$ = [$D_x(y)$: y ϵ N]
❖ node x:
  ▪ knows cost to each neighbor v: $c(x,v)$
  ▪ maintains its neighbors' distance vectors. For each neighbor v, x maintains
    $\mathbf{D_v}$ = [$D_v(y)$: y ϵ N]

Distance vector algorithm

key idea:

❖ from time-to-time, each node sends its own distance vector estimate to neighbors

❖ when x receives new DV estimate from neighbor, it updates its own DV using B-F equation:

$$D_x(y) \leftarrow min_v\{c(x,v) + D_v(y)\} \quad \text{for each node } y \in N$$

❖ under minor, natural conditions, the estimate $D_x(y)$ *converge to the actual least cost* $d_x(y)$
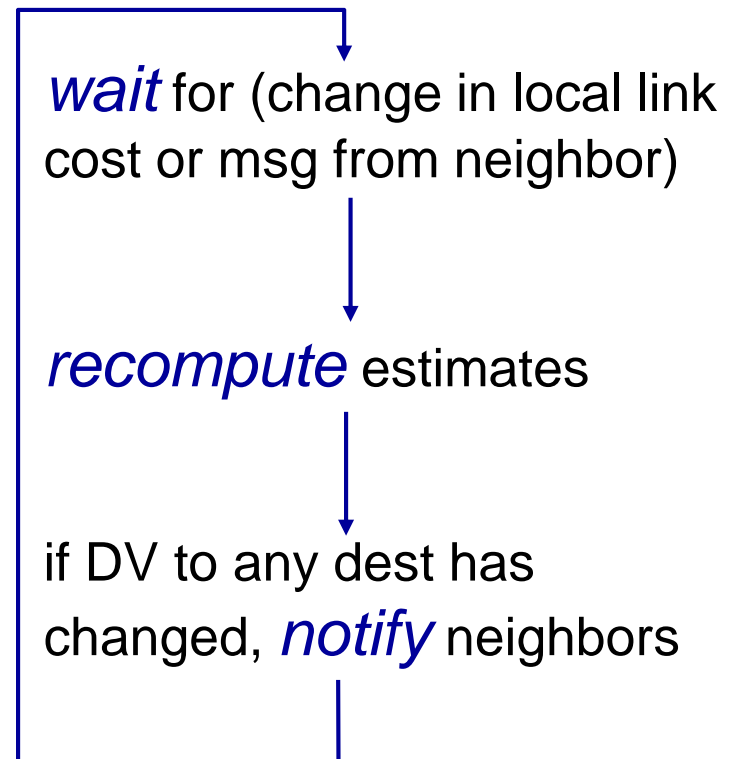
Distance vector algorithm

iterative, asynchronous:
each local iteration caused by:

❖ local link cost change

❖ DV update message from neighbor

distributed:

❖ each node notifies neighbors *only* when its DV changes

  ▪ neighbors then notify their neighbors if necessary

each node:

wait for (change in local link cost or msg from neighbor)

recompute estimates

if DV to any dest has changed, *notify* neighbors

$$D_x(y) = \min\{c(x,y) + D_y(y), c(x,z) + D_z(y)\}$$
$$= \min\{2+0, 7+1\} = 2$$

$$D_x(z) = \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\}$$
$$= \min\{2+1, 7+0\} = 3$$
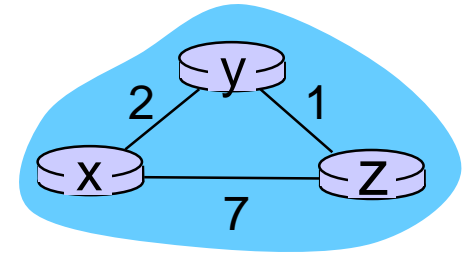
node x table

cost to

from	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

cost to

from	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

node y table

cost to

from	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

node z table

cost to

from	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

time

$$D_x(y) = \min\{c(x,y) + D_y(y), c(x,z) + D_z(y)\}$$
$$= \min\{2+0 , 7+1\} = 2$$

$$D_x(z) = \min\{c(x,y) + D_y(z), c(x,z) + D_z(z)\}$$
$$= \min\{2+1 , 7+0\} = 3$$

node x table

cost to

from	x	y	z
x	0	2	7
y	∞	∞	∞
z	∞	∞	∞

cost to

from	x	y	z
x	0	2	3
y	2	0	1
z	7	1	0

cost to

from	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

node y table

cost to

from	x	y	z
x	∞	∞	∞
y	2	0	1
z	∞	∞	∞

cost to

from	x	y	z
x	0	2	7
y	2	0	1
z	7	1	0

cost to

from	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

node z table

cost to

from	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

cost to

from	x	y	z
x	0	2	7
y	2	0	1
z	3	1	0

cost to

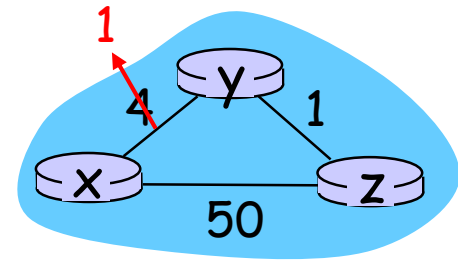| from	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

time

Distance vector: link cost changes

link cost changes:

❖ node detects local link cost change

❖ updates routing info, recalculates distance vector

❖ if DV changes, notify neighbors

"good news travels fast"

t_0 : y detects link-cost change, updates its DV, informs its neighbors.

t_1 : z receives update from y, updates its table, computes new least cost to x , sends its neighbors its DV.

t_2 : y receives z's update, updates its distance table. y's least costs do *not* change, so y does *not* send a message to z.

Distance vector: link cost changes

link cost changes:

- node detects local link cost change
- *bad news travels slow* - "count to infinity" problem!
- 44 iterations before algorithm stabilizes: see text
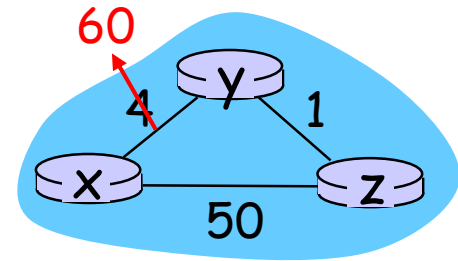
poisoned reverse:

- If Z routes through Y to get to X :
    - Z tells Y its (Z's) distance to X is infinite (so Y won't route to X via Z)
- will this completely solve count to infinity problem?

Comparison of LS and DV algorithms

message complexity

- **LS:** with n nodes, E links, O(nE) msgs sent
- **DV:** exchange between neighbors only
  - convergence time varies

speed of convergence

- **LS:** O(n²) algorithm requires O(nE) msgs
  - may have oscillations
- **DV:** convergence time varies
  - may be routing loops
  - count-to-infinity problem

robustness: what happens if router malfunctions?

LS:

- node can advertise incorrect *link* cost
- each node computes only its *own* table

DV:

- DV node can advertise incorrect *path* cost
- each node's table used by others
  - error propagate thru network

Chapter 4: outline

4.1 introduction

4.2 virtual circuit and
   datagram networks

4.3 what's inside a router

4.4 IP: Internet Protocol
- datagram format
- IPv4 addressing
- ICMP
- IPv6

4.5 routing algorithms
- link state
- distance vector
- hierarchical routing

4.6 routing in the Internet
- RIP
- OSPF
- BGP

4.7 broadcast and multicast
   routing

Hierarchical routing

our routing study thus far - idealization
- ❖ all routers identical
- ❖ network "flat"
… *not* true in practice

scale: with 600 million destinations:
- ❖ can't store all dest's 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
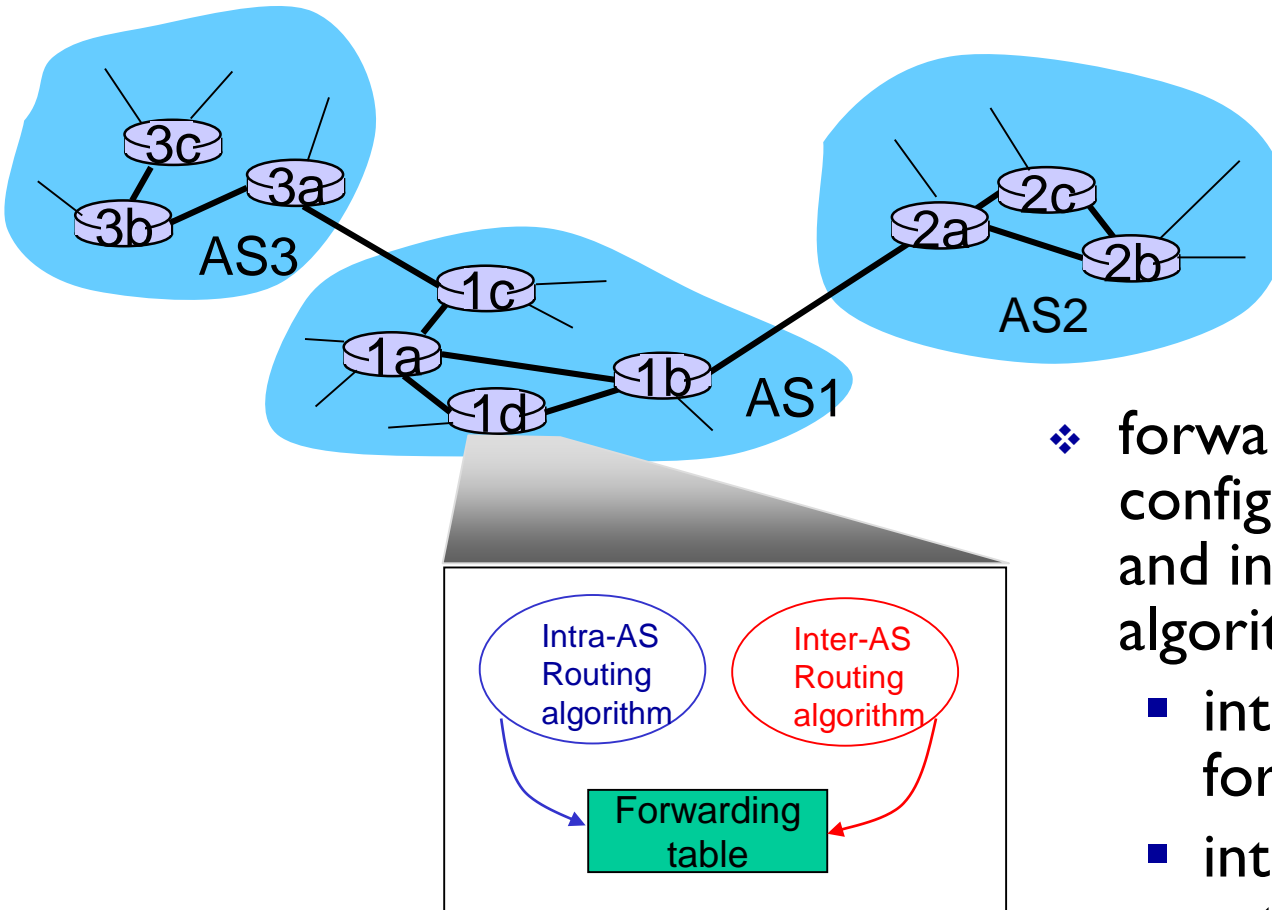
- ❖ **forwarding table configured by both intra- and inter-AS routing algorithm**
  - ▪ **intra-AS sets entries for internal dests**
  - ▪ **inter-AS & intra-AS sets entries for external dests**

