CS120: Computer Networks

Lecture 10. RIP and OSPF

Zhice Yang

Forwarding with IP Address

- Router Behavior
  - if IP.network == PortX.IP.network
    - forward to the host connected to PortX
  - else
    - forward to the router (Which? given by routing alg.)

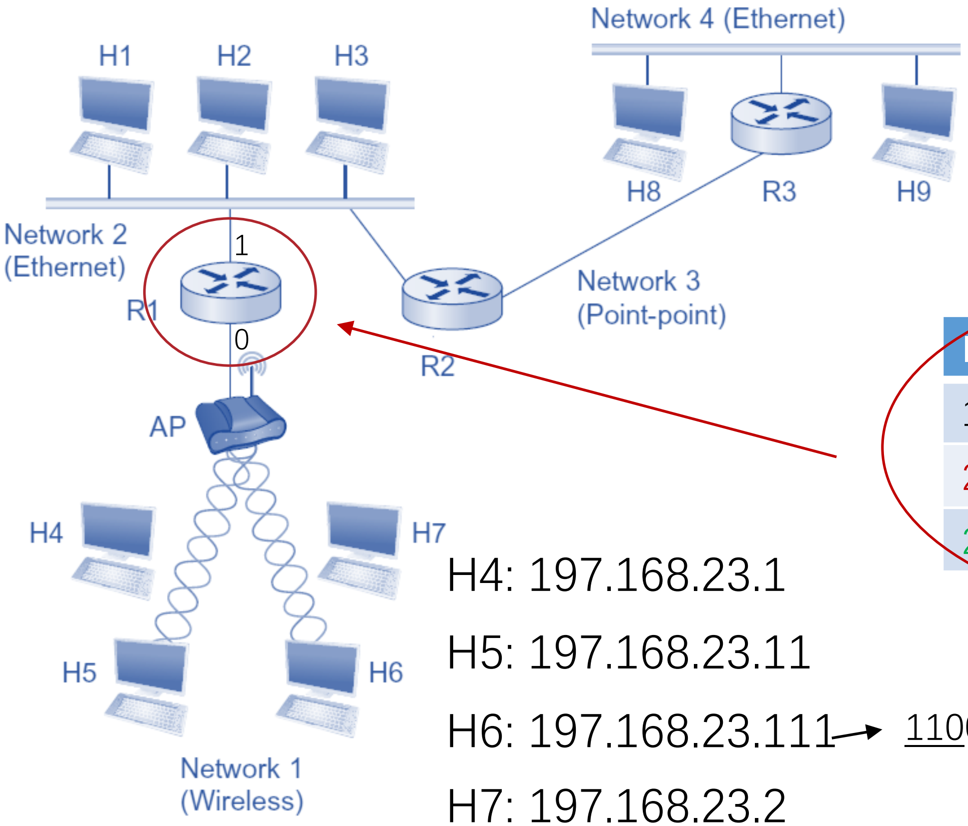
NO need to check the host part

H1: 200.155.11.5

H2: 200.155.11.3

H3: 200.155.11.2

11001000.10011011.00001011.XXXXXXXX



H8: 210.168.1.10

H9: 210.168.1.200

11010010.10101000.00000001.XXXXXXXX

IP	Next
197.168.23.*	Interface 0
200.155.11.*	Interface 1
210.168.1.*	R2

How?

H4: 197.168.23.1

H5: 197.168.23.11

H6: 197.168.23.111 → 11000101.10101000.00010111.XXXXXXXX

H7: 197.168.23.2

Routing Protocols

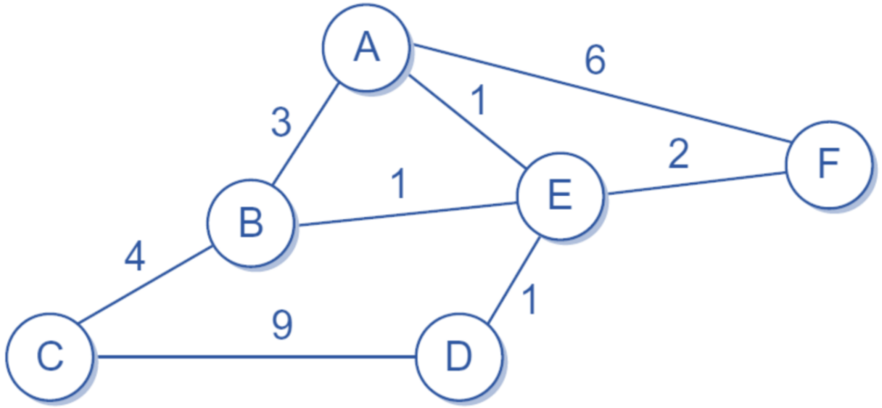
- Routing Information Protocol (RIP)
  - Algorithm: Distance Vector
- Open Shortest Path First (OSPF)
  - Algorithm: Link State
- Border Gateway Protocol (BGP)

Intradomain Routing Protocol

Interdomain Routing Protocol

Network as a Graph

- The basic problem of routing is to find the low(est)-cost path between any two nodes
  - Static approach has several shortcomings
    - Can not handle node or link failures
    - Can not handle addition of new nodes or links
    - Edge costs cannot change
  - Centralized solution does not scale
  - ➢ Distributed and dynamic protocol



