Internet Protocols EBU5403 The Network Layer (Part II) C2

Michael Chai (michael.chai@qmul.ac.uk)
Richard Clegg (r.clegg@qmul.ac.uk)
Cunhua Pan (c.pan@qmul.ac.uk)

	Part 1	Part 2	Part 3	Part 4
Ecommerce + Telecoms 1	Richard Clegg		Cunhua Pan	
Telecoms 2	Michael Chai			

Network Control Plane: outline

Routing protocols

Routing protocol goal: determine "good" paths (equivalently, routes), from sending hosts to receiving host, through network of routers

- path: sequence of routers packets will traverse in going from given initial source host to given final destination host

- "good": least "cost", "fastest", "least congested"

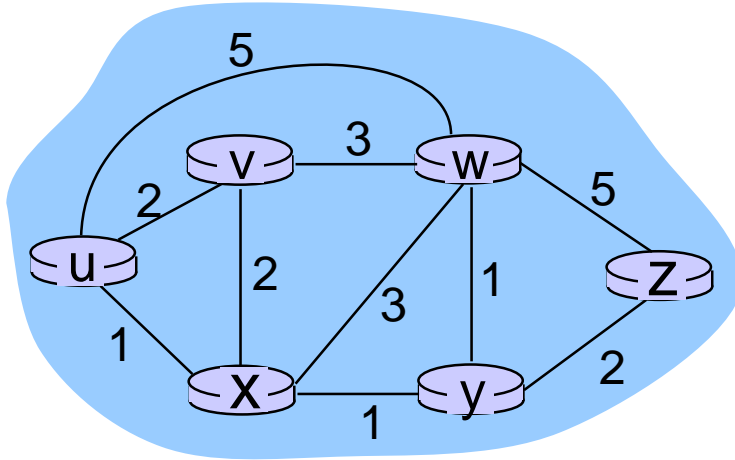- routing: a "top-10" networking challenge!

Graph abstraction of the network
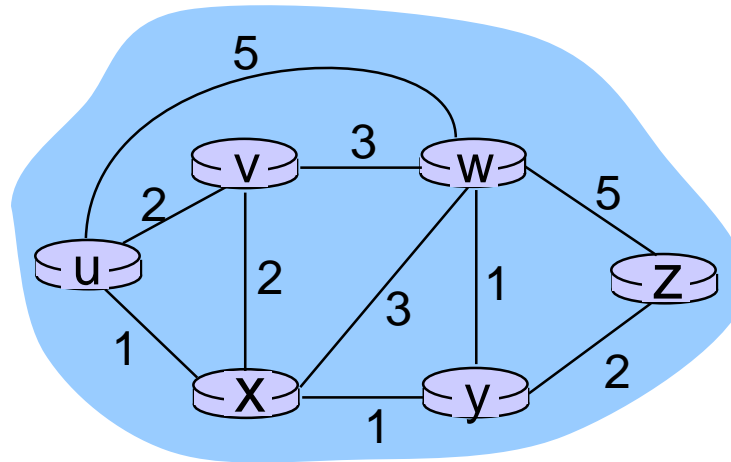
graph: G = (N,E)

N = set of routers = { u, v, w, x, y, z }

E = set of links ={ (u,v), (u,x), (u,w), (v,x), (v,w), (x,w), (x,y), (w,y), (w,z), (y,z) }

aside: graph abstraction is useful in other network contexts, e.g., P2P, where *N* is set of peers and *E* is set of TCP connections

Graph abstraction: costs

$c(x,x') = $ cost of link (x,x')
  e.g., $c(w,z) = 5$

cost could always be 1, or inversely related to bandwidth, or inversely related to congestion

cost of path $(x_1, x_2, x_3,…, x_p) = c(x_1,x_2) + c(x_2,x_3) + … + c(x_{p-1},x_p)$

key question: what is the least-cost path between u and z ?
routing algorithm: algorithm that finds that least cost path

Routing algorithm classification

Q: global or decentralized information?