Inter-AS tasks

❖ suppose router in AS1 receives datagram destined outside of AS1:

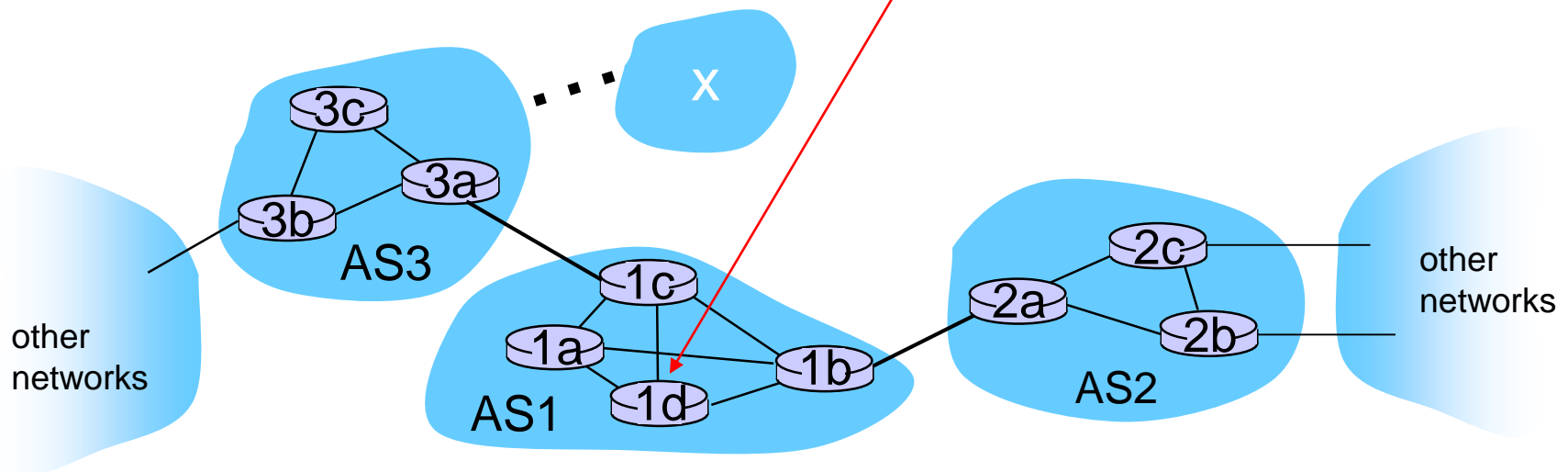 ▪ router should forward packet to gateway router, but which one?

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!
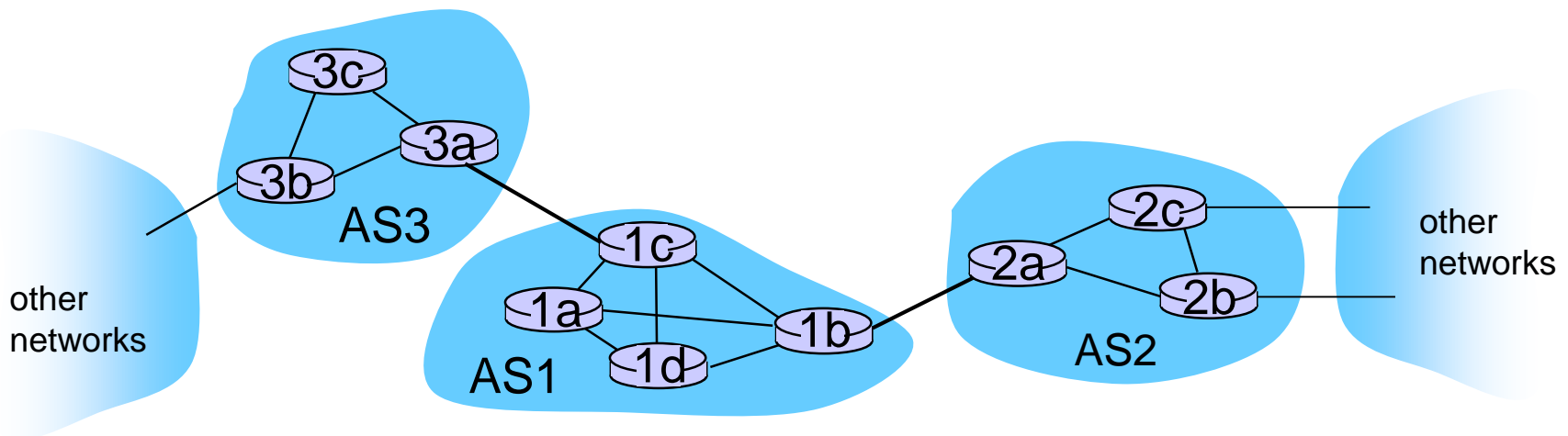
Example: setting forwarding table in router 1d

- ❖ 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
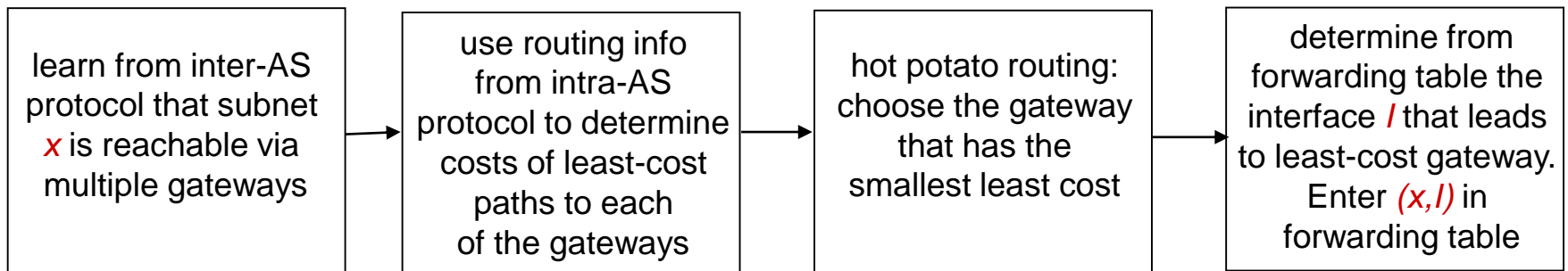 - ▪ installs forwarding table entry *(x,I)*

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!
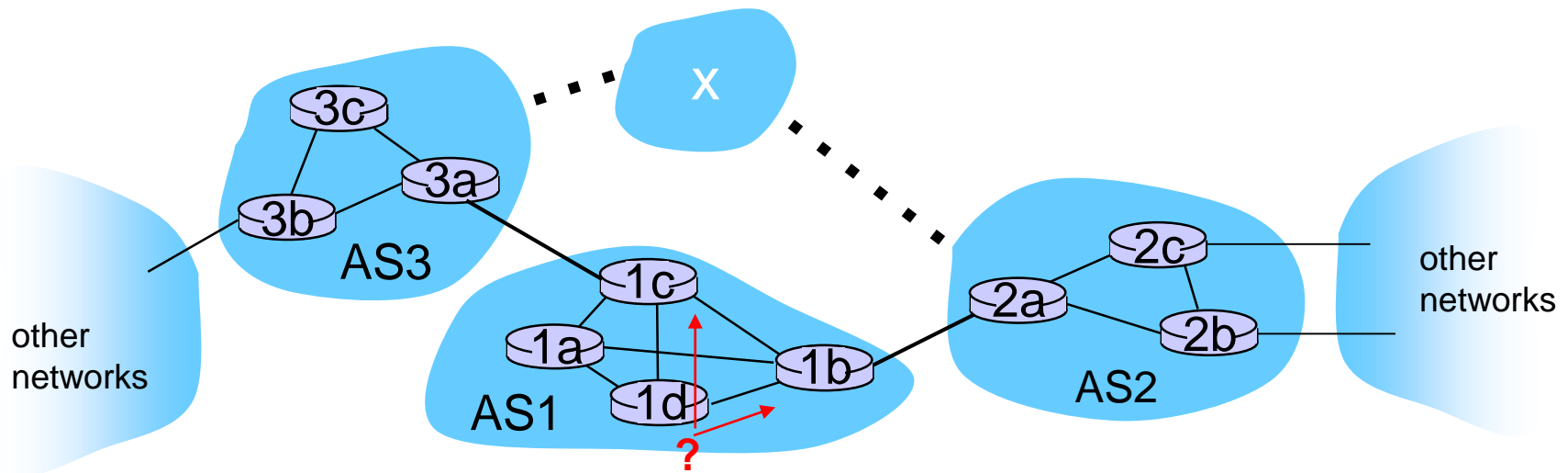
Example: choosing among multiple ASes

❖ 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.

| learn from inter-AS protocol that subnet *x* is reachable via multiple gateways | → | use routing info from intra-AS protocol to determine costs of least-cost paths to each of the gateways | → | hot potato routing: choose the gateway that has the smallest least cost | → | determine from forwarding table the interface *I* that leads to least-cost gateway. Enter *(x,I)* in forwarding table |

