Lecture 23

- Sections 5.2.2, 5.3, 5.4
- Routing protocols
  - Distance vector routing protocol
- Intra-Autonomous Systems routing (OSPF)
- Routing among the ISPs: BGP

Distance Vector Algorithm (5)

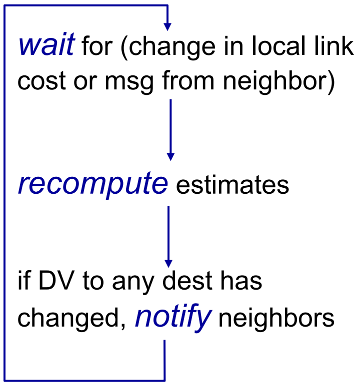
Distributed:

- each node notifies neighbors *only* when its DV changes
  - neighbors then notify their neighbors if necessary

Iterative, asynchronous: each local iteration caused by:

- local link cost change
- DV update message from neighbor

Each node:



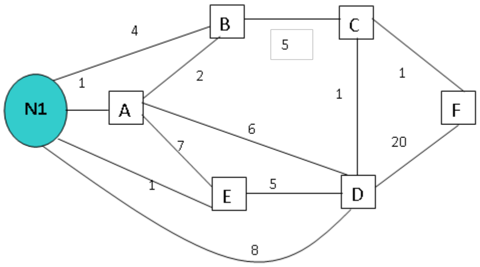
wait for (change in local link cost or msg from neighbor)

recompute estimates

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

Distance Vector Routing: Example

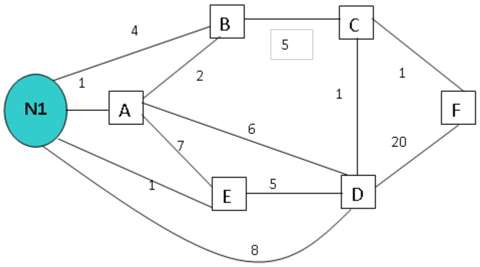
(X,i):
X: next hop to N1
i: distance to N1



Iter.	A	B	C	D	E	F
0	(-,1)	(-,4)	(•,∞)	(-,8)	(-,1)	(•,∞)

Distance Vector Routing: Example

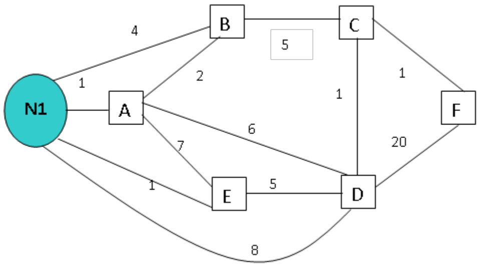
(X,i):
X: next hop to N1
i: distance to N1



Iter.	A	B	C	D	E	F
0	(-,1)	(-,4)	(•,∞)	(-,8)	(-,1)	(•,∞)
1	(-,1)	(A,3)	(B,9)	(E,6)	(-,1)	(D,28)

Distance Vector Routing: Example

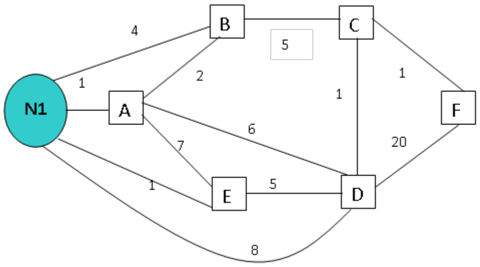
(X,i):
X: next hop to N1
i: distance to N1



Iter.	A	B	C	D	E	F
0	(-,1)	(-,4)	(•,∞)	(-,8)	(-,1)	(•,∞)
1	(-,1)	(A,3)	(B,9)	(E,6)	(-,1)	(D,28)

Distance Vector Routing: Example

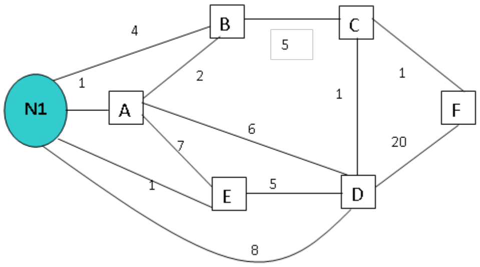
(X,i):
X: next hop to N1
i: distance to N1