global: decentralized: a large-scale network

- all routers have complete topology, link cost info
- "link state" algorithms

decentralized:

- router knows physically-connected neighbors, link costs to neighbors
- iterative process of computation, exchange of info with neighbors
- "distance vector" algorithms

Q: static or dynamic?

static:

- routes change slowly over time

dynamic:

- routes change more quickly
  - periodic update
  - in response to link cost changes

Network Control Plane: outline

A link-state routing algorithm

Dijkstra's algorithm

- net topology, link costs known to all nodes
  - accomplished via "link state broadcast"
  - all nodes have same info
- computes least cost paths from one node ('source") to all other nodes
  - gives *forwarding table* for that node
- iterative: after k iterations, know least cost path to k dest.'s

notation:

- $c(x,y)$: link cost from node x to y; = ∞ if not direct neighbors
- $D(v)$: current value of cost of path from source to dest. v
- $p(v)$: predecessor node along path from source to v
- N': set of nodes whose least cost path definitively known

Dijsktra's algorithm

```
1  Initialization:
2    N' = {u}
3    for all nodes v
4      if v adjacent to u
5         then D(v) = c(u,v)
6      else D(v) = ∞
7
8  Loop
9    find w not in N' such that D(w) is a minimum
10     add w to N'
11     update D(v) for all v adjacent to w and not in N' :
12         D(v) = min( D(v), D(w) + c(w,v) )
13     /* new cost to v is either old cost to v or known
14      shortest path cost to w plus cost from w to v */
15  until all nodes in N'
```
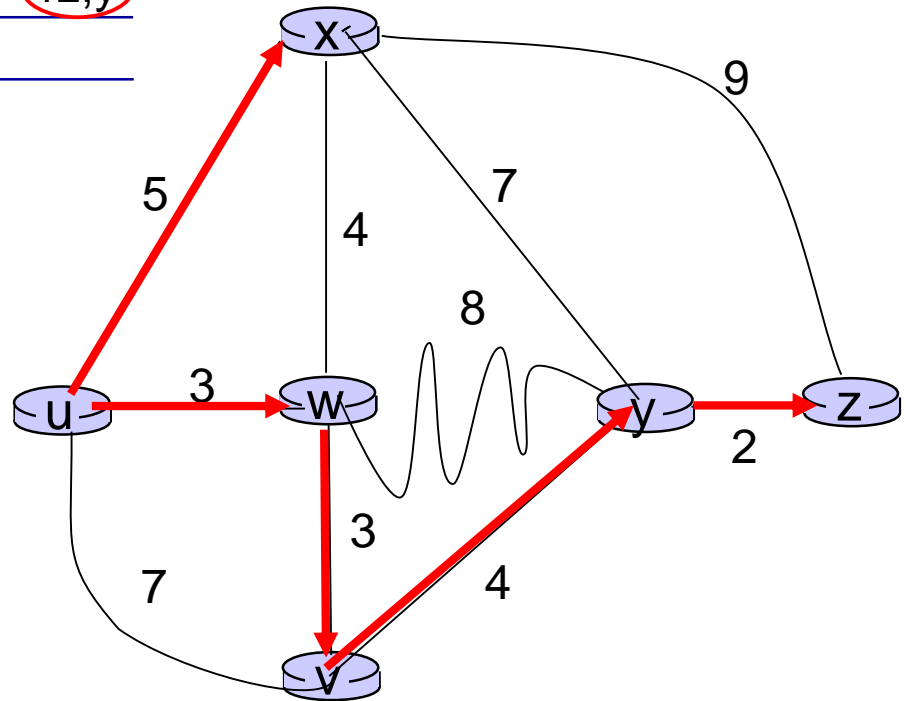
Dijkstra's algorithm: example