Chapter 4: outline

4.1 introduction

4.2 virtual circuit and datagram networks

4.3 what's inside a router

4.4 IP: Internet Protocol
- datagram format
- IPv4 addressing
- ICMP
- IPv6

4.5 routing algorithms
- link state
- distance vector
- hierarchical routing

4.6 routing in the Internet
- RIP
- OSPF
- BGP

4.7 broadcast and multicast routing

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)

❖ included in BSD-UNIX distribution in 1982
❖ 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)
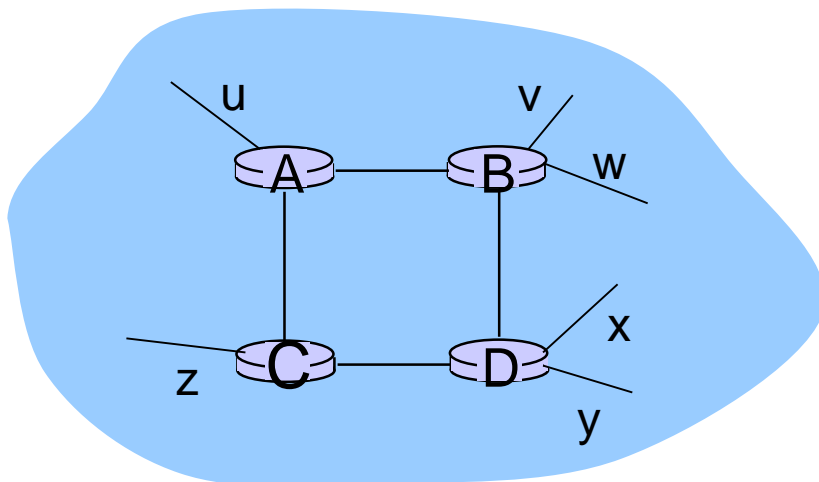
from router A to destination *subnets:*

subnet	hops
u	1
v	2
w	2
x	3
y	3
z	2

RIP: example
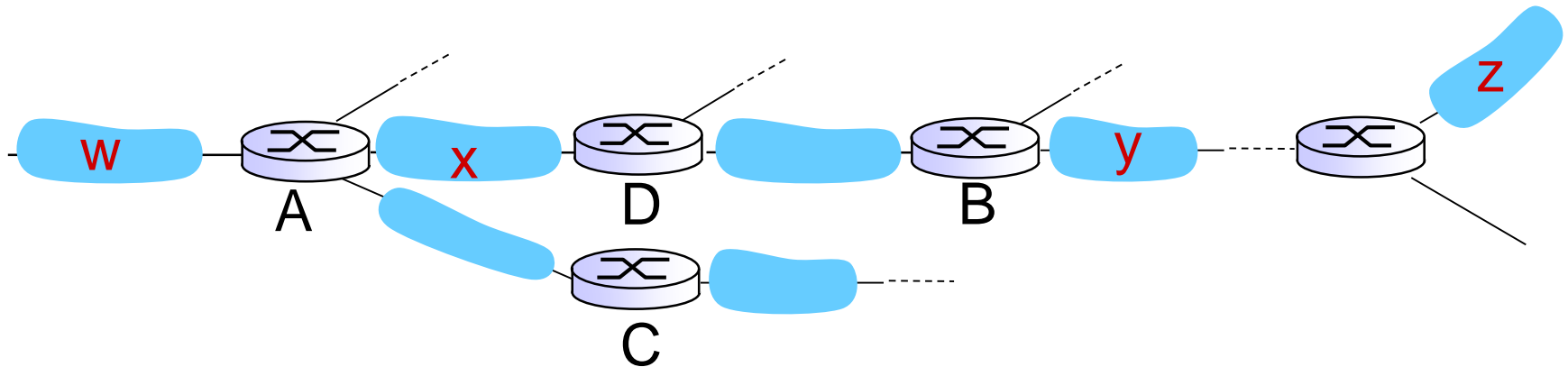
routing table in router D

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
….	….	….
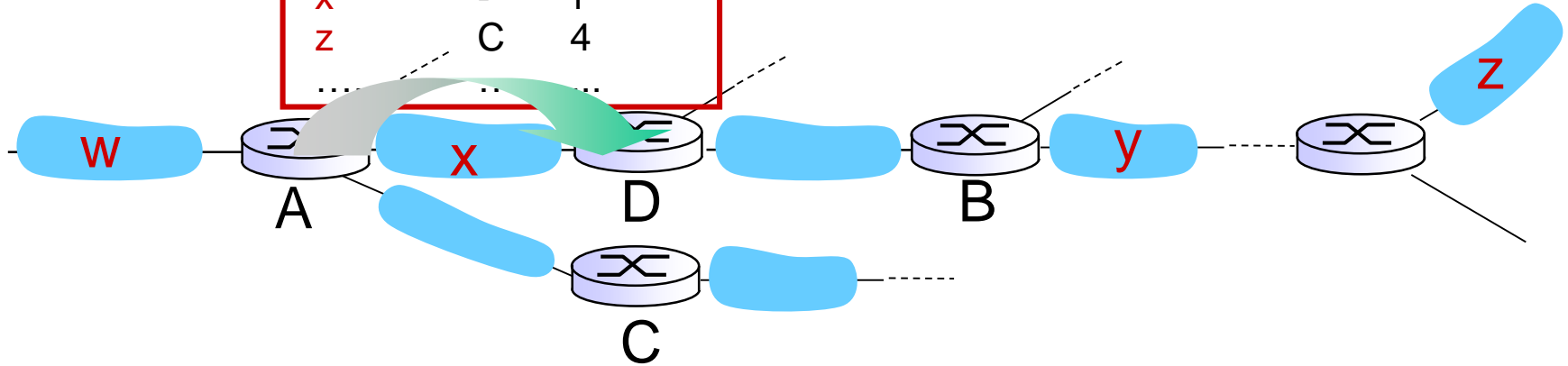
routing table in router D

destination subnet	next router	# hops to dest
w	A	2
y	B	2
z	B A	7 5
x	--	1
….	….	….

RIP: link failure, 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

routed

routed

transport (UDP)	
network (IP)	*forwarding table*
link	
physical	

	transprt (UDP)
forwarding table	network (IP)
	link
	physical
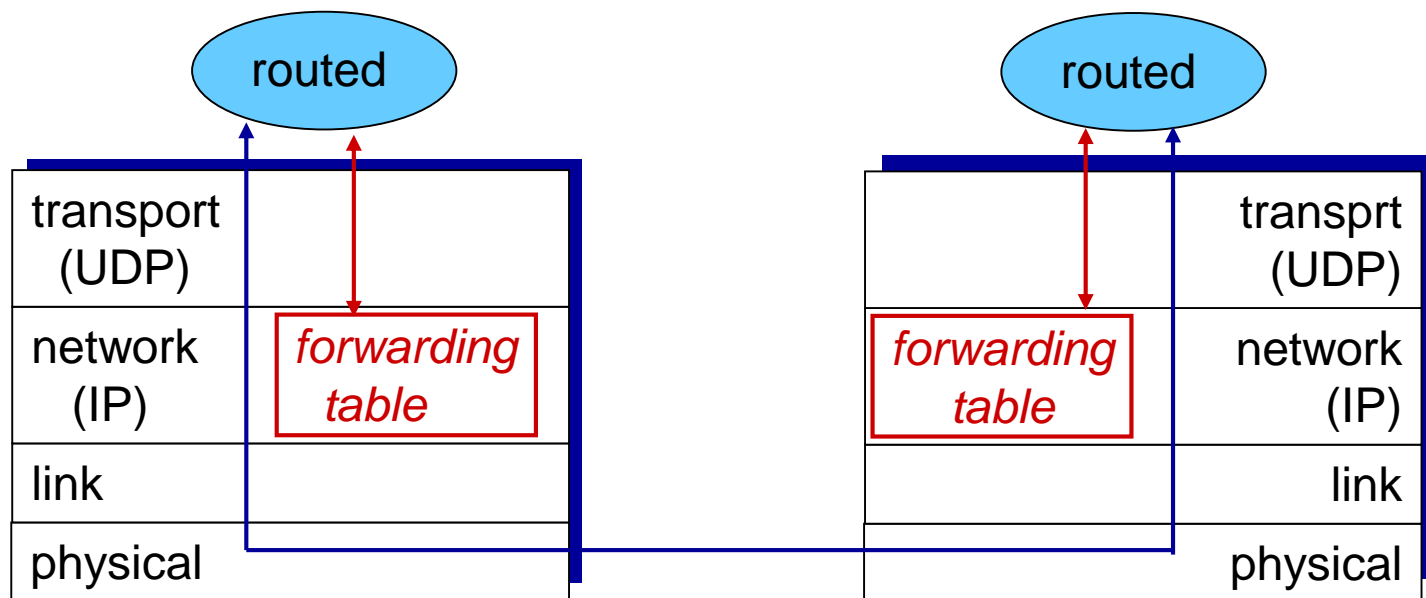
OSPF (Open Shortest Path First)

* "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
* advertisements flooded to *entire* AS
  * carried in OSPF messages directly over IP (rather than TCP or UDP
* *IS-IS routing* protocol: nearly identical to OSPF

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 (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
* hierarchical OSPF in large domains.