Distance Vector Algorithm

- Bellman-Ford equation

let

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

then

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

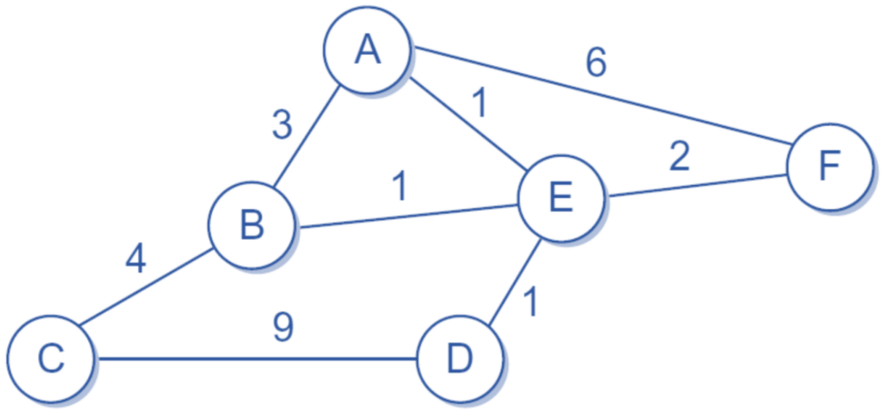
lowest-cost from neighbor v to destination y

cost to neighbor v

min taken over all neighbors v of x

Example

- $d_C(A) = ?$
  - $d_C(A) = \min(d_B(A) + 4, d_D(A) + 9) = 6$
  - $d_B(A) = 2$
  - $d_D(A) = 2$



Distance Vector Algorithm

- x maintains its distance vector estimate $\boldsymbol{D_x(y)} = \{D_x(y): y \in N\}$
- x knows:
  - cost to each neighbor v: $c(x, v)$
  - neighbors' distance vectors estimate: $\boldsymbol{D_v(y)} = \{D_v(y): y \in N\}$
- Algorithm idea:
  - From time-to-time, each node sends its own distance vector estimate to neighbors
  - When x receives new distance vector estimates from neighbors, it updates its own distance vector estimate using Bellman-Ford equation
  - Under minor, natural conditions, the estimate $D_x(y)$ will converge to the actual lowest cost $d_x(y)$

Distance Vector Algorithm

y	$D_A(y)$
A	0
B	inf
C	inf
D	inf
E	inf
F	inf
G	inf

y	$D_B(y)$
A	inf
B	0
C	inf
D	inf
E	inf
F	inf
G	inf

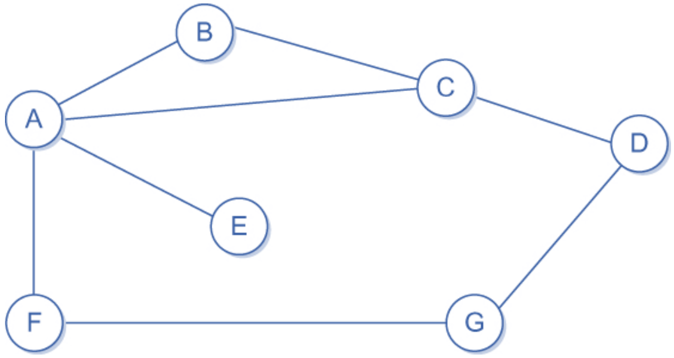
y	$D_C(y)$
A	inf
B	inf
C	0
D	inf
E	inf
F	inf
G	inf

y	$D_D(y)$
A	inf
B	inf
C	inf
D	0
E	inf
F	inf
G	inf

y	$D_E(y)$
A	inf
B	inf
C	inf
D	inf
E	0
F	inf
G	inf

y	$D_F(y)$
A	inf
B	inf
C	inf
D	inf
E	inf
F	0
G	inf

y	$D_G(y)$
A	inf
B	inf
C	inf
D	inf
E	inf
F	inf
G	0



Distance Vector Algorithm

y	$D_A(y)$
A	0
B	1
C	1
D	inf
E	1
F	1
G	inf

y	$D_B(y)$
A	1
B	0
C	1
D	inf
E	inf
F	inf
G	inf

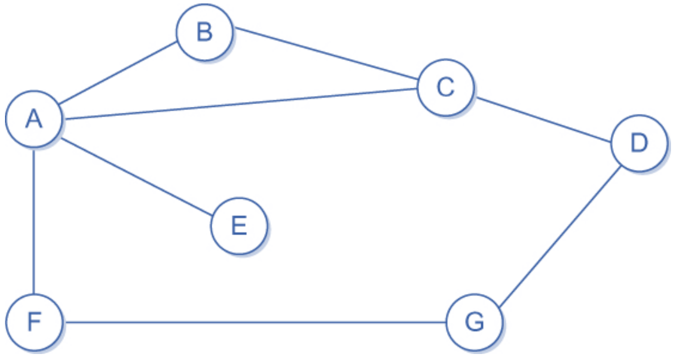
y	$D_C(y)$
A	1
B	1
C	0
D	1
E	inf
F	inf
G	inf

y	$D_D(y)$
A	inf
B	inf
C	1
D	0
E	inf
F	inf
G	1

y	$D_E(y)$
A	1
B	inf
C	inf
D	inf
E	0
F	inf
G	inf

y	$D_F(y)$
A	1
B	inf
C	inf
D	inf
E	inf
F	0
G	1

y	$D_G(y)$
A	inf
B	inf
C	inf
D	1
E	inf
F	1
G	0



Distance Vector Algorithm

- Every T seconds each router sends its table to its neighbor
- Each router then updates its table based on the new information