Iter.	A	B	C	D	E	F
0	(-,1)	(-,4)	(•,∞)	(-,8)	(-,1)	(•,∞)
1	(-,1)	(A,3)	(B,9)	(E,6)	(-,1)	(D,28)

Distance Vector Routing: Example

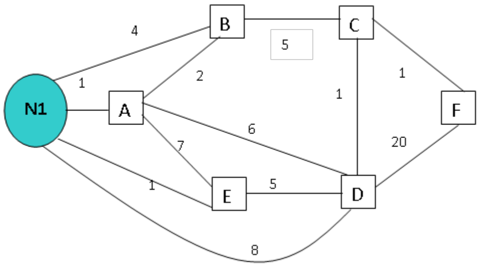
(X,i):
X: next hop to N1
i: distance to N1



Iter.	A	B	C	D	E	F
0	(-,1)	(-,4)	(•,∞)	(-,8)	(-,1)	(•,∞)
1	(-,1)	(A,3)	(B,9)	(E,6)	(-,1)	(D,28)
2	(-,1)	(A,3)	(D,7)	(E,6)	(-,1)	(C,10)

Distance Vector Routing: Example

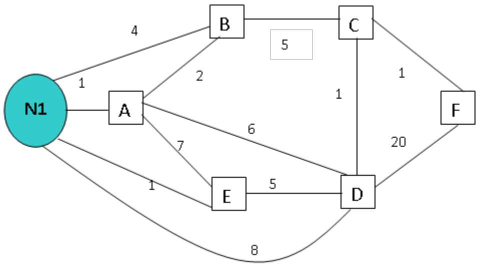
(X,i):
X: next hop to N1
i: distance to N1



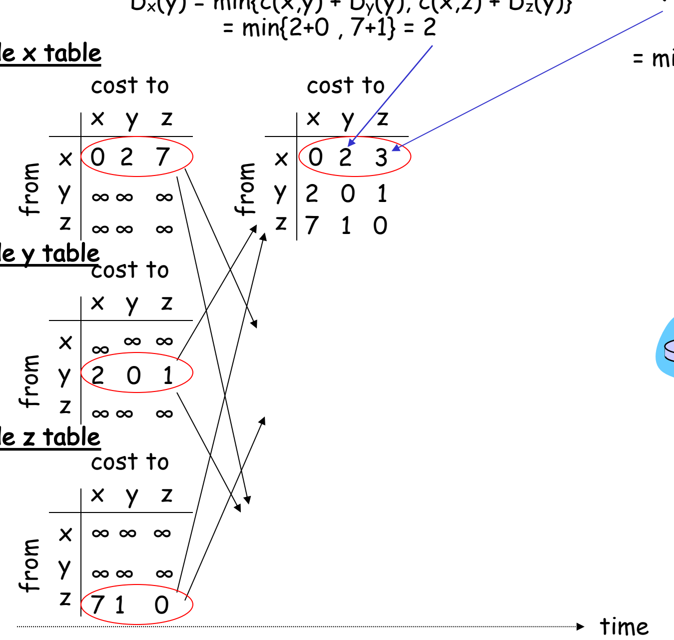
Iter.	A	B	C	D	E	F
0	(-,1)	(-,4)	(•,∞)	(-,8)	(-,1)	(•,∞)
1	(-,1)	(A,3)	(B,9)	(E,6)	(-,1)	(D,28)
2	(-,1)	(A,3)	(D,7)	(E,6)	(-,1)	(C,10)

Distance Vector Routing: Example

(X,i):
X: next hop to N1
i: distance to N1



Iter.	A	B	C	D	E	F
0	(-,1)	(-,4)	(•,∞)	(-,8)	(-,1)	(•,∞)
1	(-,1)	(A,3)	(B,9)	(E,6)	(-,1)	(D,28)
2	(-,1)	(A,3)	(D,7)	(E,6)	(-,1)	(C,10)
3	(-,1)	(A,3)	(D,7)	(E,6)	(-,1)	(C,8)



