

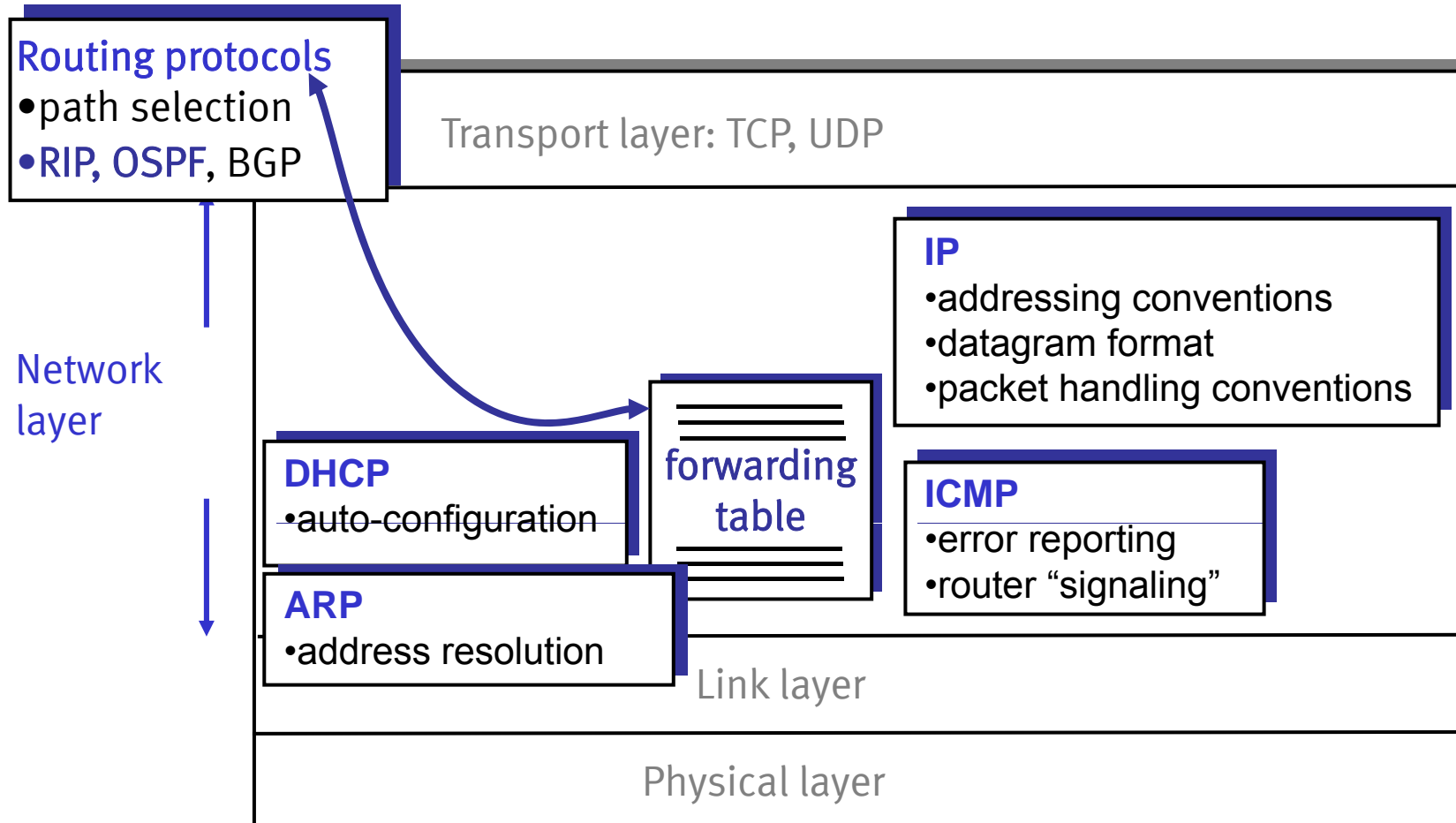


WESTFÄLISCHE  
WILHELMS-UNIVERSITÄT  
MÜNSTER

# Computer Networks, Winter Term 2009/2010

## Intradomain Routing

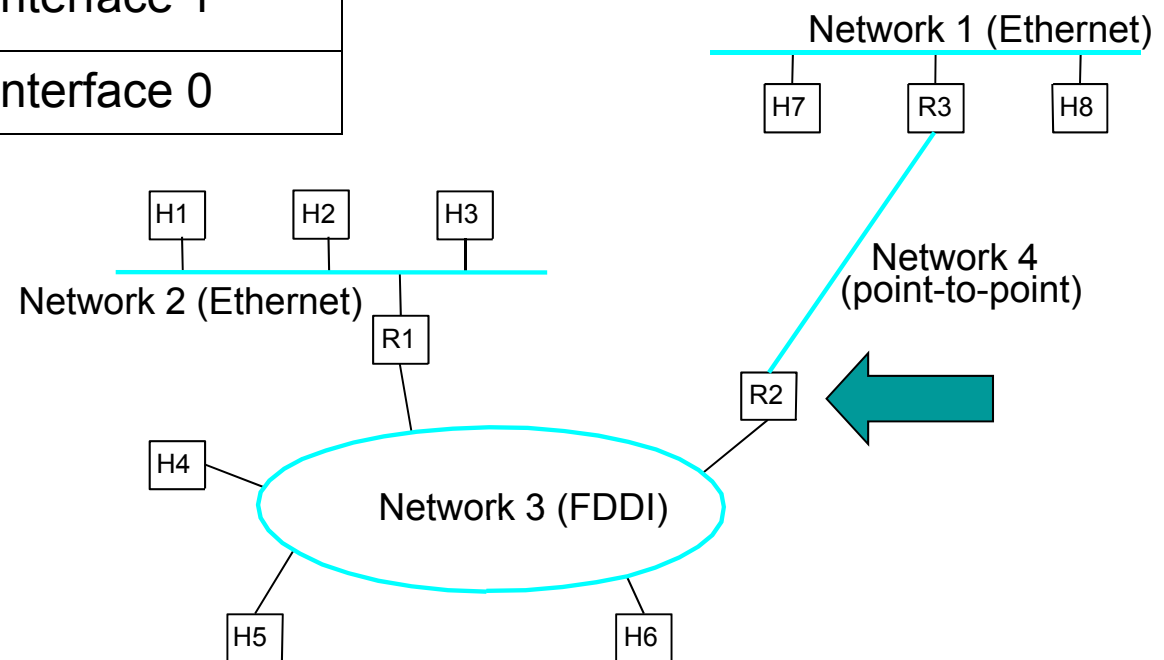
Host, router network layer functions:



Slide based on Kurose, Ross: Computer Networking:  
A Top Down Approach (4<sup>th</sup> ed.). Addison-Wesley, 2007.

# Recall: Sample Forwarding Table

Network Number	Next Hop
1	R3
2	R1
3	Interface 1
4	Interface 0

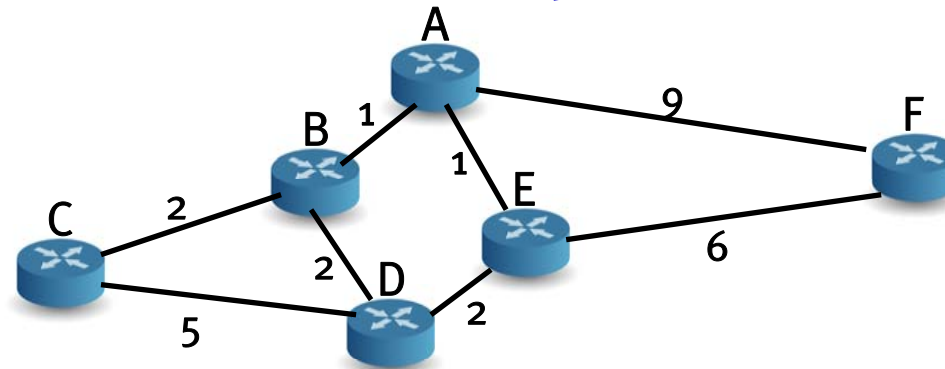


- How to populate forwarding tables?
- How to find lowest cost paths?
- Global vs decentralized knowledge?

- Basics
- Distance vector routing
  - RIP
- Link state routing
  - OSPF
- Link metrics
- Mobile hosts

- Switches and routers need to determine **paths** through networks
- **Forwarding**
  - Selection of output port based on destination address and routing table
- **Routing**
  - Process to build routing table
- Both based on (not necessarily distinct) **tables**
- Solutions need to scale (but not all do)

- Consider network as **weighted graph**, e.g.,



- Challenge

- Find **lowest-cost** path between two nodes (= routers/networks)
- Factors
  - (Semi-)Static: topology
  - Dynamic: load

- Goal

- **Dynamic** and **distributed** algorithms

- Intradomain routing

- Routing within single administrative domain

- Popular Interior Gateway Protocols

- **RIP**: Route Information Protocol
  - Decentralized, distance-vector algorithm
  - Based on hop-count
- **OSPF**: Open Shortest Path First
  - Global, link-state algorithm
  - Based on network map