Distance Vector Algorithm

y	$D_A(y)$
A	0
B	1
C	1
D	inf
E	1
F	1
G	inf

y	$D_B(y)$
A	1
B	0
C	1
D	inf
E	inf
F	inf
G	inf

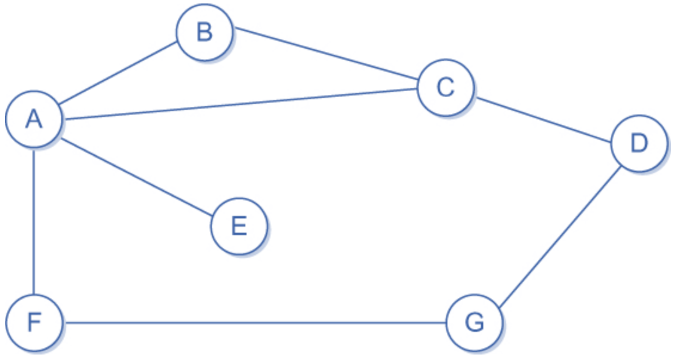
y	$D_C(y)$
A	1
B	1
C	0
D	1
E	inf
F	inf
G	inf

y	$D_D(y)$
A	inf
B	inf
C	1
D	0
E	inf
F	inf
G	1

y	$D_E(y)$
A	1
B	inf
C	inf
D	inf
E	0
F	inf
G	inf

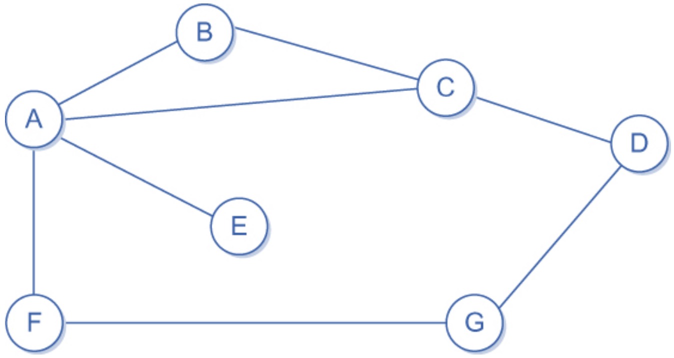
y	$D_F(y)$
A	1
B	inf
C	inf
D	inf
E	inf
F	0
G	1

y	$D_G(y)$
A	inf
B	inf
C	inf
D	1
E	inf
F	1
G	0



Distance Vector Algorithm

y	$D_A(y)$	$D_B(y)$	$D_C(y)$	$D_D(y)$	$D_E(y)$	$D_F(y)$	$D_G(y)$
A	0	1	1	inf	1	1	inf
B	1	0	1	inf	inf	inf	inf
C	1	1	0	1	inf	inf	inf
D	inf	inf	1	0	inf	inf	1
E	1	inf	inf	inf	0	inf	inf
F	1	inf	inf	inf	inf	0	1
G	inf	inf	inf	1	inf	1	0



Distance Vector Algorithm

y	$D_A(y)$
A	0
B	1
C	1
D	2
E	1
F	1
G	inf

y	$D_B(y)$
A	1
B	0
C	1
D	inf
E	inf
F	inf
G	inf

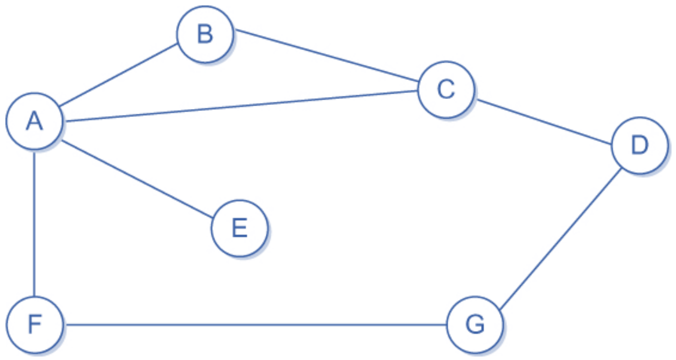
y	$D_C(y)$
A	1
B	1
C	0
D	1
E	inf
F	inf
G	inf

y	$D_D(y)$
A	inf
B	inf
C	1
D	0
E	inf
F	inf
G	1

y	$D_E(y)$
A	1
B	inf
C	inf
D	inf
E	0
F	inf
G	inf

y	$D_F(y)$
A	1
B	inf
C	inf
D	inf
E	inf
F	0
G	1

y	$D_G(y)$
A	inf
B	inf
C	inf
D	1
E	inf
F	1
G	0



Distance Vector Algorithm

y	$D_A(y)$
A	0
B	1
C	1
D	2
E	1
F	1
G	inf

y	$D_B(y)$
A	1
B	0
C	1
D	inf
E	inf
F	inf
G	inf

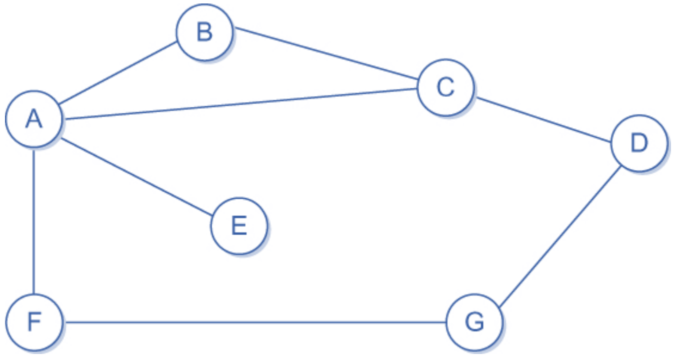
y	$D_C(y)$
A	1
B	1
C	0
D	1
E	inf
F	inf
G	inf

y	$D_D(y)$
A	inf
B	inf
C	1
D	0
E	inf
F	inf
G	1

y	$D_E(y)$
A	1
B	inf
C	inf
D	inf
E	0
F	inf
G	inf

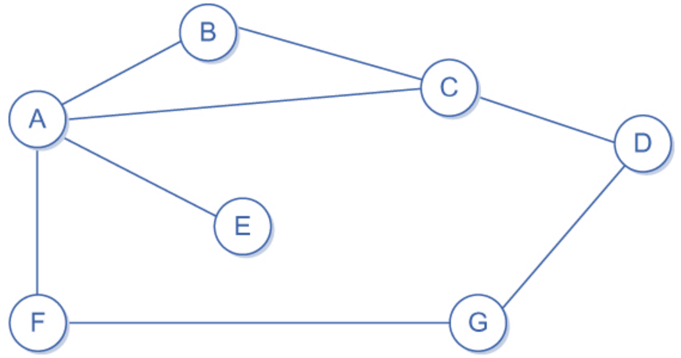
y	$D_F(y)$
A	1
B	inf
C	inf
D	inf
E	inf
F	0
G	1