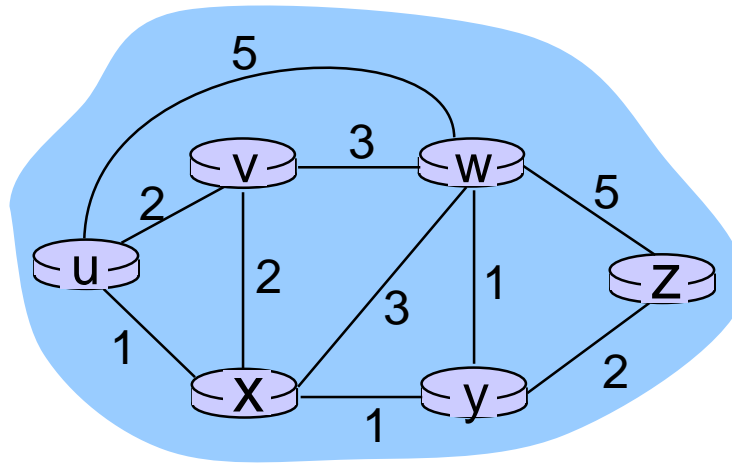| Step	N'	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	(3,u)	5,u	∞	∞
1	uw	6,w		(5,u)	11,w	∞
2	uwx	(6,w)			11,w	14,x
3	uwxv				(10,v)	14,x
4	uwxvy					(12,y)
5	uwxvyz					

notes:

❖ construct shortest path tree by tracing predecessor nodes

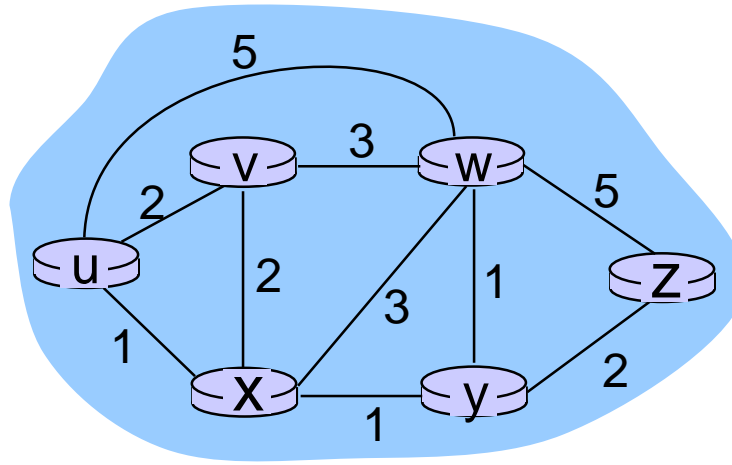❖ ties can exist (can be broken arbitrarily)

Test your understanding

- Use Dijkstra's algorithm to generate the Shortest Path Tree originating from router U and use it to construct the corresponding routing table.
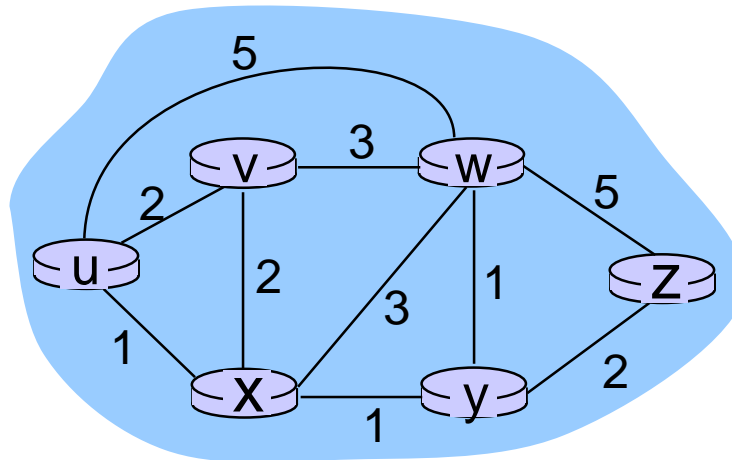
Dijkstra's algorithm: Solution

Step	N'	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	u	2,u	5,u	1,u	∞	∞

Dijkstra's algorithm: Solution

Step	N'	D(v),p(v)	D(w),p(w)	D(x),p(x)	D(y),p(y)	D(z),p(z)
0	u	2,u	5,u	1,u	∞	∞
1	ux	2,u	4,x		2,x	∞
2	uxy	2,u	3,y			4,y
3	uxyv		3,y			4,y
4	uxyvw					4,y
5	uxyvwz					