- Interdomain routing

- Routing at global scale (Internet)
- Exterior Gateway Protocols, e.g., **BGP**

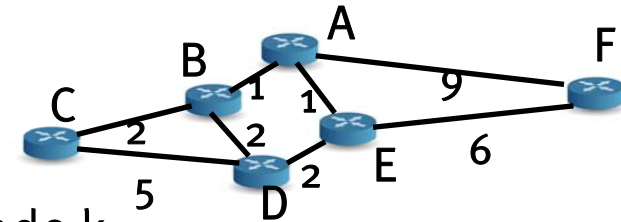
} Subsequent  
lecture

- Know a smarter router
  - Hosts know local router
    - DHCP
  - Local routers know site routers
    - Subnets, OSPF
  - Site routers know core router
    - CIDR
  - Core routers know “everything”
    - CIDR, AS, BGP

- Basics
- **Distance vector routing**
  - RIP
- Link state routing
  - OSPF
- Link metrics
- Mobile hosts

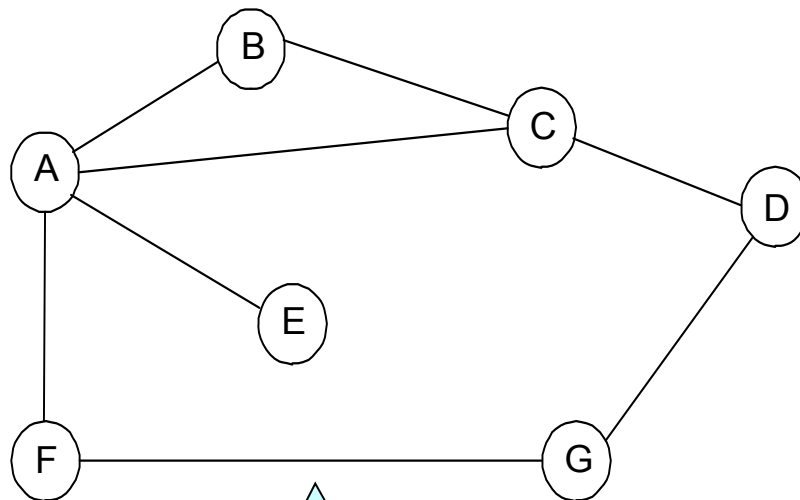
## Distance Vector Routing (1/2)

11



- Distance vector for a node
  - k-th entry indicates cost/distance to reach node k
  - E.g., for node C: (A/3, B/2, C/0, D/4, E/4, F/10)
- Routing idea
  - Distance vectors **initialized** with costs of direct links
    - E.g., for node C: (A/-, B/2, C/0, D/4, E/-, F/-)
  - Nodes **exchange** distance vectors
  - Nodes choose next-hops with **lowest costs**
- Bellman-Ford equation
  - $d_x(y) = \min_z \{ c(x, z) + d_z(y) \}$ 
    - $d_x(y)$  denotes minimum cost to reach y from x
    - $c(x, z)$  denotes cost of link between x and z

- Each node maintains set of triples
  - (Destination, Cost, NextHop)
- Exchange updates with **directly connected** neighbors
  - Updates are distance vectors
  - Periodically (e.g., every 30s)
  - Whenever table changes (“triggered” update)
- Update local table if receive a “**better**” route
  - Smaller cost
  - Came from next-hop
- Refresh existing routes; delete if they time out



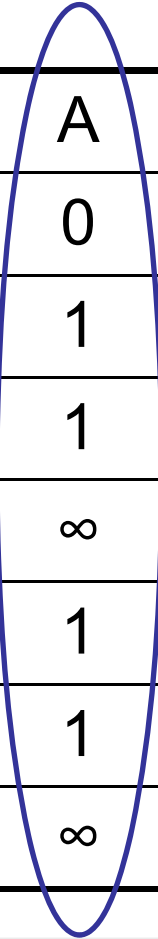
Cost of each link is 1

Destination	Cost	NextHop
A	1	A
C	1	C
D	2	C
E	2	A
F	2	A
G	3	A

Routing table at B

## Example (cont'd): Initial Information at each Node

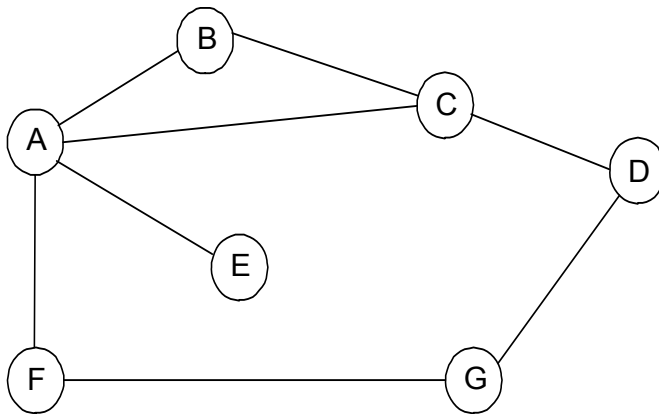
14



	A	B	C	D	E	F	G
A	0	1	1	$\infty$	1	1	$\infty$
B	1	0	1	$\infty$	$\infty$	$\infty$	$\infty$
C	1	1	0	1	$\infty$	$\infty$	$\infty$
D	$\infty$	$\infty$	1	0	$\infty$	$\infty$	1
E	1	$\infty$	$\infty$	$\infty$	0	$\infty$	$\infty$
F	1	$\infty$	$\infty$	$\infty$	$\infty$	0	1
G	$\infty$	$\infty$	$\infty$	1	$\infty$	1	0