y	$D_G(y)$
A	inf
B	inf
C	inf
D	1
E	inf
F	1
G	0

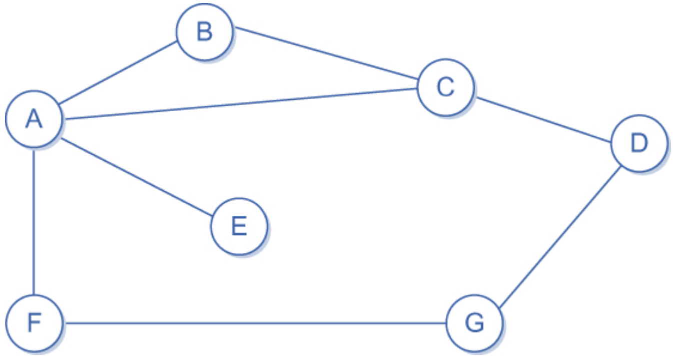


Distance Vector Algorithm

y	$D_A(y)$	$D_B(y)$	$D_C(y)$	$D_D(y)$	$D_E(y)$	$D_F(y)$	$D_G(y)$
A	0	1	1	inf	1	1	inf
B	1	0	1	inf	inf	inf	inf
C	1	1	0	1	inf	inf	inf
D	2	inf	1	0	inf	inf	1
E	1	inf	inf	inf	0	inf	inf
F	1	inf	inf	inf	inf	0	1
G	2	inf	inf	1	inf	1	0



Distance Vector Algorithm



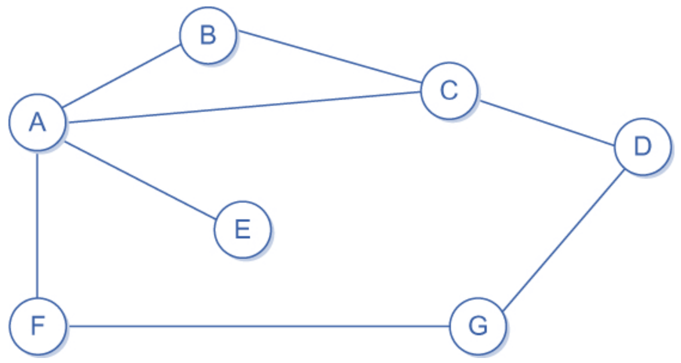
Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	3
C	1	1	0	1	2	2	2
D	2	2	1	0	3	2	1
E	1	2	2	3	0	2	3
F	1	2	2	2	2	0	1
G	2	3	2	1	3	1	0

y	$D_A(y)$	via
A	0	A
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	2	F

Distance Vector Algorithm

- Good News Travels Fast

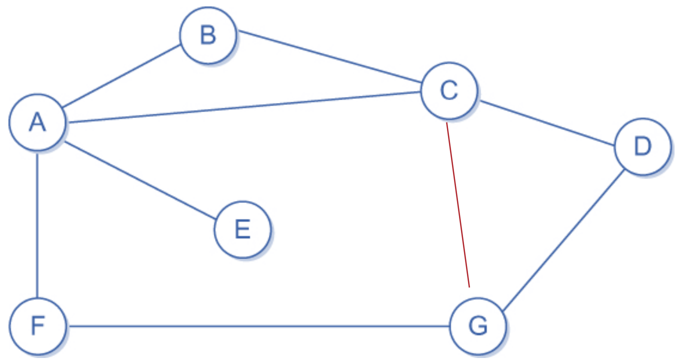
Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	3
C	1	1	0	1	2	2	2
D	2	2	1	0	3	2	1
E	1	2	2	3	0	2	3
F	1	2	2	2	2	0	1
G	2	3	2	1	3	1	0



Distance Vector Algorithm

- Good News Travels Fast

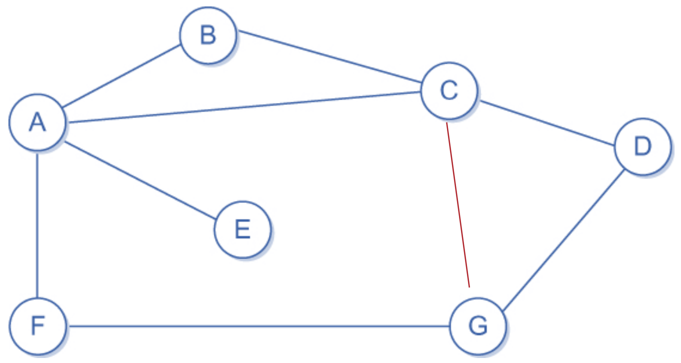
Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	3
C	1	1	0	1	2	2	$\boxed{1}$
D	2	2	1	0	3	2	1
E	1	2	2	3	0	2	3
F	1	2	2	2	2	0	1
G	2	3	$\boxed{1}$	1	3	1	0



Distance Vector Algorithm

- Good News Travels Fast

Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	2
C	1	1	0	1	2	2	1
D	2	2	1	0	3	2	1
E	1	2	2	3	0	2	3
F	1	2	2	2	2	0	1
G	2	2	1	1	3	1	0