Hierarchical OSPF

boundary router

backbone router

backbone

area
border
routers

internal
routers

area 1

area 2

area 3
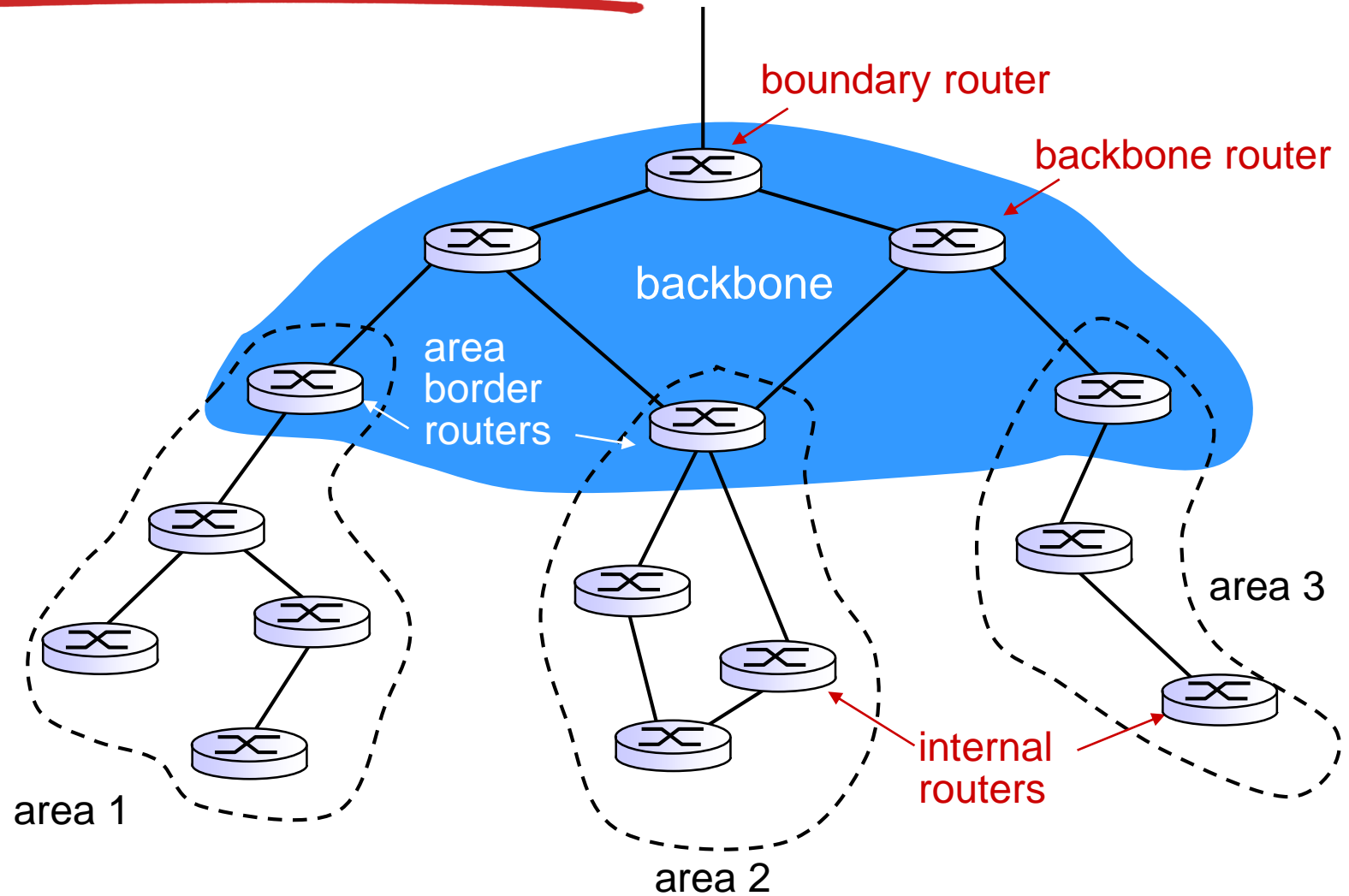
Hierarchical OSPF

❖ *two-level hierarchy:* local area, backbone.
  ▪ link-state advertisements only in area
  ▪ each nodes has detailed area topology; only know direction (shortest path) to nets in other areas.
❖ *area border routers:* "summarize" distances to nets in own area, advertise to other Area Border routers.
❖ *backbone routers:* run OSPF routing limited to backbone.
❖ *boundary routers:* connect to other AS's.

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
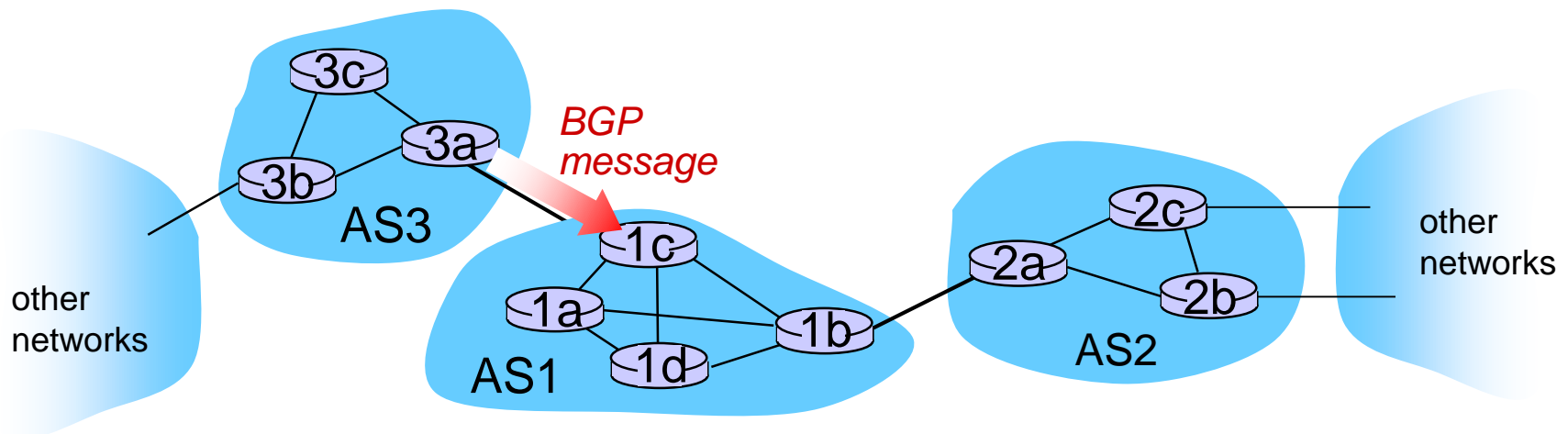
BGP basics: distributing path information

❖ using eBGP session between 3a and 1c, AS3 sends prefix reachability info to AS1.

▪ 1c can then use iBGP do 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.
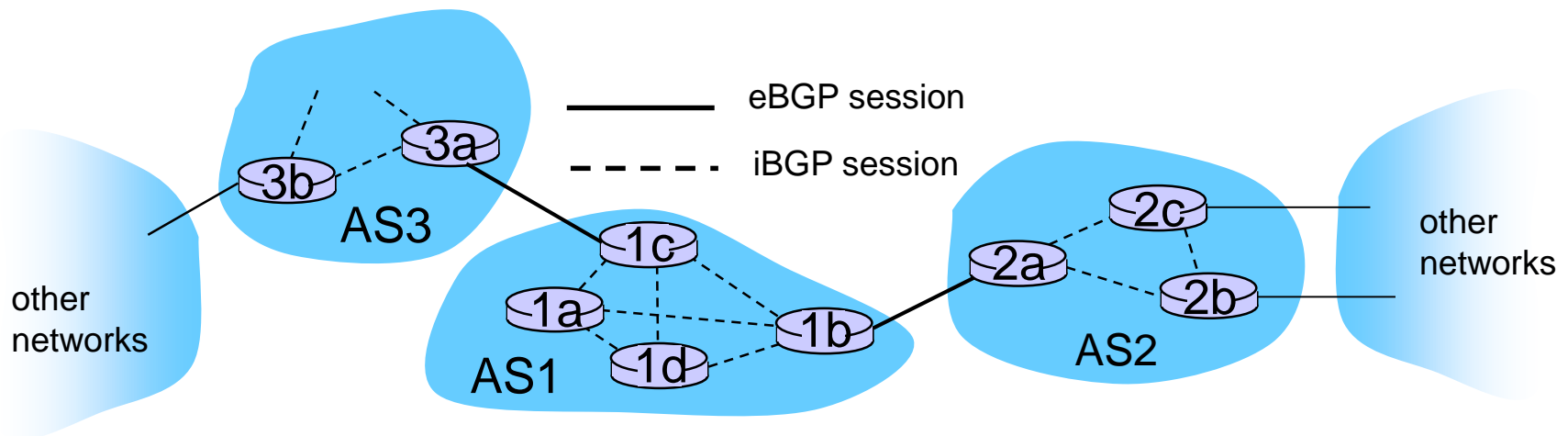
Path attributes and BGP routes

- ❖ 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 1 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 messages

❖ BGP messages exchanged between peers over TCP connection

❖ BGP messages:

- OPEN: opens TCP connection to peer and authenticates sender

- UPDATE: advertises new path (or withdraws old)

- KEEPALIVE: keeps connection alive in absence of UPDATES; also ACKs OPEN request

- NOTIFICATION: reports errors in previous msg; also used to close connection

Putting it Altogether:
How Does an Entry Get Into a Router´s Forwarding Table?

❖ Answer is complicated!

❖ Ties together hierarchical routing (Section 4.5.3) with BGP (4.6.3) and OSPF (4.6.2).

❖ Provides nice overview of BGP!

How does entry get in forwarding table?

routing algorithms

local forwarding table

prefix	output port
138.16.64/22	3
124.12/16	2
212/8	4
…………	…

entry

Assume prefix is in another AS.