## Example (cont'd): Initial Table at A

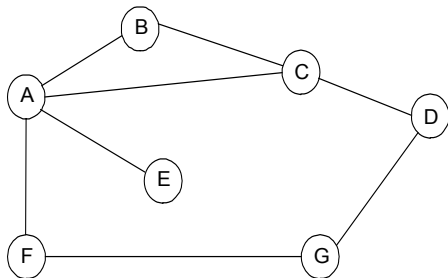
15



Dest	Cost	NextHop
B	1	B
C	1	C
D	$\infty$	-
E	1	E
F	1	F
G	$\infty$	-

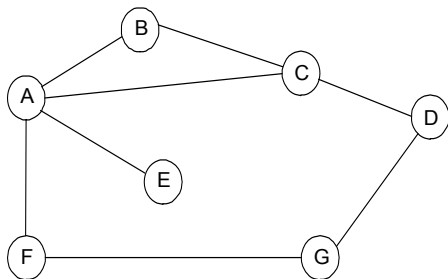
- F tells A it can reach G at cost 1

(Recall: updates occur periodically or are triggered)



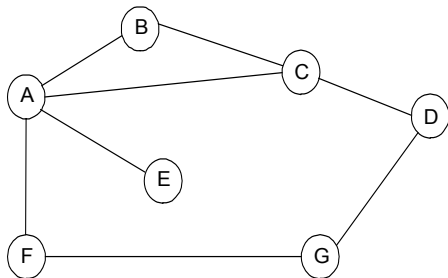
Dest	Cost	NextHop
B	1	B
C	1	C
D	$\infty$	-
E	1	E
F	1	F
G	$\infty$	-

- F tells A it can reach G at cost 1



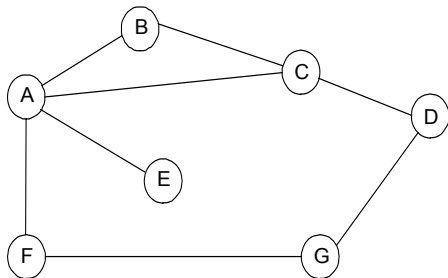
Dest	Cost	NextHop
B	1	B
C	1	C
D	$\infty$	-
E	1	E
F	1	F
G	2	F

- F tells A it can reach G at cost 1
- C tells A it can reach D at cost 1



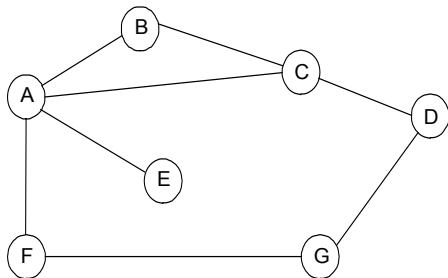
Dest	Cost	NextHop
B	1	B
C	1	C
D	$\infty$	-
E	1	E
F	1	F
G	2	F

- F tells A it can reach G at cost 1
- C tells A it can reach D at cost 1

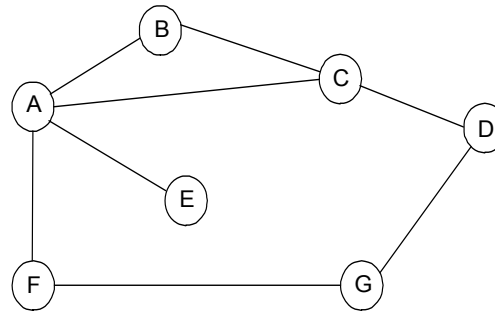


Dest	Cost	NextHop
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	2	F

- F tells A it can reach G at cost 1
- C tells A it can reach D at cost 1
- C tells A it can reach B at cost 1 → no change