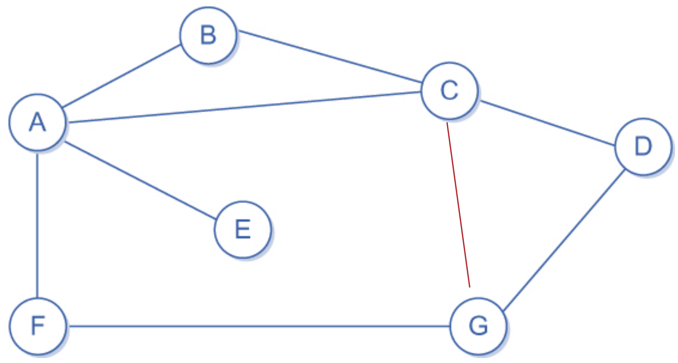


CS 120 LEC 10

Distance Vector Algorithm

- Good News Travels Fast

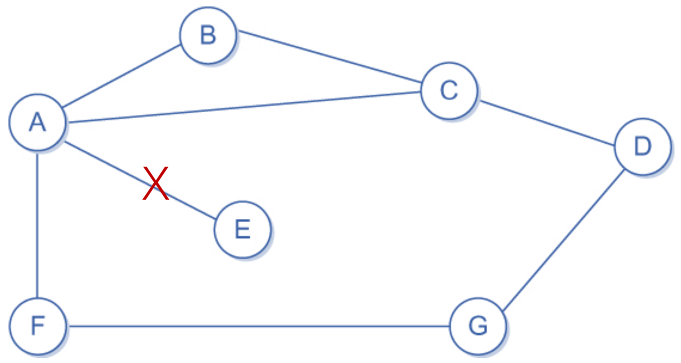
Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	2
C	1	1	0	1	2	2	1
D	2	2	1	0	3	2	1
E	1	2	2	3	0	2	3
F	1	2	2	2	2	0	1
G	2	2	1	1	3	1	0



Distance Vector Algorithm

- Bad News Travels Slow

Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	3
C	1	1	0	1	2	2	2
D	2	2	1	0	3	2	1
E	inf	2	2	3	0	2	3
F	1	2	2	2	2	0	1
G	2	3	2	1	3	1	0

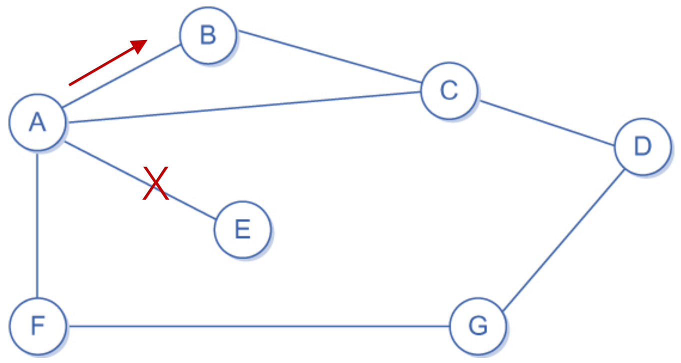


CS 120 LEC 10

Distance Vector Algorithm

- Bad News Travels Slow

Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	3
C	1	1	0	1	2	2	2
D	2	2	1	0	3	2	1
E	$\boxed{\text{inf}}$	$\boxed{\text{inf}}$	2	3	0	2	3
F	1	2	2	2	2	0	1
G	2	3	2	1	3	1	0

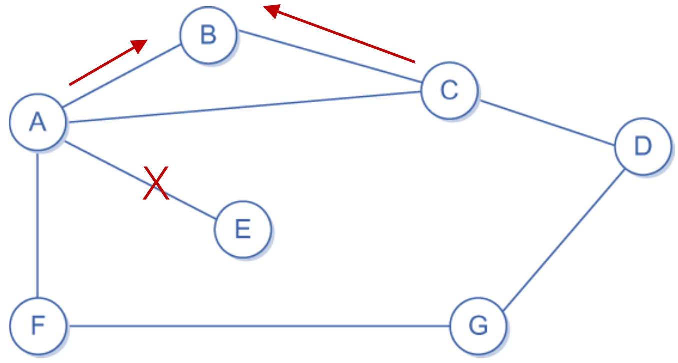


CS 120 LEC 10

Distance Vector Algorithm

- Bad News Travels Slow

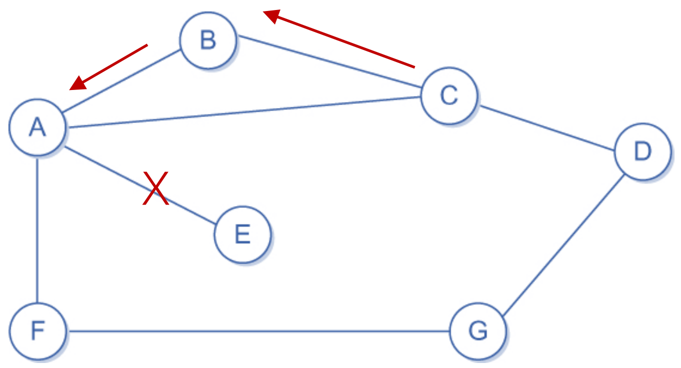
Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	3
C	1	1	0	1	2	2	2
D	2	2	1	0	3	2	1
E	inf	3	2	3	0	2	3
F	1	2	2	2	2	0	1
G	2	3	2	1	3	1	0