Dest IP

1

3 2
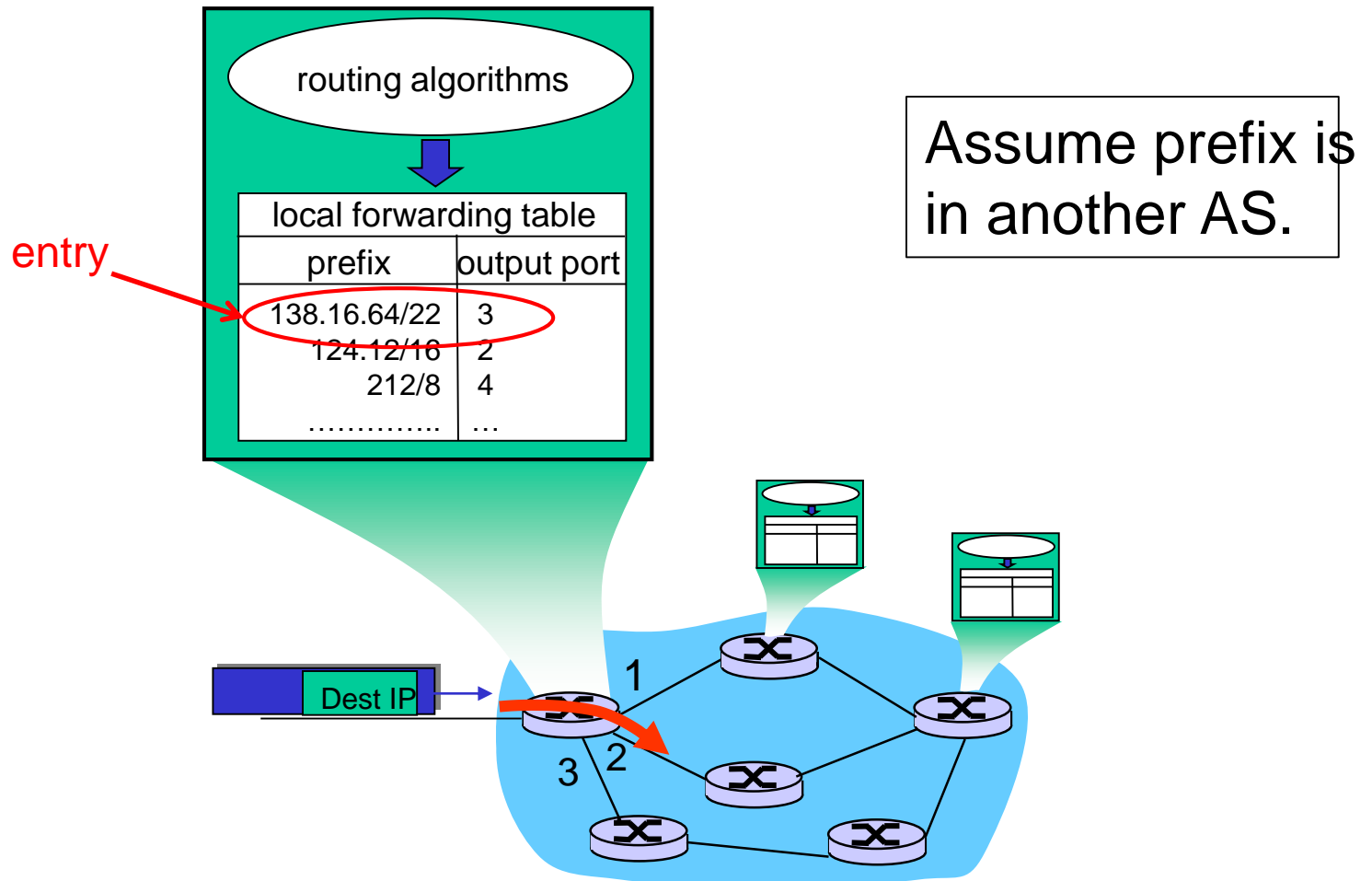
How does entry get in forwarding table?

High-level overview

1. Router becomes aware of prefix
2. Router determines output port for prefix
3. Router enters prefix-port in forwarding table

Router becomes aware of prefix
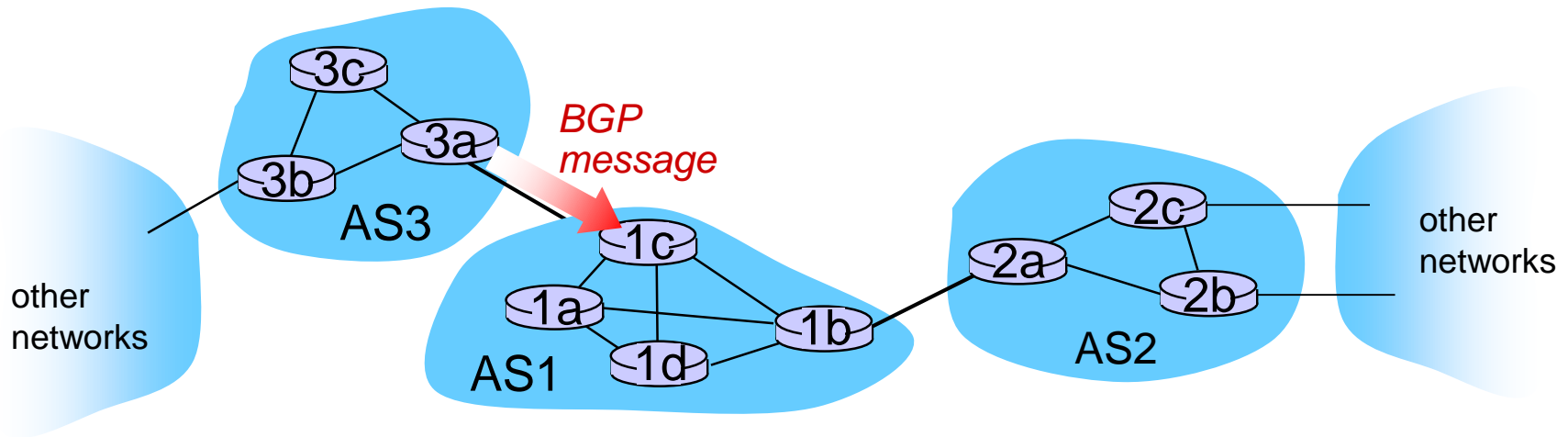
- BGP message contains "routes"
- "route" is a prefix and attributes: AS-PATH, NEXT-HOP,...
- Example: route:
  - Prefix:138.16.64/22 ; AS-PATH: AS3 AS131 ; NEXT-HOP: 201.44.13.125

Router may receive multiple routes
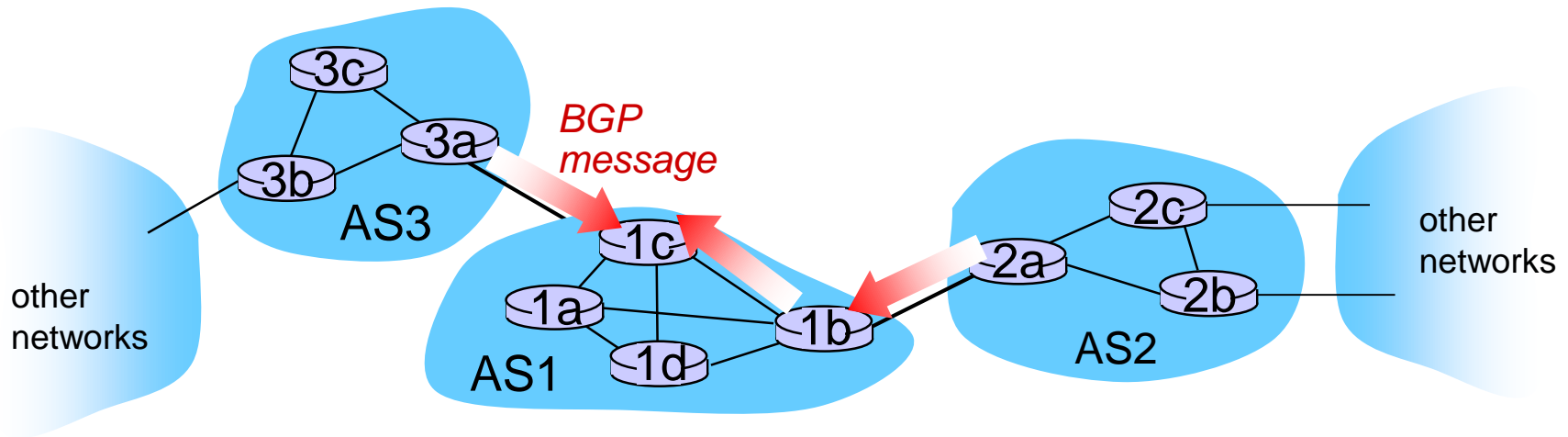
❖ Router may receive multiple routes for same prefix
❖ Has to select one route

Select best BGP route to prefix

❖ Router selects route based on shortest AS-PATH

❖ Example:

select

❖ AS2 AS17 to 138.16.64/22
❖ AS3 AS131 AS201 to 138.16.64/22

❖ What if there is a tie? We'll come back to that!

Find best intra-route to BGP route

- ❖ Use selected route's NEXT-HOP attribute
  - ▪ Route's NEXT-HOP attribute is the IP address of the router interface that begins the AS PATH.
- ❖ Example:
  - ❖ AS-PATH: AS2 AS17 ; NEXT-HOP: 111.99.86.55
- ❖ Router uses OSPF to find shortest path from 1c to 111.99.86.55
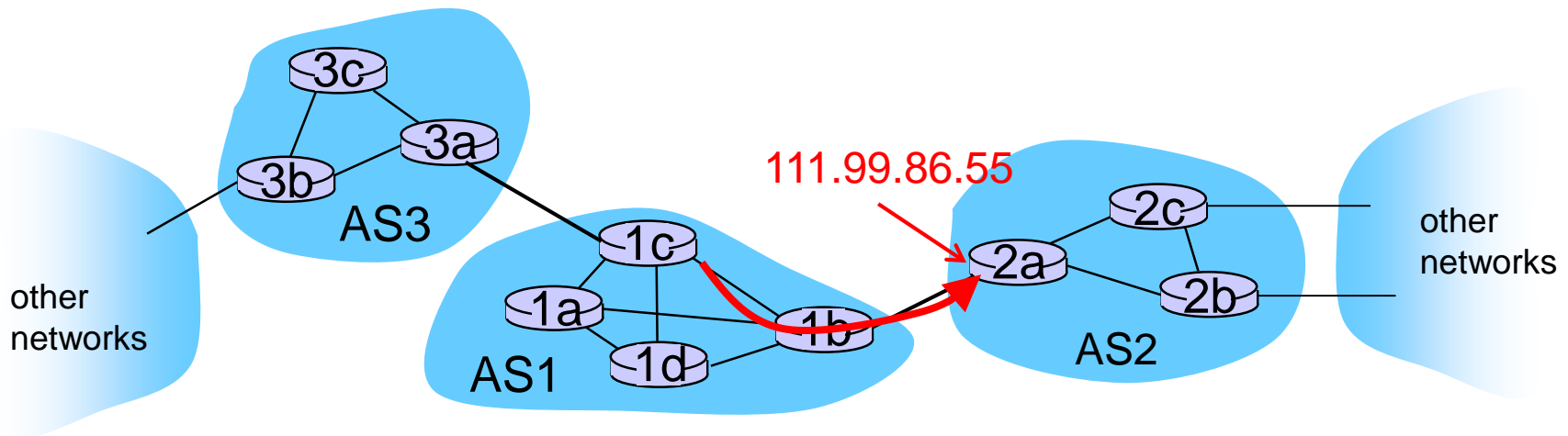
Router identifies port for route