Dest	Cost	NextHop
B	1	B
C	1	C
D	2	C
E	1	E
F	1	F
G	2	F

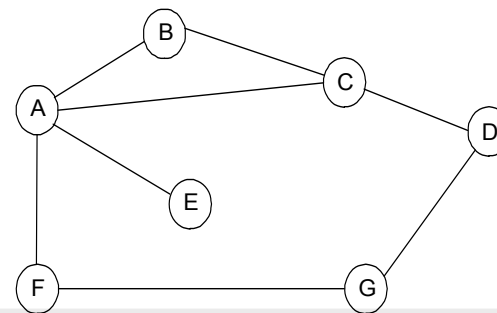


## Final Information

	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

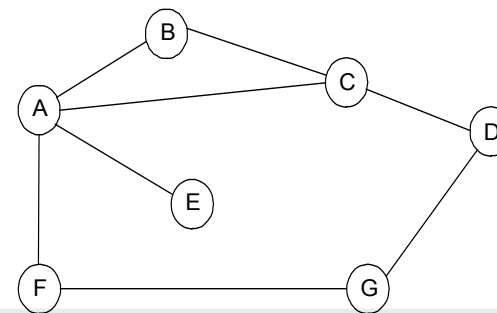
## • Example 1

- F detects that link to G has failed
- F sets distance to G to  $\infty$  and sends update to A
- A sets distance to G to  $\infty$  since it uses F to reach G
- A receives periodic update from C with 2-hop path to G
- A sets distance to G to 3 and sends update to F
- F decides it can reach G in 4 hops via A



## • Example 2

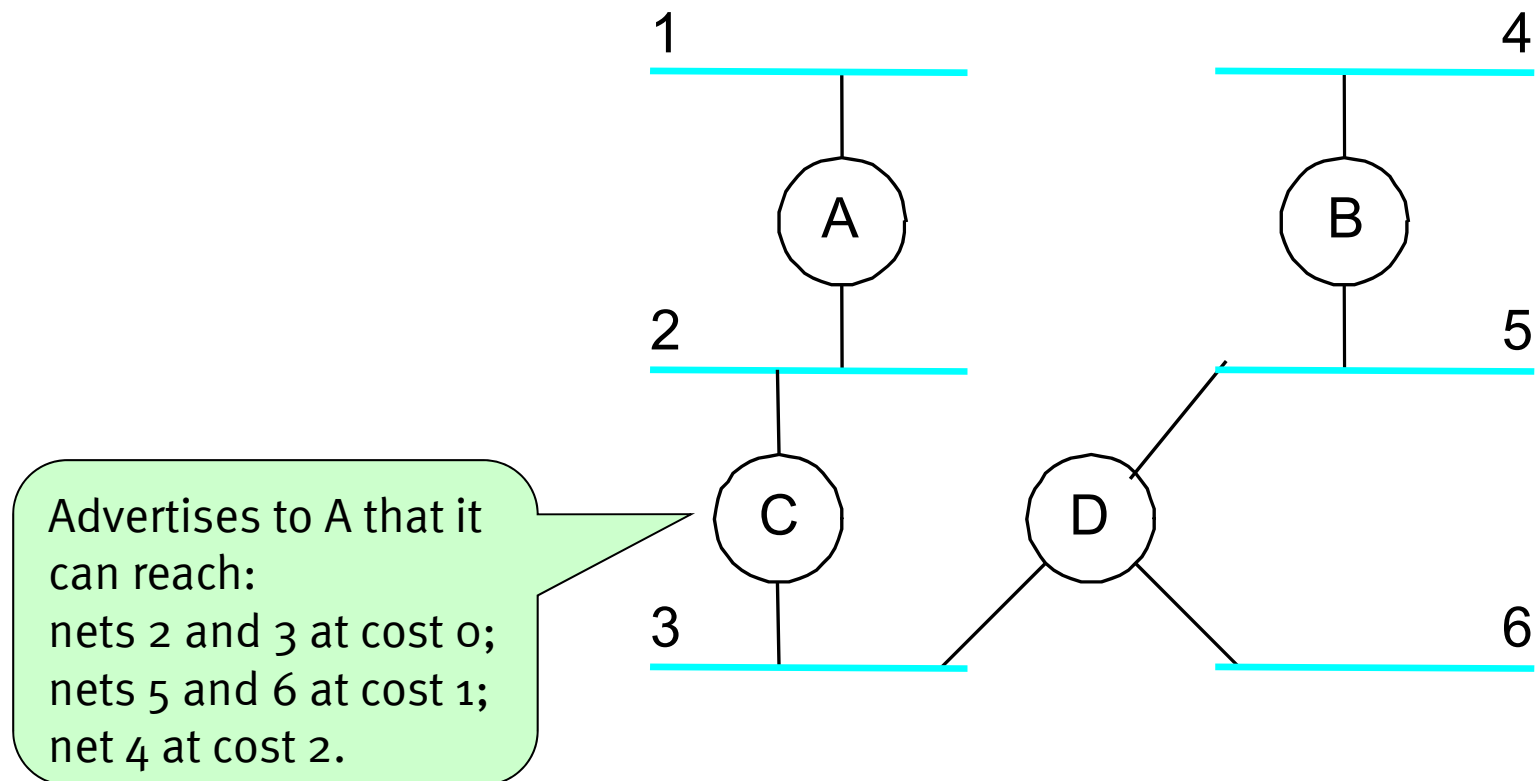
- Link from A to E fails
- A advertises distance of  $\infty$  to E
- B and C advertise a distance of 2 to E
- B decides it can reach E in 3 hops; advertises this to A
- A decides it can reach E in 4 hops; advertises this to C
- C decides that it can reach E in 5 hops...