Distance Vector Algorithm

- Bad News Travels Slow

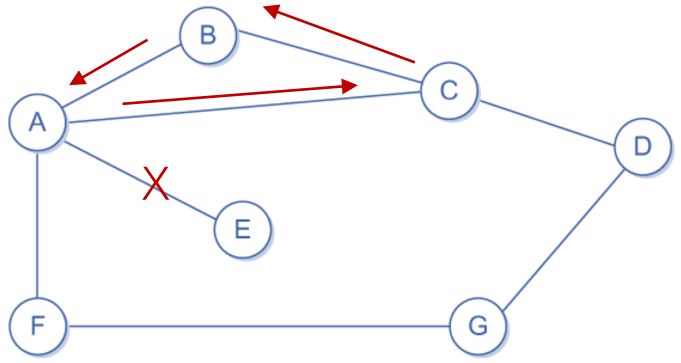
Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	3
C	1	1	0	1	2	2	2
D	2	2	1	0	3	2	1
E	4	3	2	3	0	2	3
F	1	2	2	2	2	0	1
G	2	3	2	1	3	1	0



Distance Vector Algorithm

- Bad News Travels Slow

Information Stored at Node	Distance to Reach Node						
	A	B	C	D	E	F	G
A	0	1	1	2	1	1	2
B	1	0	1	2	2	2	3
C	1	1	0	1	2	2	2
D	2	2	1	0	3	2	1
E	4	3	5	3	0	2	3
F	1	2	2	2	2	0	1
G	2	3	2	1	3	1	0

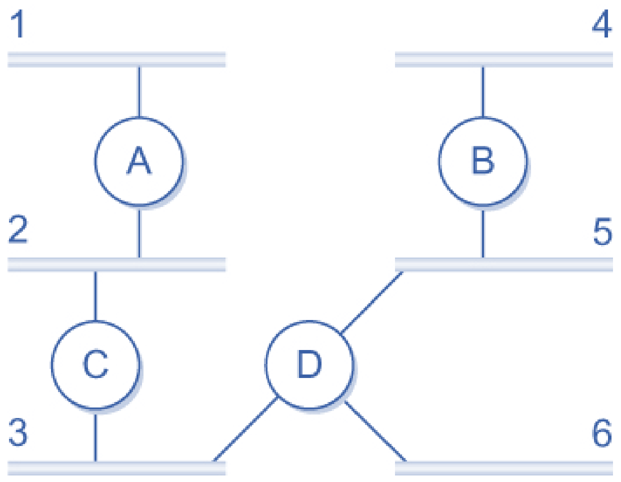


Count-to-infinity Problem

CS 120 LEC 10

Routing Information Protocol (RIP)

- Included in BSD-UNIX distribution in 1982
- Use distance vector algorithm
  - Distance metric: # hops (max = 15 hops), each link has cost 1
  - Distance Vectors exchanged with neighbors every 30 sec in response message
    - Each message: list of up to 25 destination subnets



Routing Table A

SubnetNum	Distance	NextHop
1	0	Interface to 1
2	0	Interface to 2
3	1	C
4	3	C
5	2	C
6	2	C

Routing Protocols

- Routing Information Protocol (RIP)
  - Algorithm: Distance Vector
- Open Shortest Path First (OSPF)
  - Algorithm: Link State
- Border Gateway Protocol (BGP)

Intradomain Routing Protocol

Interdomain Routing Protocol

Link State Routing

- Network topology is known to all routers
  - Accomplished via broadcasting link state packets (LSP) to all routers
- Routing Algorithm: computes shortest paths from one node (source) to all other nodes
  - Based on Dijkstra's algorithm

Dijkstra's Algorithm

Initialization:

M = {s}

for all nodes v

    if v adjacent to s

then $D_s(v) = c(u, v)$

else $D_s(v) = \text{inf}$

Loop

  find w not in M such that $D_s(w)$ is a minimum

    add w to M

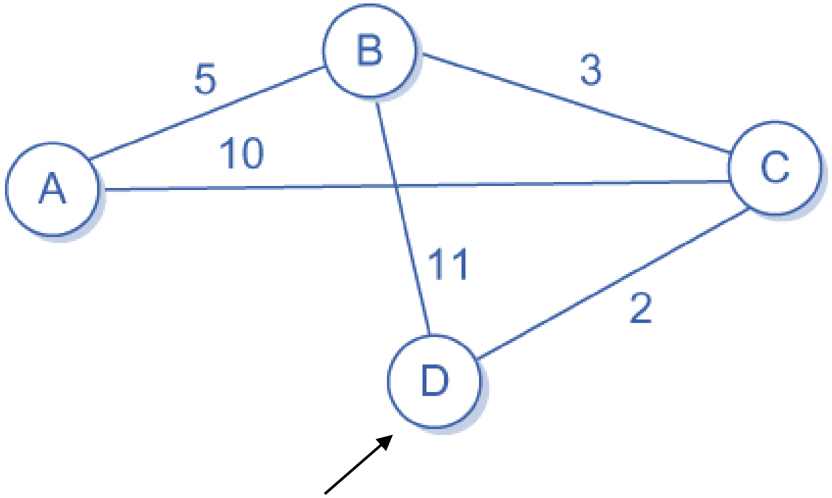
    update $D_s(v)$ for all v adjacent to w and not in M:

      $D_s(v) = \min(D_s(v), D_s(w) + c(w, v))$

until all nodes in M

- M: set of node processed
- S: node of the local router
- v: node of other routers
- $D_s(v)$ distance from s to v
- $c(u, v)$ link weight between node u and v