- ❖ Identifies port along the OSPF shortest path
- ❖ Adds prefix-port entry to its forwarding table:
  - ▪ (138.16.64/22 , port 4)
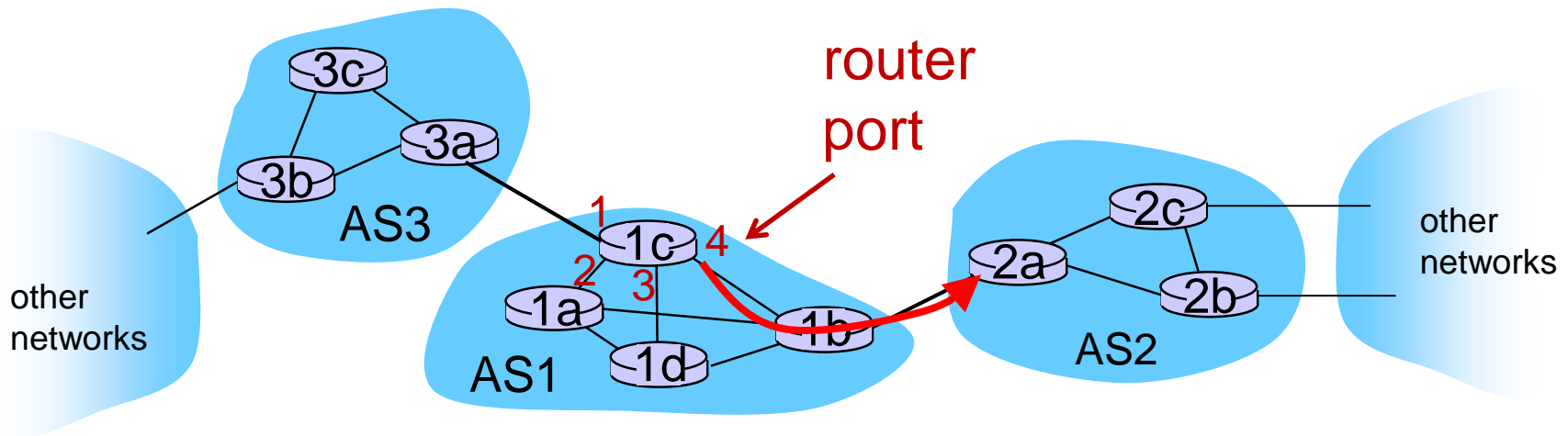
Hot Potato Routing

❖ Suppose there two or more best inter-routes.
❖ Then choose route with closest NEXT-HOP
   ▪ Use OSPF to determine which gateway is closest
   ▪ Q: From 1c, chose AS3 AS131 or AS2 AS17?
   ▪ A: route AS3 AS131 since it is closer
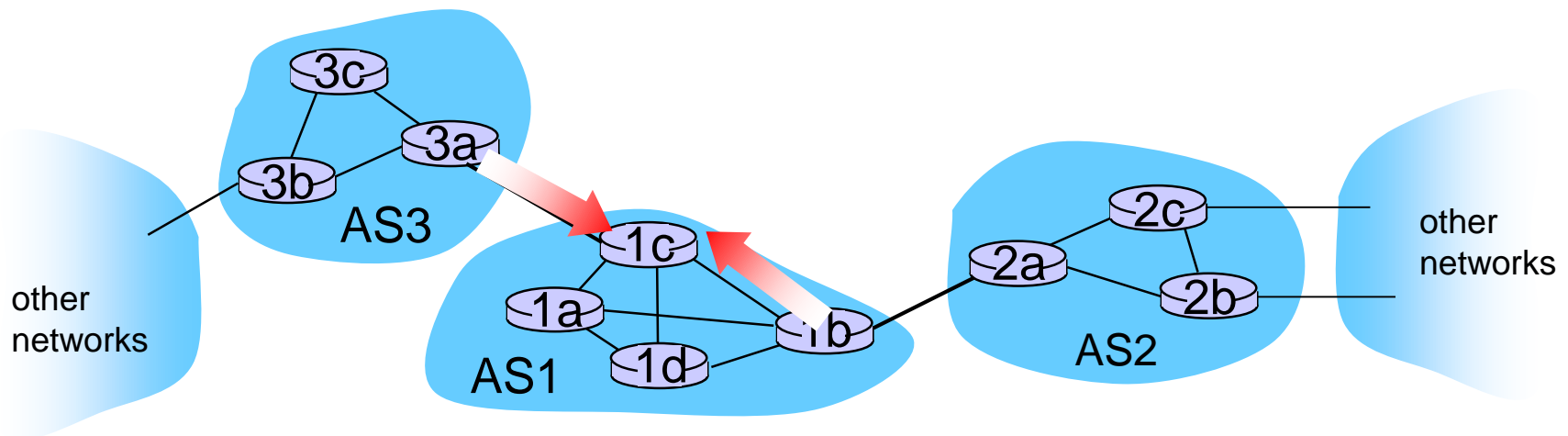
How does entry get in forwarding table?

Summary

1. Router becomes aware of prefix
   - via BGP route advertisements from other routers
2. Determine router output port for prefix
   - Use BGP route selection to find best inter-AS route
   - Use OSPF to find best intra-AS route leading to best inter-AS route
   - Router identifies router port for that best route
3. Enter prefix-port entry in forwarding table

BGP routing policy

legend:
provider network
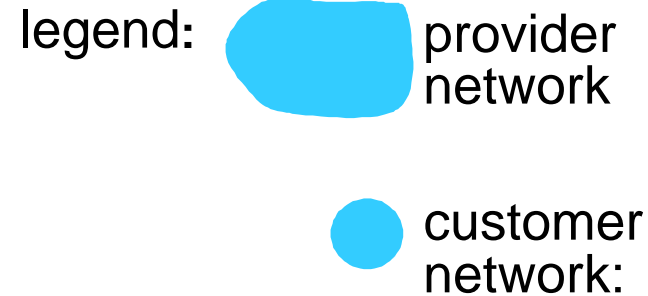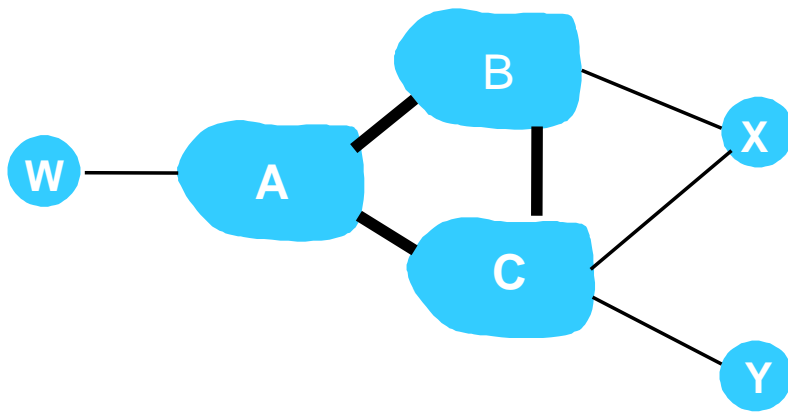
customer network:

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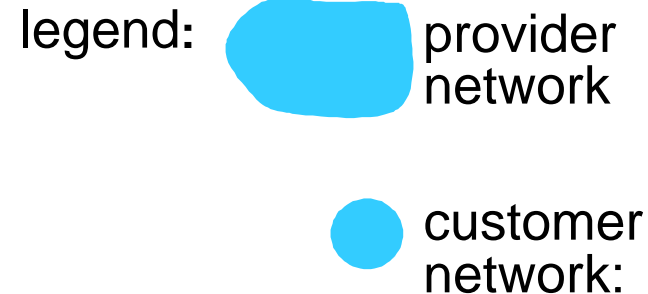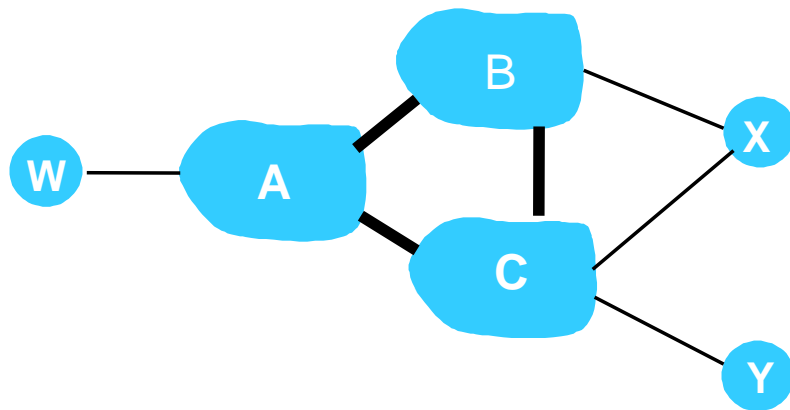
- ❖ A advertises path AW to B
- ❖ B advertises path BAW to X
- ❖ Should B advertise path BAW to C?
  - No way! B gets no "revenue" for routing CBAW since neither W nor C are B's customers
  - B wants to force C to route to w via A
  - B wants to route *only* to/from its customers!

Why different Intra-, Inter-AS routing ?

policy:

- ❖ inter-AS: admins want 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

Chapter 4: outline

4.1 introduction

4.2 virtual circuit and datagram networks

4.3 what's inside a router

4.4 IP: Internet Protocol
- datagram format
- IPv4 addressing
- ICMP
- IPv6

4.5 routing algorithms
- link state
- distance vector
- hierarchical routing

4.6 routing in the Internet
- RIP
- OSPF
- BGP

4.7 broadcast and multicast routing

Broadcast routing

❖ deliver packets from source to all other nodes

❖ source duplication is inefficient:

source
duplication

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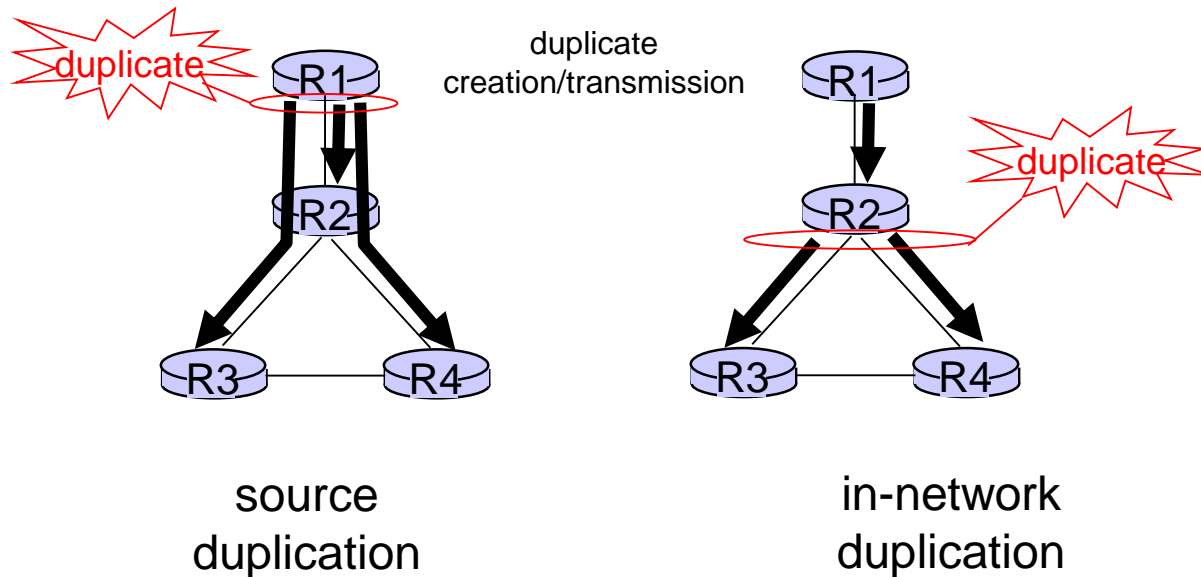
❖ source duplication: how does source determine recipient addresses?