Dijkstra's algorithm: Solution

resulting shortest-path tree from u:

resulting forwarding table in u:

destination	link
v	(u,v)
x	(u,x)
y	(u,x)
w	(u,x)
z	(u,x)

Network Contol Plane: outline

Distance vector algorithm

Bellman-Ford equation (dynamic programming)

let

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

then

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

cost from neighbor v to destination y

cost to neighbor v

min taken over all neighbors v of x

Bellman-Ford example
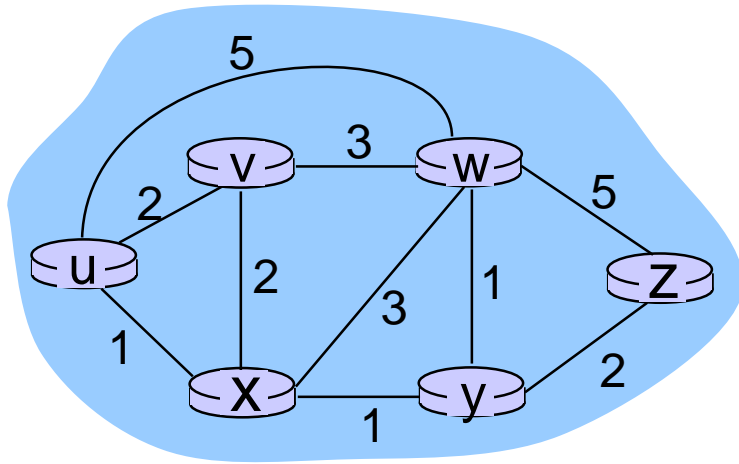
clearly, $d_v(z) = 5$, $d_x(z) = 3$, $d_w(z) = 3$

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 $D_x = [D_x(y): y \in N]$
- node x:
  - knows cost to each neighbor v: $c(x,v)$
  - maintains its neighbors' distance vectors. For each neighbor v, x maintains
    $D_v = [D_v(y): y \in N]$

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

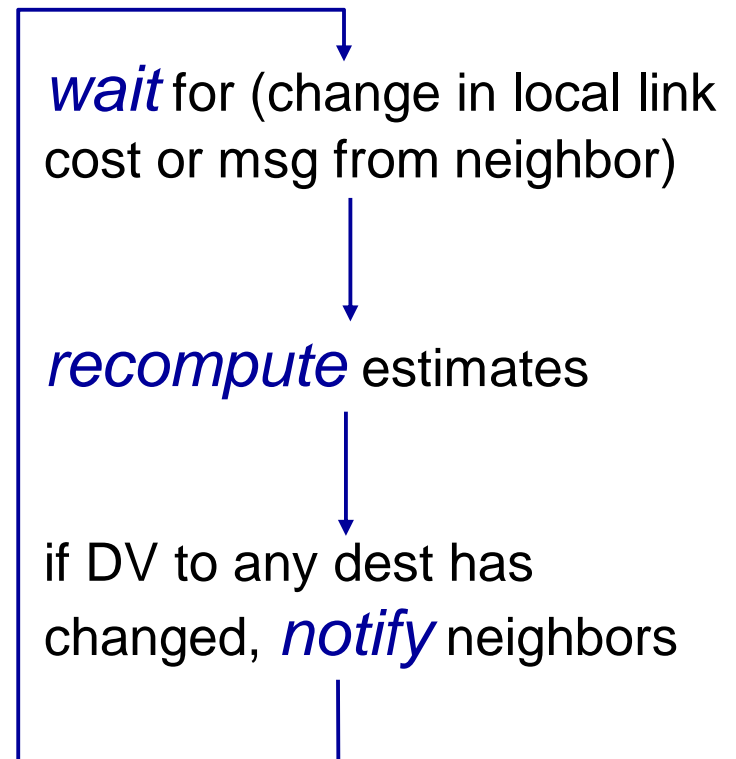*iterative, asynchronous:*
each local iteration caused by:

- 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

node x table

	cost to		
	x	y	z
x	0	2	7
from y	∞	∞	∞
z	∞	∞	∞

node y table

	cost to		
	x	y	z
x	∞	∞	∞
from y	2	0	1
z	∞	∞	∞

node z table

	cost to		
	x	y	z
x	∞	∞	∞
from y	∞	∞	∞
z	7	1	0

time

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

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

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

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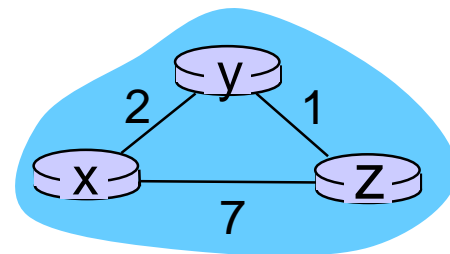
node x table

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

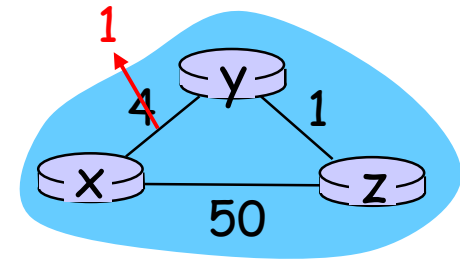| from	x	y	z
x	0	2	3
y	2	0	1
z	3	1	0

time

2, 1, 7 (edge costs between x, y, z)

23

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.

* Check out the online interactive exercises for more examples: http://gaia.cs.umass.edu/kurose_ross/interactive/

Distance vector: link cost changes

link cost changes:

❖ node detects local link cost change

❖ *bad news travels slow -* "count to infinity" problem!
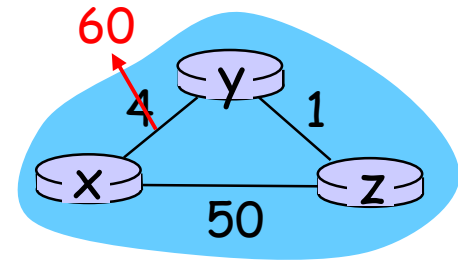
poisoned reverse:

❖ 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)

Test your understanding

Which of the following statement is untrue for Distance Vector algorithm.

A. Routers compute the best path from information passed to them from their neighbour.

B. The least cost route/path is computed by adding distance vectors from router to router.

C. Routers get a common view of entire network topology

D. Frequent, periodic updates; slow convergence.

E. Routers broadcast copies of their routing tables to neighbour routers.

Test your understanding

Which of the following statement is untrue for Distance Vector algorithm.

A. Routers compute the best path from information passed to them from their neighbour.

B. The least cost route/path is computed by adding distance vectors from router to router.

C. Routers get a common view of entire network topology (not true – this is true for Link state algorithm)

D. Frequent, periodic updates; slow convergence.

E. Routers broadcast copies of their routing tables to neighbour routers.

Network Control Plane: outline

Making routing scalable

our routing study thus far - idealized
- all routers identical
- network "flat"

… *not* true in practice

scale: with billions of 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

Internet approach to scalable routing

aggregate routers into regions known as "autonomous systems" (AS) (a.k.a. "domains") – big areas of internet

think of a large university or company or ISP.

intra-AS routing

- routing among hosts, routers in same AS ("network")
- all routers in AS must run *same* intra-domain protocol
- routers in *different* AS can run *different* intra-domain routing protocol
- gateway router: at "edge" of its own AS, has link(s) to router(s) in other AS'es

inter-AS routing

- routing among AS'es
- gateways perform inter-domain routing (as well as intra-domain routing)