Dijkstra's Algorithm

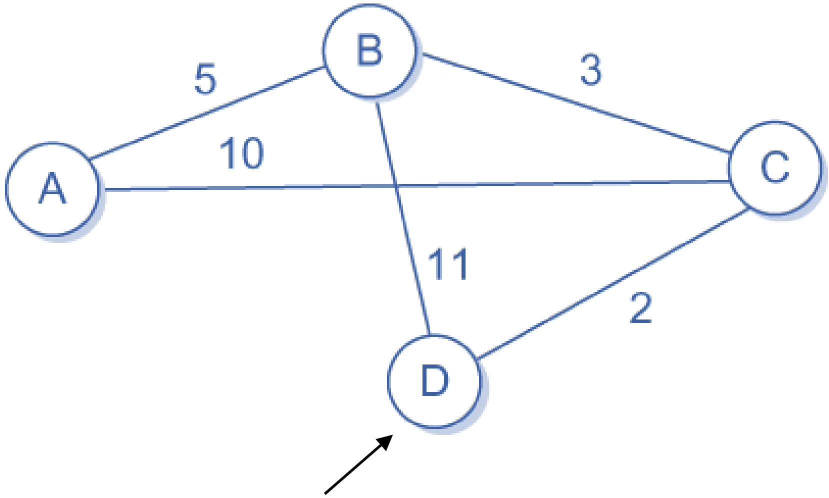


Target Router

M	$D_D(A)$	$D_D(B)$	$D_D(C)$
{D}	Inf, from D	11, from D	2, from D

Dijkstra's Algorithm

M	$D_D(A)$	$D_D(B)$	$D_D(C)$
{D}	Inf, from D	11, from D	2, from D
{D, C}	12, from C	5, from C	2, from D

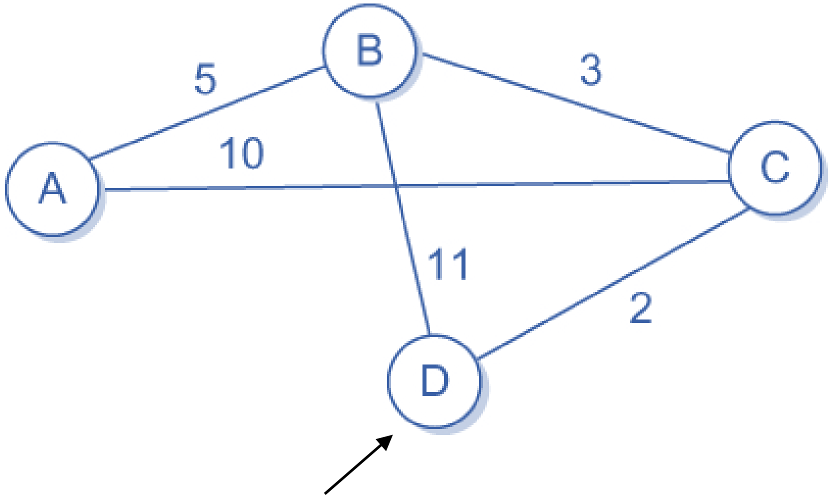


Target Router

CS 120 LEC 10

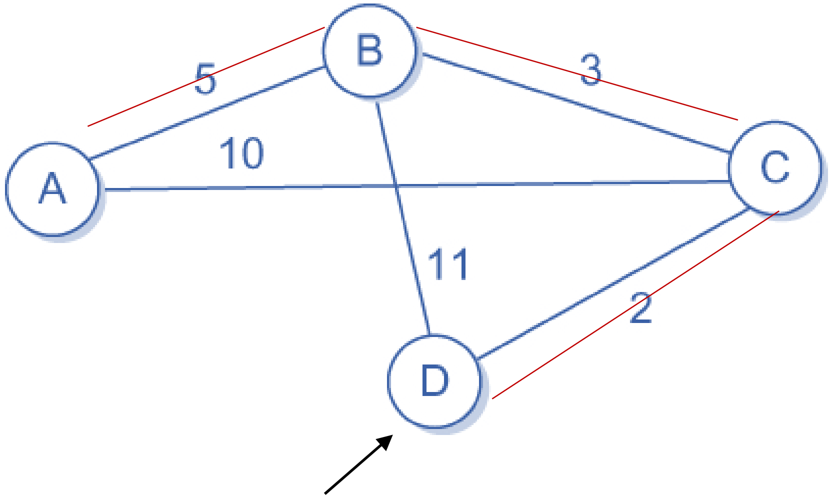
Dijkstra's Algorithm

M	$D_D(A)$	$D_D(B)$	$D_D(C)$
{D}	Inf, from D	11, from D	2, from D
{D, C}	12, from C	5, from C	2, from D
{D, C, B}	10, from B	5, from C	2, from D



Target Router

Dijkstra's Algorithm

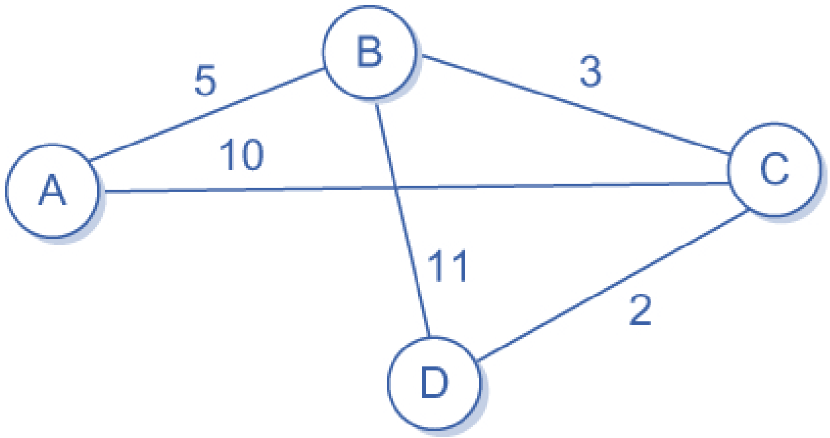


M	$D_D(A)$	$D_D(B)$	$D_D(C)$
{D}	Inf, from D	11, from D	2, from D
{D, C}	12, from C	5, from C	2, from D
{D, C, B}	10, from B	5, from C	2, from D

Dijkstra's Algorithm (Another notation)