- Set “infinity” to constant, e.g., 16
  - Maximum number of hops to get across network

- Basics
- Distance vector routing
  - RIP
- Link state routing
  - OSPF
- Link metrics
- Mobile hosts

- Standard based on distance vectors
- Routers advertise cost of reaching networks



- Basics
- Distance vector routing
  - RIP
- **Link state routing**
  - OSPF
- Link metrics
- Mobile hosts

## • Strategy

- Nodes know state and cost of “their” links
  - **Directly** connected links
  - **Link state**
- **Flood** (see next slides) link state to **entire** network
  - Not just neighbors
  - Not entire routing table
- Collection of link states is **complete network map**
  - Every router computes **shortest paths** in this map

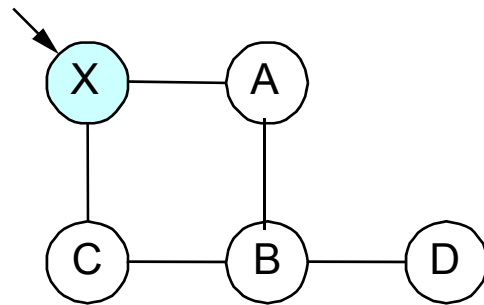
## • Link State Packet (LSP)

- ID of creating node
- Cost of link to each directly connected neighbor
- Sequence number (SEQNO)
- Time-to-live (TTL) for this packet

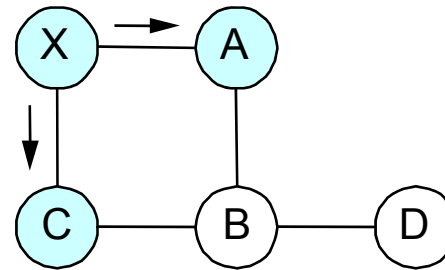
- Store most recent LSP from each node
- Forward LSP to all nodes but sender
- Generate new LSP periodically
  - Increment SEQNO
- Start SEQNO at 0 when reboot
- Decrement TTL of each stored LSP
  - Discard when TTL=0
- Notice
  - Duplicate messages
  - Limited scalability
  - Alternative: Trickle
    - Levis et al.: The Emergence of a Networking Primitive in Wireless Sensor Networks, CACM 51(7), 2008



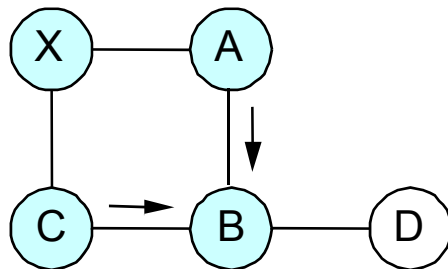
# Sample Flooding



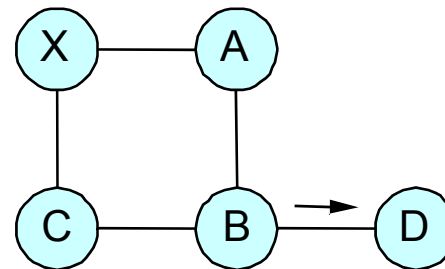
(a)



(b)



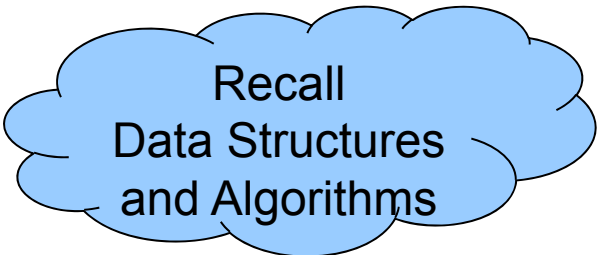
(c)



(d)

- Dijkstra's algorithm

- Aka **shortest path first, forward search**
- Computes shortest paths from source node  $s$ 
  - Shortest path spanning tree
- Terminology
  - $N$  denotes set of nodes in the graph
  - $s$  denotes router computing shortest paths
  - $M$  denotes the set of nodes incorporated so far
  - $c(i, j)$  denotes non-negative cost (weight) for edge  $(i, j)$
  - $C(n)$  denotes cost of the path from  $s$  to node  $n$