In-network duplication

❖ *flooding:* when node receives broadcast packet, sends copy to all neighbors
   ▪ problems: cycles & broadcast storm
❖ *controlled flooding:* node only broadcasts pkt if it hasn't broadcast same packet before
   ▪ node keeps track of packet ids 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
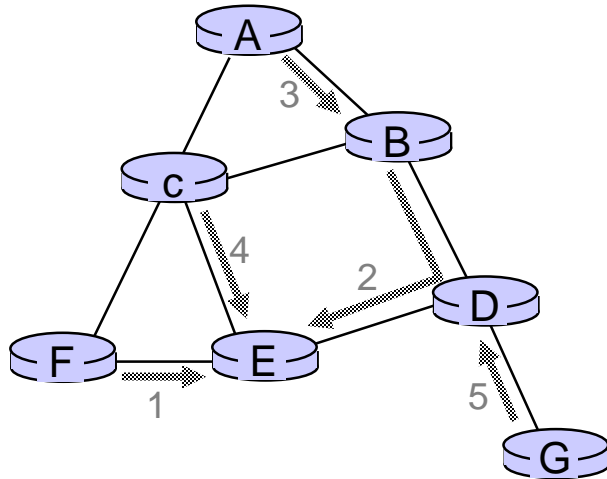❖ nodes then forward/make copies only along spanning tree
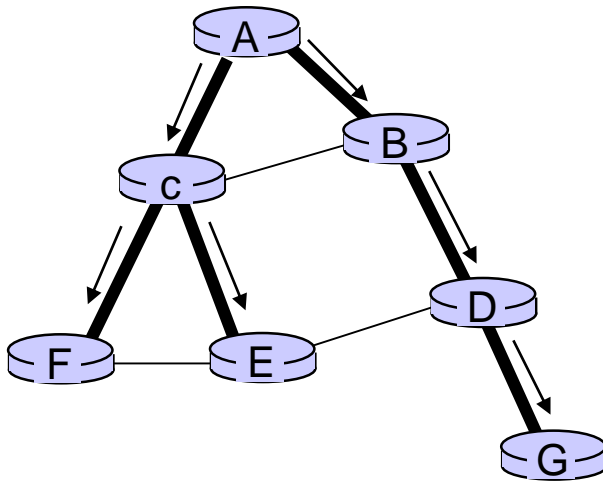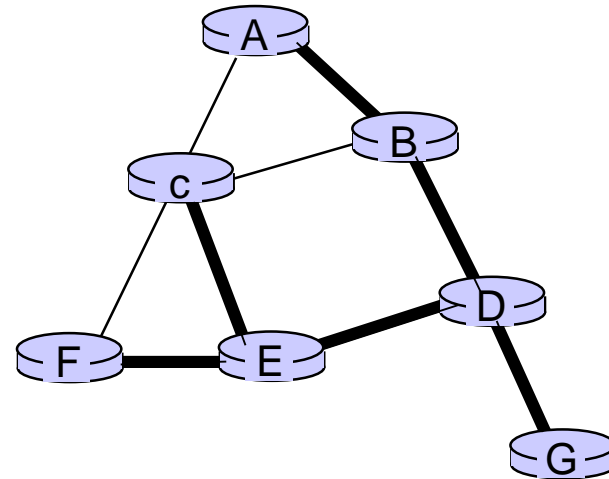

(a) broadcast initiated at A
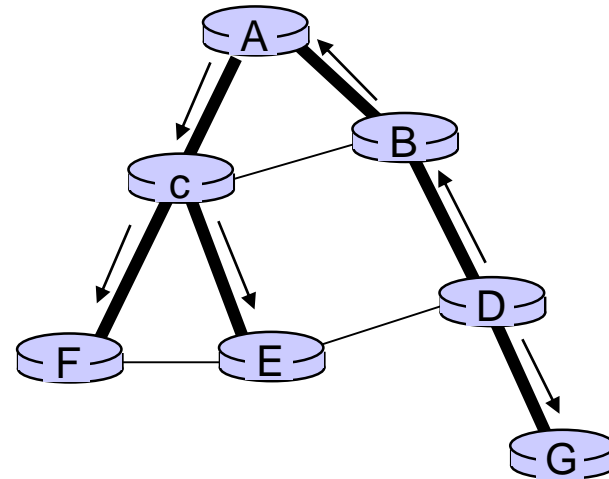
(b) broadcast initiated at D

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 (center: E)

(b) constructed spanning tree

Multicast routing: problem statement

goal: find a tree (or trees) connecting routers having local mcast group members

- ❖ *tree:* not all paths between routers used
- ❖ *shared-tree:* same tree used by all group members
- ❖ *source-based:* different tree from each sender to rcvrs
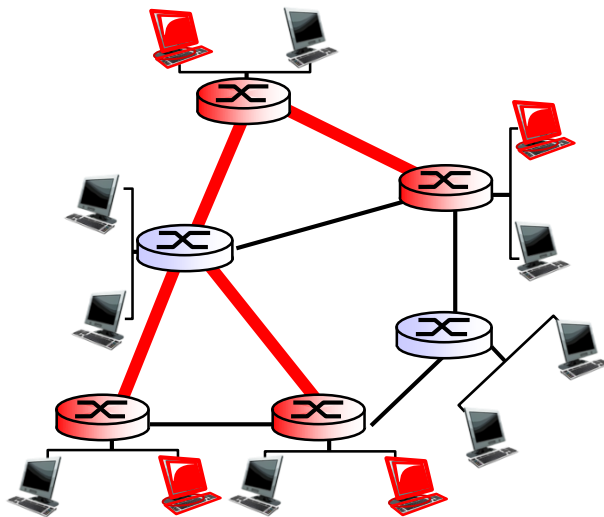

legend

- group member
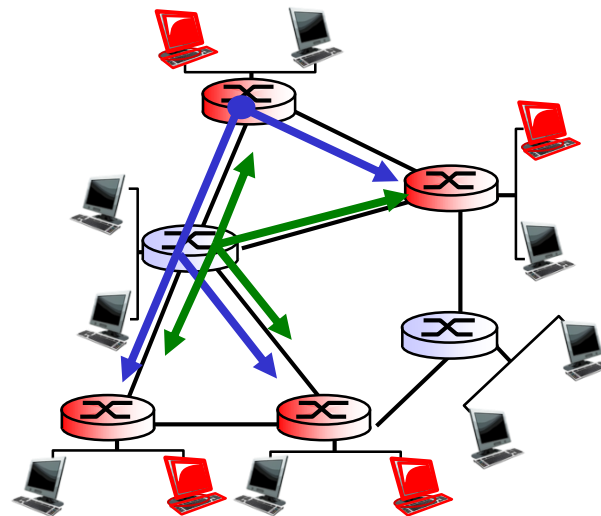- not group member
- router with a group member
- router without group member

shared tree   source-based trees
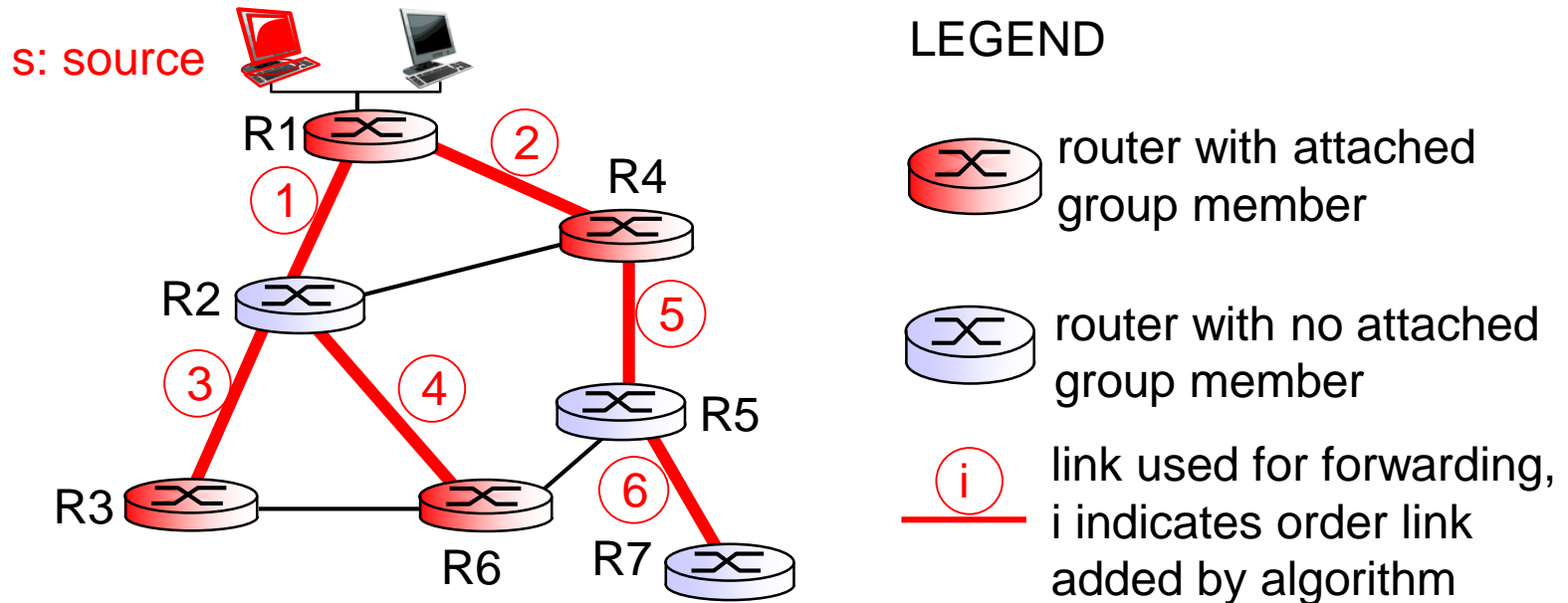
Approaches for building mcast trees