- <Destination, Cost, Nexthop>

Step	Confirmed	Tentative
1	(D,0,–)	
2	(D,0,–)	(B,11,B) (C,2,C)
3	(D,0,–) (C,2,C)	(B,11,B)
4	(D,0,–) (C,2,C)	(B,5,C) (A,12,C)
5	(D,0,–) (C,2,C) (B,5,C)	(A,12,C)
6	(D,0,–) (C,2,C) (B,5,C)	(A,10,C)
7	(D,0,–) (C,2,C) (B,5,C) (A,10,C)	



Open Shortest Path First (OSPF)

- "Open": nonproprietary standard created under Engineering Task Force (IETF).
  - Security: all OSPF messages authenticated (to prevent malicious intrusion)
  - Hierarchical routing: OSPF in large domains
  - Load balancing: multiple same-cost paths allowed (only one path in RIP)

OSPF: Link State Announcement

- Link State Announcement (LSA) contains key Information to describe the network topology
  - ID of the router
  - **Neighbors of the router, and the cost to the neighbor (link state of its neighbors)**
    - i.e., neighbor ID and link weight
- LSA is broadcasted to other routers
- Routers use the link states of other routers to construct the topology of entitle net
  - Based on that to calculate the shortest path

OSPF: Link State Announcement

- Flooding method
  - Basic Flooding
    - Transmit LSA to adjacent routers
    - Routers ACK the received LSA
    - Routers store the received (new) LSA
    - Forward LSA to adjacent routers
  - Key Designs: avoid duplication and out-of-date LSA
    - Seq: each LSA contains an increasing sequence number
      - Differentiate duplicated and old LSA.
        - Restart? Send out-of-order LSA to neighbors, neighbors ack the last stored seq
    - Aging: each LSA is associated with an increasing age
      - Flush LSA when reaching the max age (typically 1 hour)
      - Routers periodically broadcast LSA to refresh LSA (typically 0.5 hour)

Forwarding Table vs. Routing Table

- Forwarding table
  - Determines local forwarding
    - Optimized for looking up an address when forwarding a packet
  - Normally in hardware
  - Contains mappings from network numbers to outgoing interfaces and next-hop destination MAC addresses
- Routing table
  - Built by the routing algorithm as a precursor to build the forwarding table
    - Optimized for calculating changes in network topology
  - Normally in software
  - Contains mappings from network numbers to next hop routers (IP)

Software

Hardware

Forwarding Table

Dest MAC	Port
AB.CD.EF.12.34.56	2

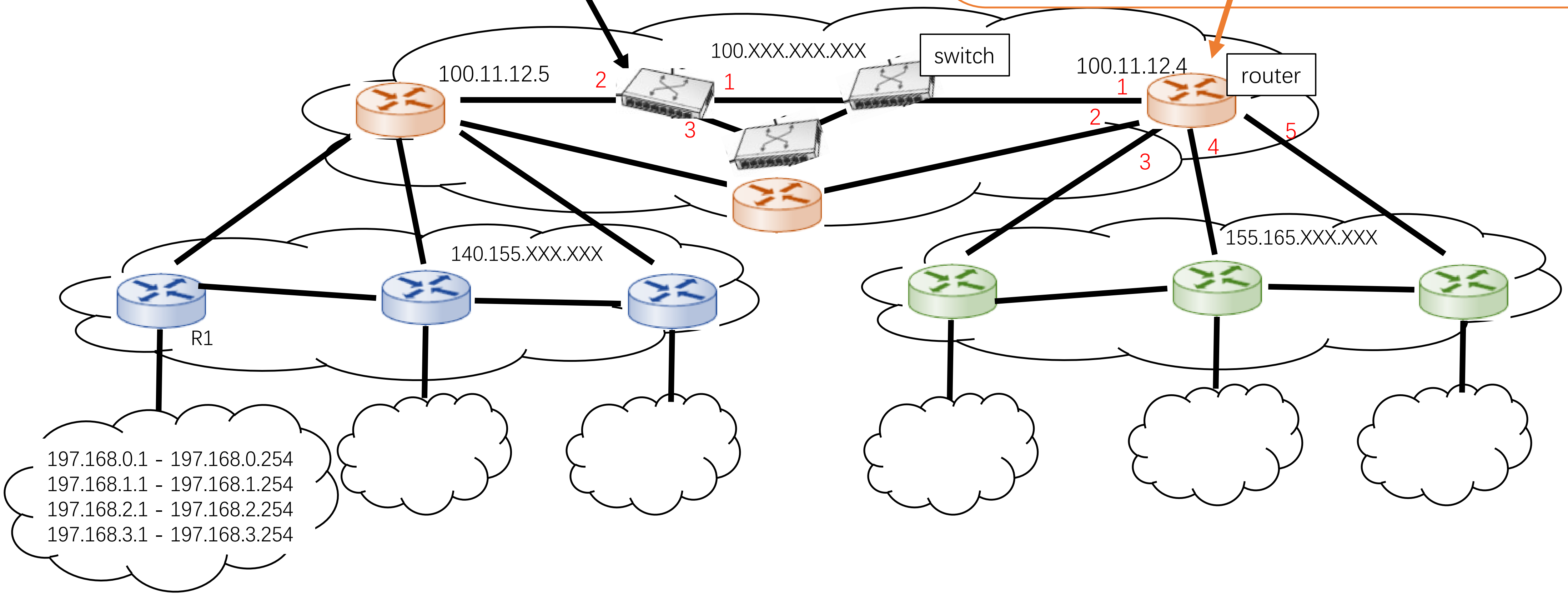
Routing Table

SubnetNum	NextHop
197.168.0.0/22	100.11.12.5

ARP

Forwarding Table

SubnetNum	Interface	Next-Hop MAC
197.168.0.0/22	1	AB.CD.EF.12.34.56



Reference

- Textbook 3.3