Interconnected ASes
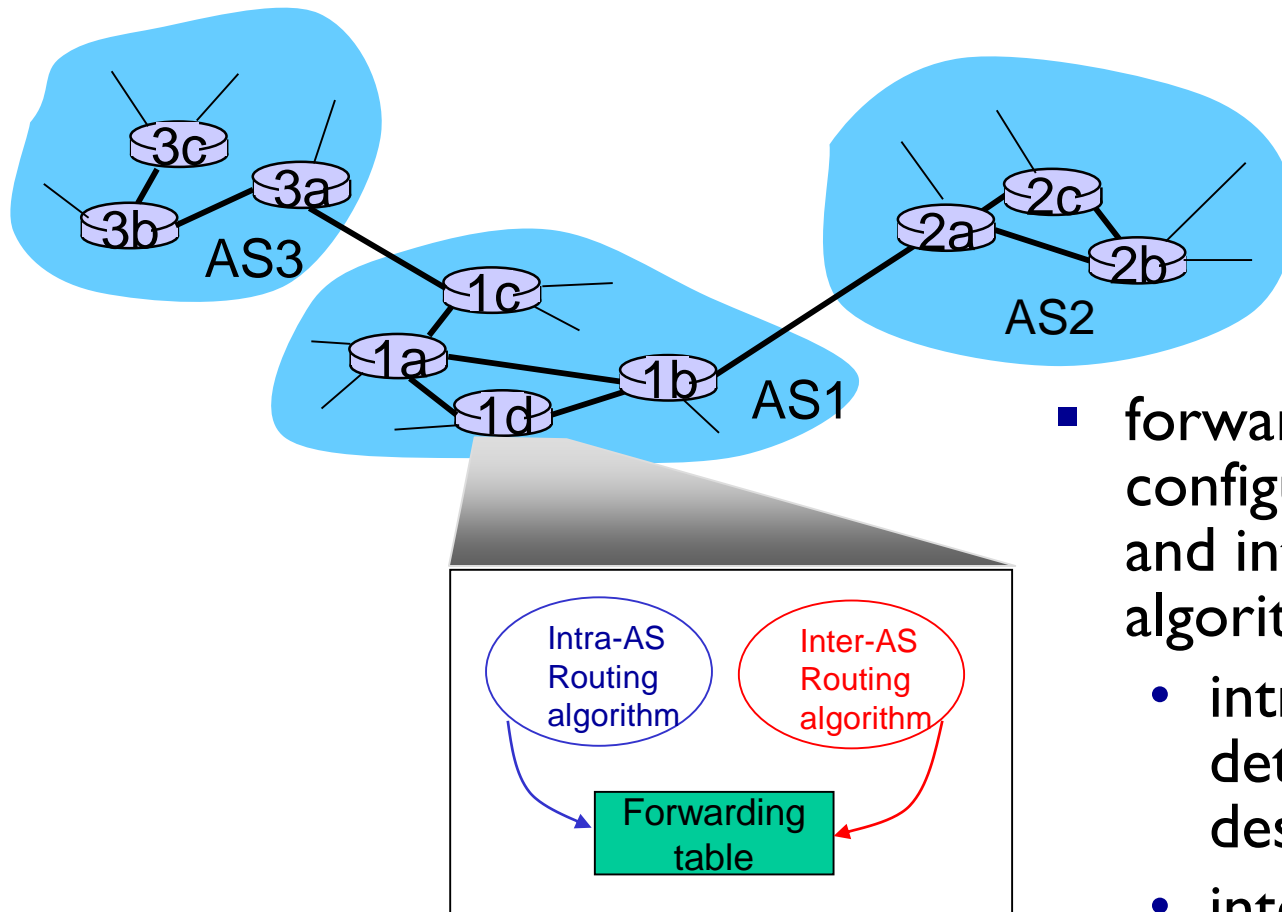
- forwarding table configured by both intra- and inter-AS routing algorithm
  - intra-AS routing determine entries for destinations within AS
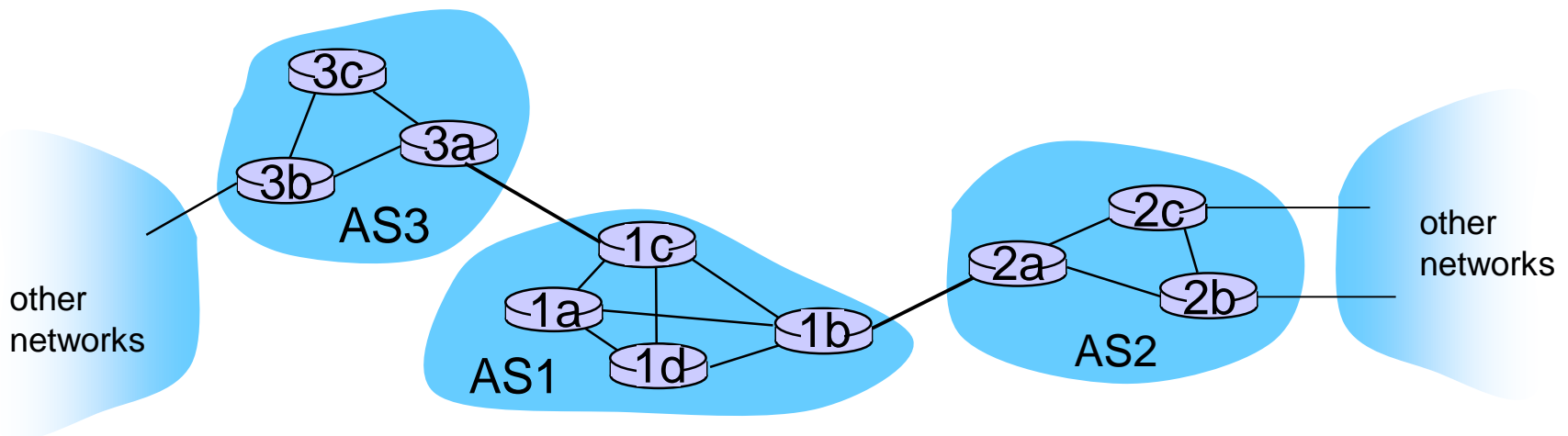 - inter-AS & intra-AS determine entries for external destinations