Recall  
Data Structures  
and Algorithms

```
 $M = \{s\}$ 
```

```
for each  $n$  in  $N - \{s\}$ 
```

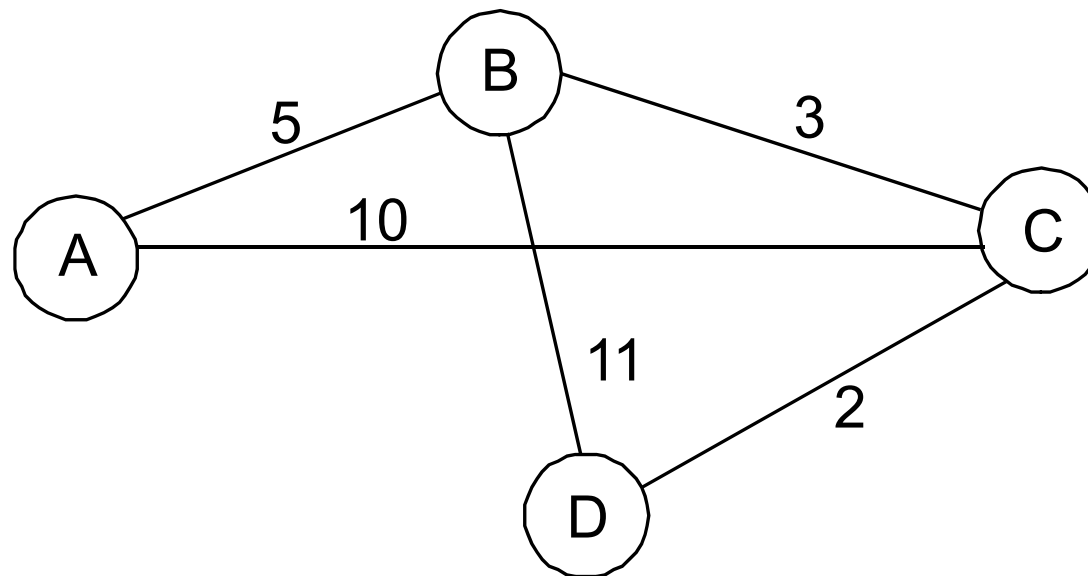
```
     $C(n) = c(s, n)$ 
```

```
while ( $N \neq M$ )
```

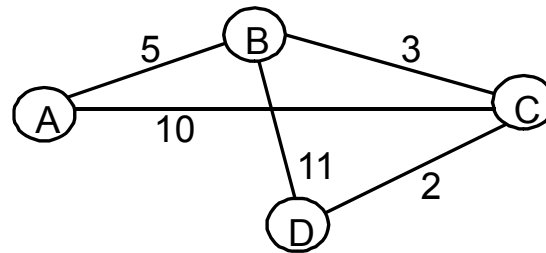
```
     $M = M$  union  $\{t\}$  such that  $C(t)$  is the minimum for all  $t$  in  $(N - M)$ 
```

```
    for each  $n$  in  $(N - M)$ 
```

```
         $C(n) = \text{MIN}(C(n), C(t) + c(t, n))$ 
```

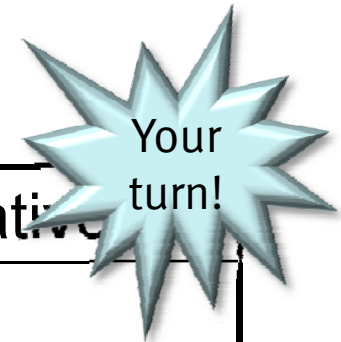


- Each router maintains lists
  - Tentative and
  - Confirmedeach with entries of the form (Destination, Cost, NextHop)
- Start with own ID put in Confirmed
- Pick elements just Confirmed to determine new Tentative elements
- Gradually move element with minimal cost from Tentative to Confirmed
- Stop when Confirmed covers entire graph



## Example: Node D

34



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)	

- Basics
- Distance vector routing
  - RIP
- Link state routing
  - OSPF
- Link metrics
- Mobile hosts