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

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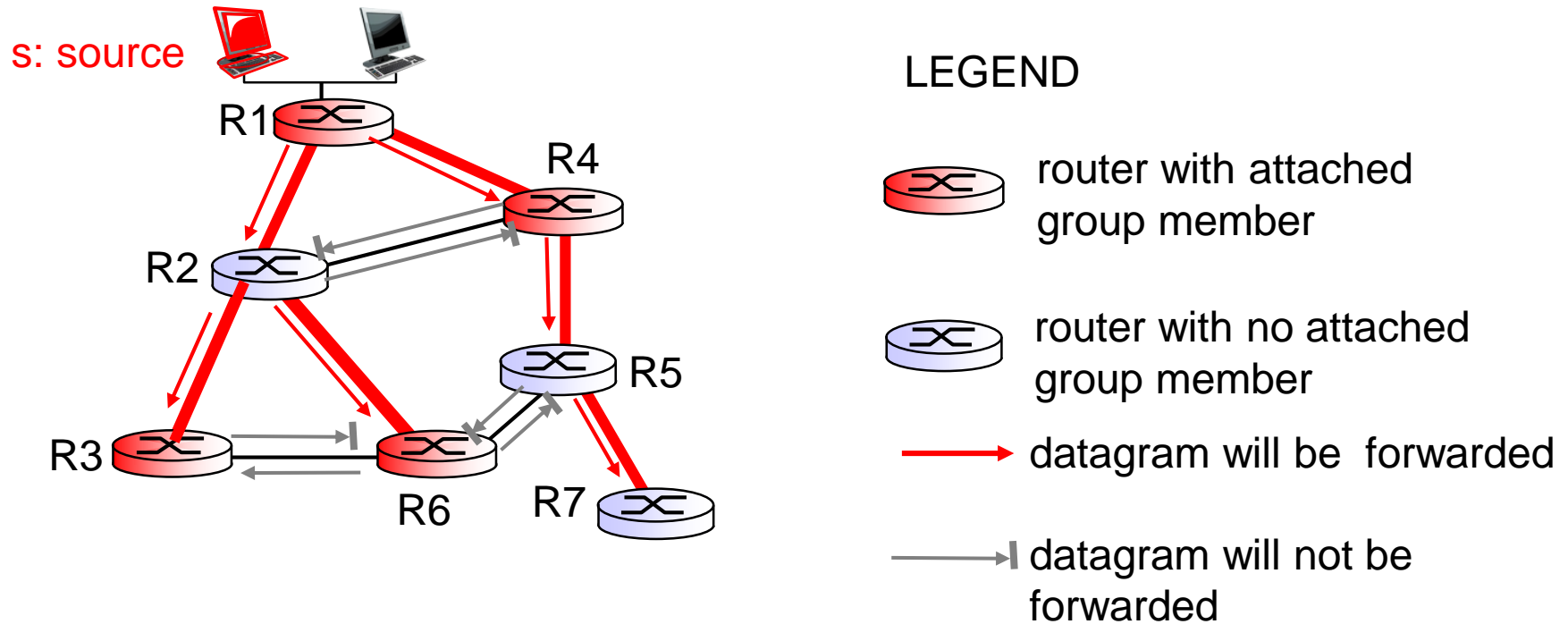
  ▪ Dijkstra's algorithm

s: source

LEGEND

router with attached group member

router with no attached group member

(i) link used for forwarding, i indicates order link added by algorithm

Reverse path forwarding

❖ rely on router's knowledge of unicast shortest path from it to sender

❖ each router has simple forwarding behavior:

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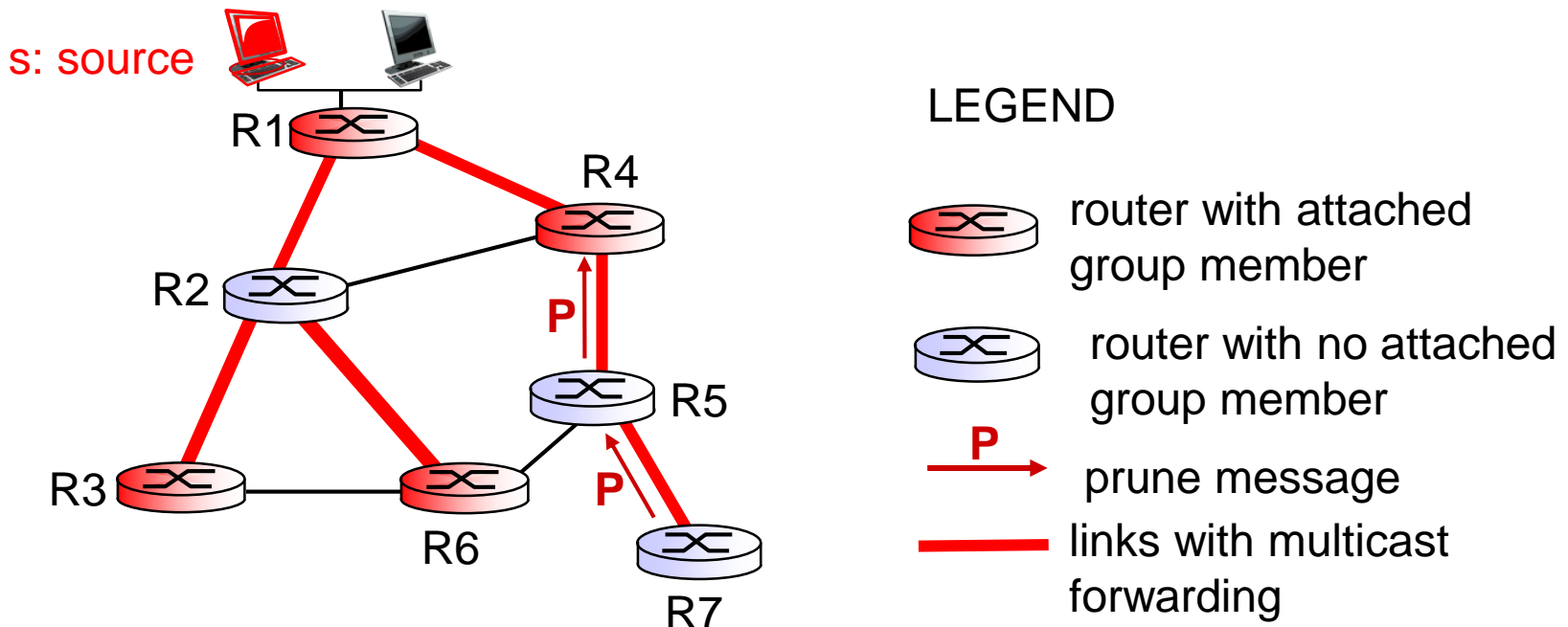 *else* ignore datagram

Reverse path forwarding: example

LEGEND

router with attached group member

router with no attached group member

datagram will be forwarded

datagram will not be forwarded

❖ result is a source-specific *reverse* SPT
  ▪ may be a bad choice with asymmetric links

Reverse path forwarding: pruning

❖ forwarding tree contains subtrees with no mcast group members

  ▪ no need to forward datagrams down subtree
  ▪ "prune" msgs sent upstream by router with no downstream group members

s: source

LEGEND

router with attached group member

router with no attached group member

P ➔ prune message

links with multicast forwarding

R1
R2
R3
R4
R5
R6
R7

P
P

Shared-tree: steiner tree

❖ *steiner tree:* minimum cost tree connecting all routers with attached group members

❖ problem is NP-complete

❖ excellent heuristics exists

❖ not used in practice:
  ▪ computational complexity
  ▪ information about entire network needed
  ▪ monolithic: rerun whenever a router needs to join/leave

Center-based trees

❖ single delivery tree shared by all
❖ one router identified as *"center"* of tree
❖ to join:
  ▪ edge router sends unicast *join-msg* addressed to center router
  ▪ *join-msg* "processed" by intermediate routers and forwarded towards center
  ▪ *join-msg* either hits existing tree branch for this center, or arrives at center
  ▪ path taken by *join-msg* becomes new branch of tree for this router

Center-based trees: example

suppose R6 chosen as center:
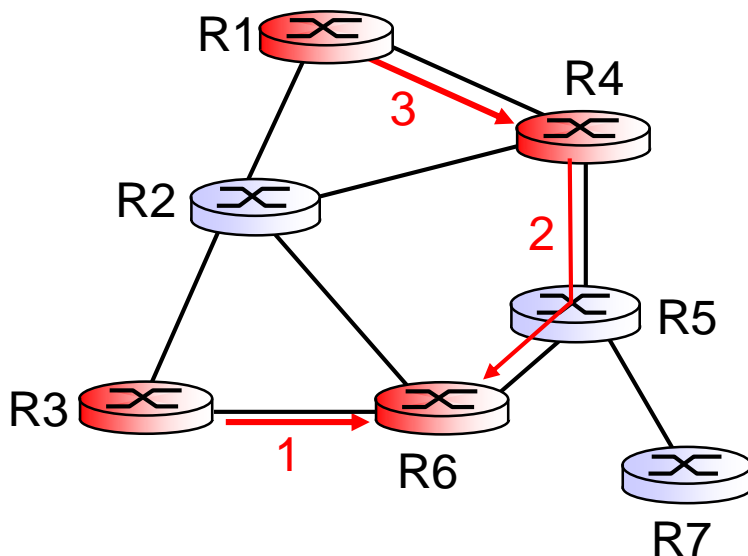
LEGEND

router with attached group member

router with no attached group member

1 → path order in which join messages generated