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!

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 (IS-IS protocol essentially same as OSPF)
  - IGRP: Interior Gateway Routing Protocol (Cisco proprietary for decades, until 2016)

Test your understanding

An Autonomous System …

A. Is a backbone network of core routers.

B. Is a group of networks and routers under a single administrative authority

C. Is a set of networks that operate in isolation from the rest of the Internet.

D. Is a computer network that controls a mobile robot.

Test your understanding

An Autonomous System …

A. Is a backbone network of core routers.

B. Is a group of networks and routers under a single administrative authority

C. Is a set of networks that operate in isolation from the rest of the Internet.

D. Is a computer network that controls a mobile robot.

OSPF (Open Shortest Path First)

- "open": publicly available
- uses link-state algorithm
  - link state packet dissemination
  - topology map at each node
  - route computation using Dijkstra's algorithm
- router floods OSPF link-state advertisements to all other routers in *entire* AS
  - carried in OSPF messages directly over IP (rather than TCP or UDP)
  - link state: for each attached link

OSPF "advanced" features

- *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 multi-cast support:
  - Multicast OSPF (MOSPF) uses same topology data base as OSPF
- hierarchical OSPF in large domains.

Test your understanding

Open Shortest Path First (OSPF) includes:

A.   A flooding mechanism
B.   Dijkstra's algorithm
C.   Routers get a common view of entire network topology
D.   All of the above

Test your understanding

Open Shortest Path First (OSPF) includes:

A. A flooding mechanism
B. Dijkstra's algorithm
C. Routers get a common view of entire network topology
D. All of the above

What have we learned?

- **Link State Routing**
  - Dijkstra algorithm
  - Routers each learn a map of the network
- **Distance Vector Routing**
  - Bellman-Ford algorithm
  - Routers each learn the distance to all other routers
- **Routing in Autonomous systems**
  - Intra AS (routing within one system)
  - Inter AS (routing between different systems)
- **OSPF (Open Shortest Path First)**
  - Link vector routing for intra AS problem