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

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

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

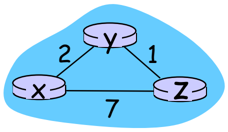
from		x	y	z
	x	∞	∞	∞
	y	2	0	1
	z	∞	∞	∞

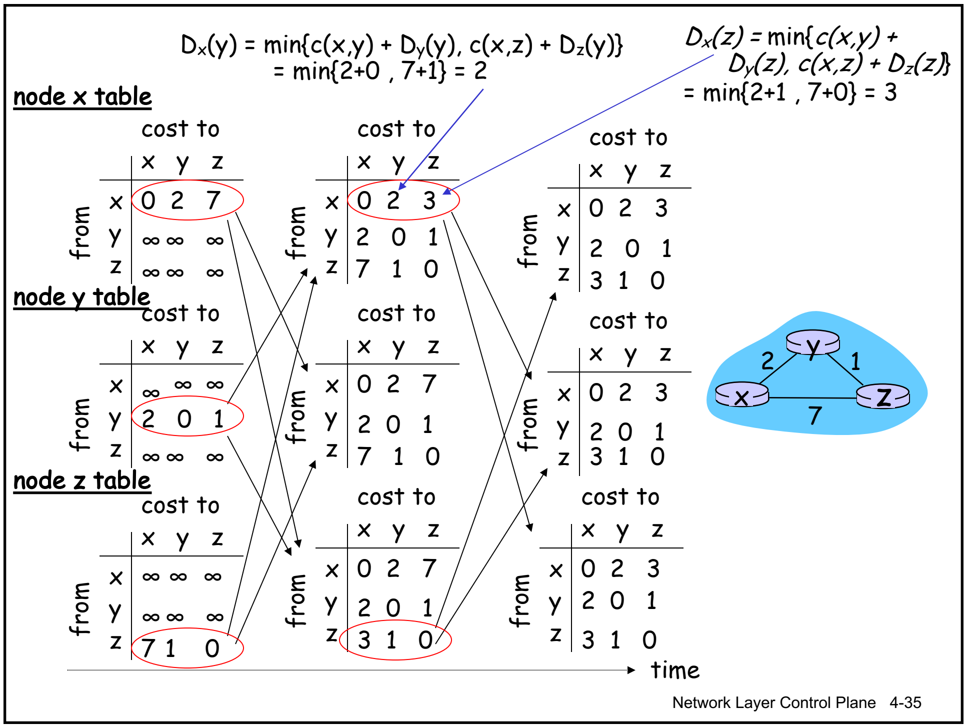
node z table

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

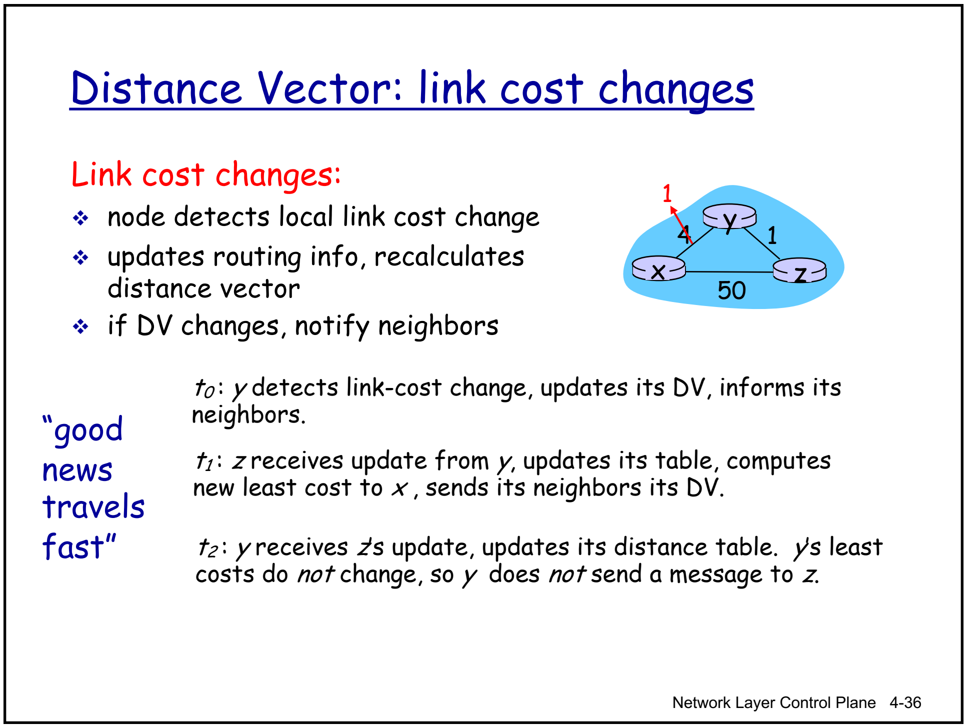
from	x	y	z
x	∞	∞	∞
y	∞	∞	∞
z	7	1	0

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



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.