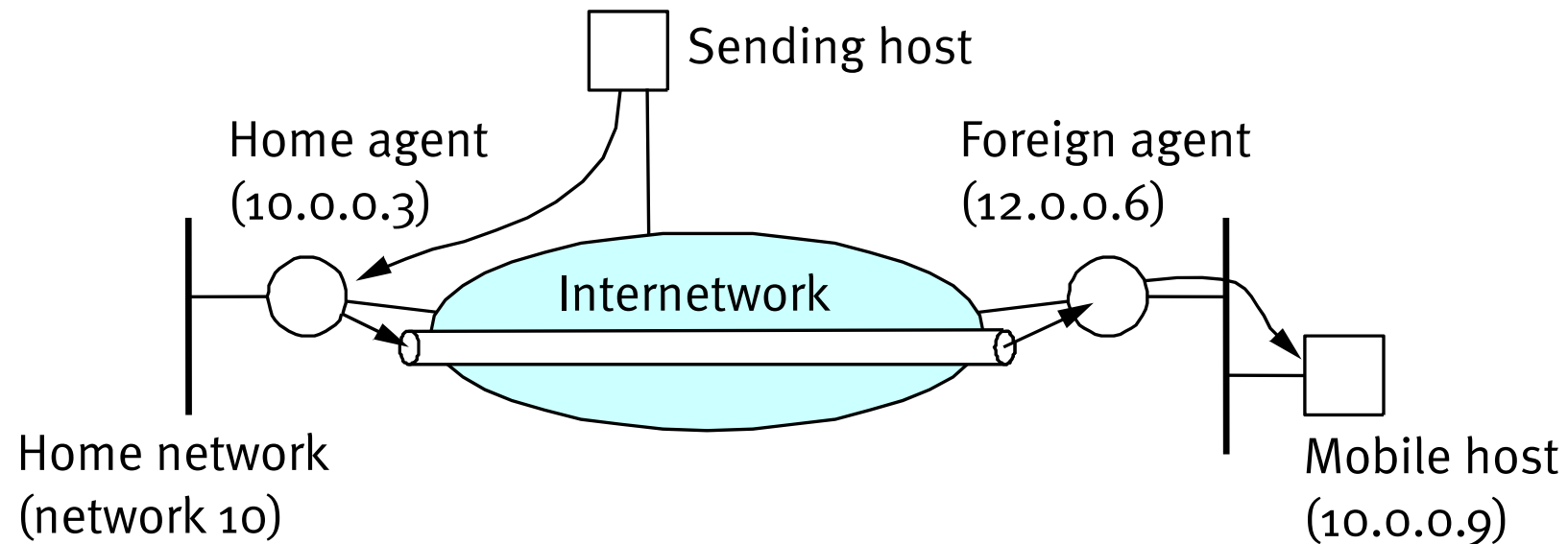
- Widely used link-state routing protocol
  - Forward Search Algorithm
- Additions to basic algorithm
  - Authentication of routing messages
  - Additional hierarchy: [Areas](#)
    - Routers within area exchange LSPs
    - Special area: Backbone area with area border routers ([ABRs](#))
      - Packets travel via backbone
      - ABRs summarize link states of connected areas
  - Load balancing
    - Traffic distributed among multiple routes with same cost

- Basics
- Distance vector routing
  - RIP
- Link state routing
  - OSPF
- **Link metrics**
- Mobile hosts

- Determine cost of link
- Factors
  - Bandwidth
  - Length/latency
  - Load
  - Monetary cost
- Common practice
  - Static:  $c/\text{bandwidth}$
  - Cost changes under control of administrator
    - Avoid instabilities

- Original ARPANET metric
  - Measures number of packets enqueued on each link
  - Took neither latency nor bandwidth into consideration
- New ARPANET metric
  - Stamp each incoming packet with its arrival time (AT)
  - Record departure time (DT)
    - If timeout, reset DT to departure time for retransmission
  - When link-level ACK arrives, compute
    - $\text{Delay} = (\text{DT} - \text{AT}) + \text{TransmissionTime} + \text{Latency}$ 
      - TransmissionTime and Latency are static costs based on link's bandwidth and latency
  - Link cost = average delay over some time period
- Revised ARPANET metric: Fine Tuning
  - Compressed dynamic range
  - Replaced Delay with link utilization

- Basics
- Distance vector routing
  - RIP
- Link state routing
  - OSPF
- Link metrics
- **Mobile hosts**



0. Mobile host (MH, 10.0.0.9) has home agent HA
1. MH contacts foreign agent (FA)
  - Informs about HA
2. FA contacts HA
  - Informs about c/o-address, e.g., 12.0.0.6
3. Sender uses 10.0.0.9
  - HA **intercepts** (proxy ARP)
    - Optimization: Send binding update to sender with c/o address
  - HA **tunnels** to FA
  - FA delivers to MH
- Notice: FA+MH on single host possible

- Distance vectors and link states to find least cost paths
  - Distance vectors for **decentralized** routing
    - **Count to infinity**
  - Link state for **global** routing
    - Build network map based on link states
    - OSPF typical for intradomain routing
- Link costs mostly static
- Tunneling for mobile hosts

- Apply distance vector and link state routing on sample graphs to build forwarding tables
- Discuss strengths and weaknesses of distance vector and link state routing
- Explain tunneling (for IPsec, IPv6-in-IPv4, and Mobile IP)