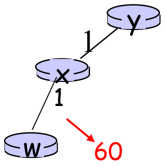
t_1: *z* receives update from *y*, updates its table, computes new least cost to *x* , sends its neighbors its DV.

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

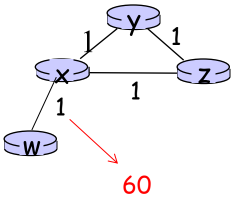
Link cost changes:

- **bad news travels slow** - "count to infinity" problem!
- Many iterations before algorithm stabilizes: how many?



Poisoned reverse:

- If Y routes through X to get to W :
  - Y tells X its (Y's) distance to W is infinite (so X won't route to W via Y)
- 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

Speed of Convergence

- LS: exactly n iterations
- 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 through network

Chapter 5: Network Layer

5. 1 Introduction

5.2 Routing algorithms

- 5.2.1 link-state routing algorithm
- 5.2.2 distance vector routing

5.3 Intra-Autonomous System (AS) routing: OSPF

Internet approach to scalable routing

aggregate routers into regions known as "autonomous systems" (AS) (a.k.a. "domains")

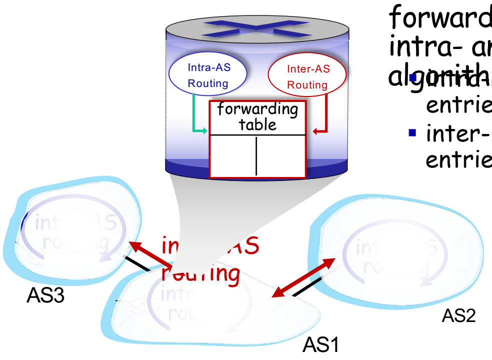
intra-AS (aka "intra-domain"): routing among *within same AS ("network")*

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

inter-AS (aka "inter-domain"): routing *among* AS'es

- gateways perform inter-domain routing (as well as intra-domain routing)

Interconnected ASes



forwarding table configured by intra- and inter-AS routing algorithms

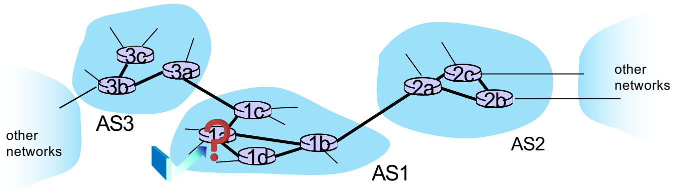
- intra-AS routing determine entries for destinations within AS
- inter-AS & intra-AS determine entries for external destinations

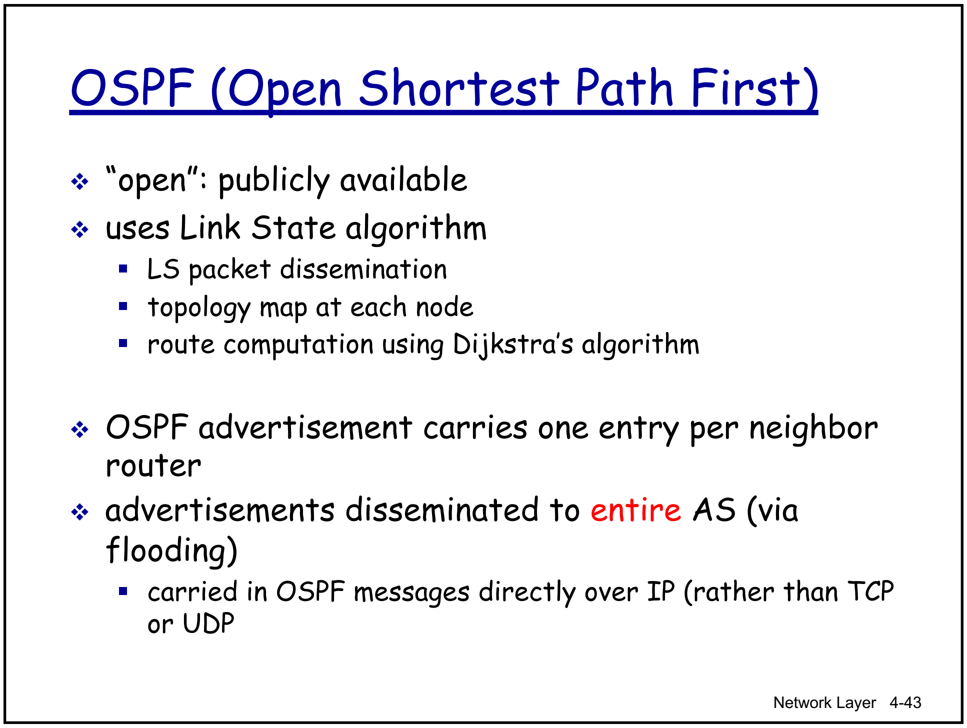
Inter-AS routing: a role in intradomain forwarding

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

AS1 inter-domain routing must:

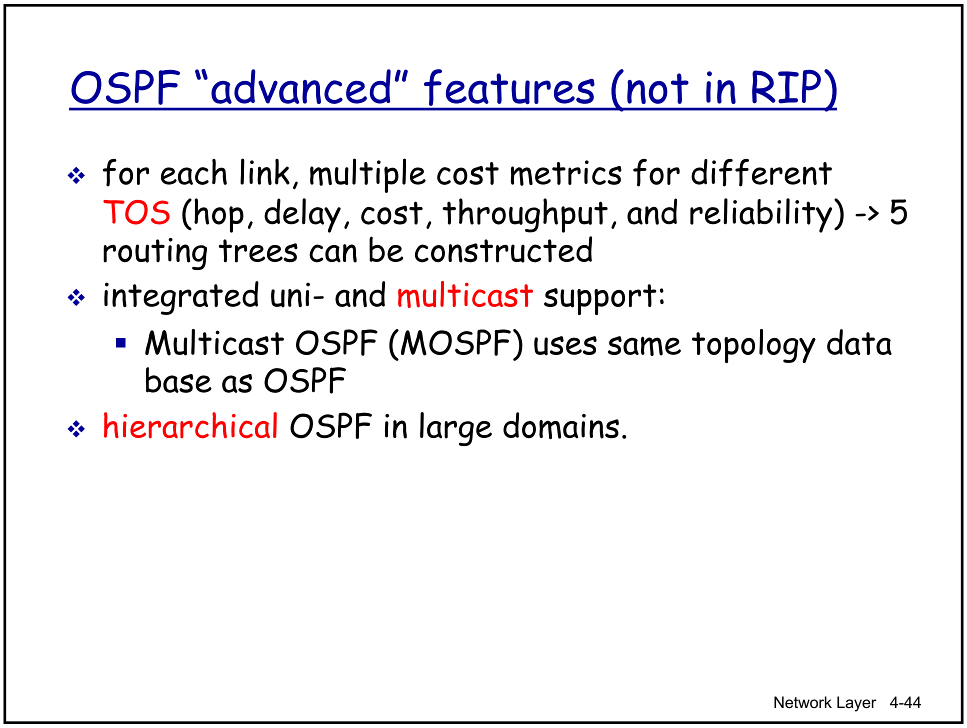
1. learn which destinations reachable through AS2, which through AS3
2. propagate this reachability info to all routers in AS1





OSPF (Open Shortest Path First)

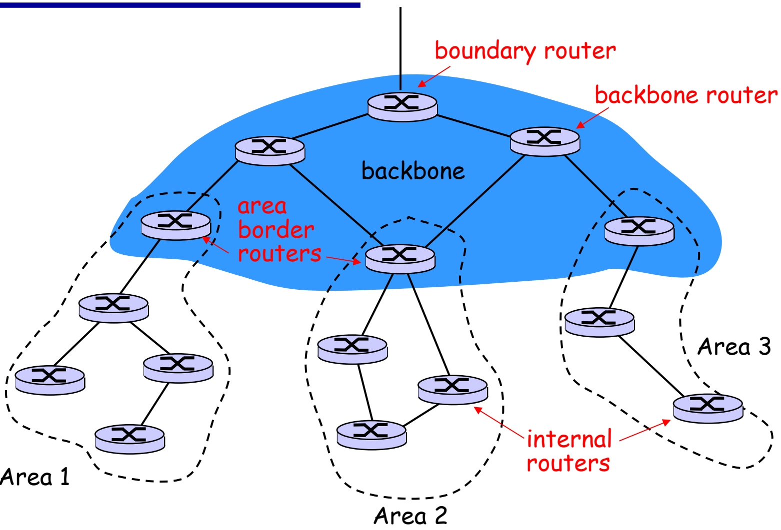
- "open": publicly available
- uses Link State algorithm
  - LS packet dissemination
  - topology map at each node
  - route computation using Dijkstra's algorithm
- OSPF advertisement carries one entry per neighbor router
- advertisements disseminated to entire AS (via flooding)
  - carried in OSPF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



OSPF "advanced" features (not in RIP)

- for each link, multiple cost metrics for different TOS (hop, delay, cost, throughput, and reliability) -> 5 routing trees can be constructed
- integrated uni- and multicast support:
  - Multicast OSPF (MOSPF) uses same topology data base as OSPF
- hierarchical OSPF in large domains.

Hierarchical OSPF



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.

Chapter 5: Network Layer

5. 1 Introduction

5.2 Routing algorithms

- 5.2.1 link-state routing algorithm
- 5.2.2 distance vector routing

5.3 Intra-Autonomous System (AS) routing: OSPF

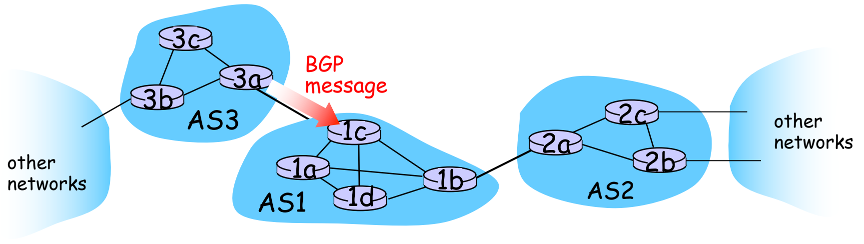
5.4 Routing among the ISPs

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.

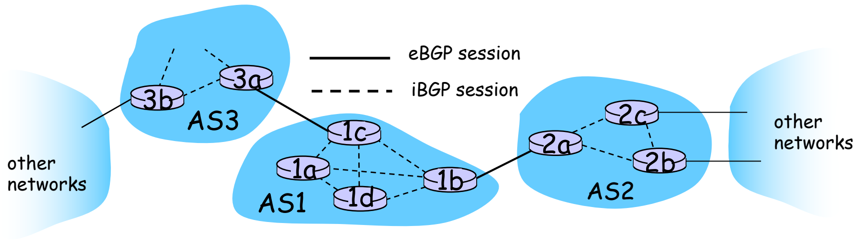
BGP basics

- **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 & 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

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

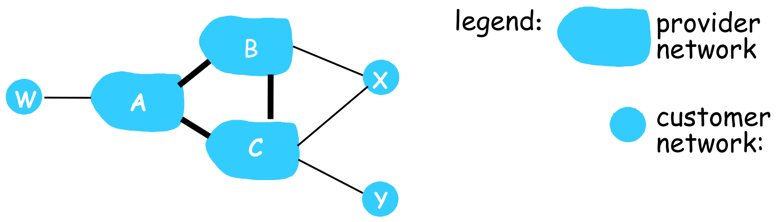
Scale:

- hierarchical routing saves table size, reduced update traffic

Performance:

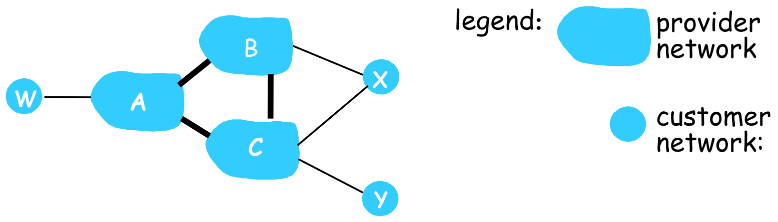
- Intra-AS: can focus on performance
- Inter-AS: policy may dominate over performance

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)



- 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!