Routing Algorithms

EE450: Introduction to Computer Networks

Professor A. Zahid

IP Packet Delivery

- Two Processes are required to accomplish IP packet delivery, namely the Routing Process and the Forwarding Process
  - Routing is the process of discovering and selecting the path to the destination according to some metrics.
  - Forwarding is the process of inserting the IP packet into a Layer-2 frame and forwarding the frame to the next hop (which could be the destination host or another intermediate router).

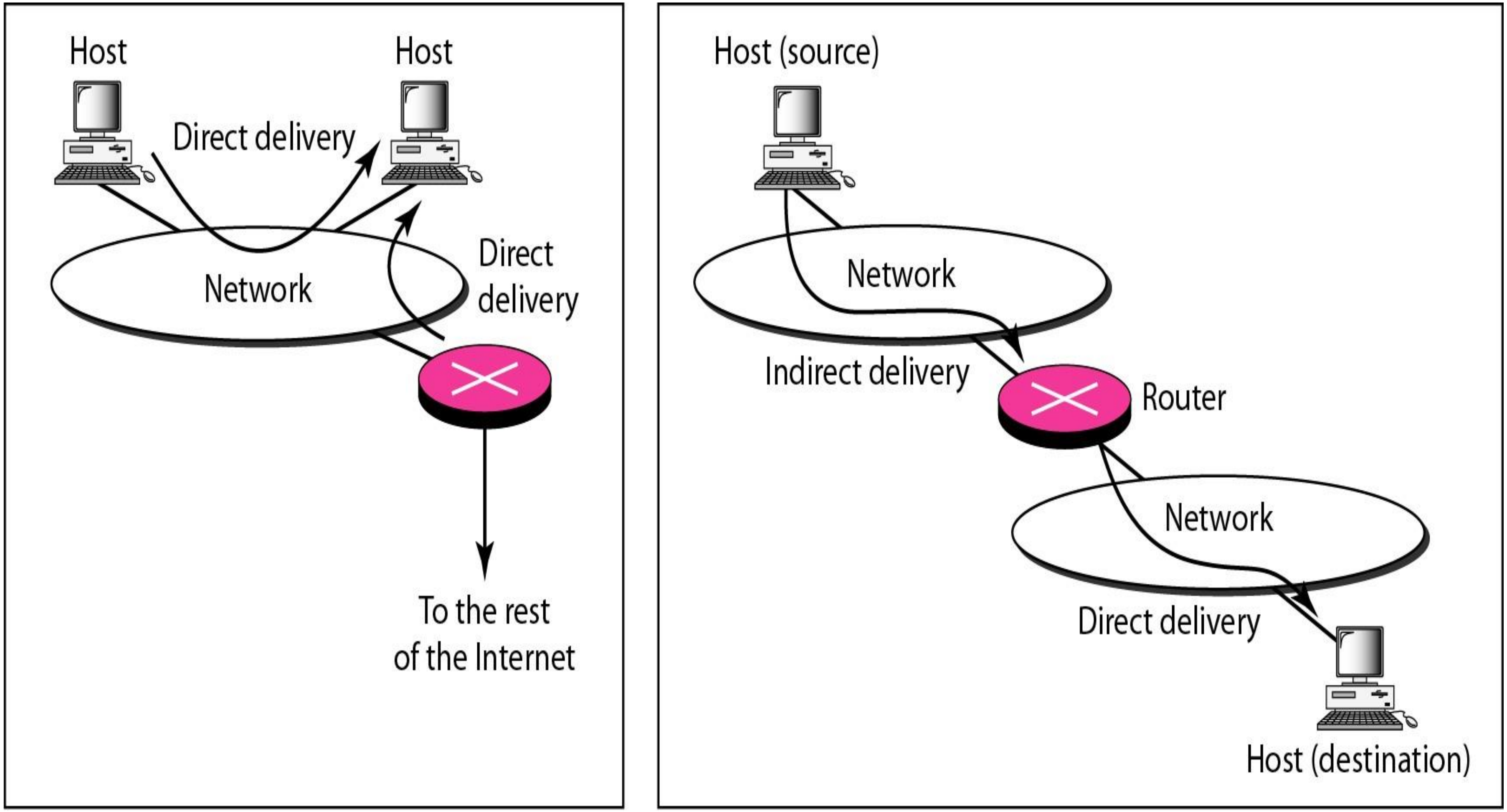
Routing Tables

- Routing Tables are built up by the routing algorithms. They generally consist of:
  - Destination Network Address: The network portion of the IP address for the destination network
  - Subnet Mask: used to distinguish the network address from the host address
  - The IP address of the next hop to which the interface forwards the IP packet for delivery
  - The Interface with which the route is associated

Forwarding Tables

- After the routing lookup is completed and the next hop is determined, The IP packet is forwarded according to a local or remote delivery models
  - Local delivery model is when the destination and the host are on the same local network. In this case, the IP packet is inserted into a MAC-frame which is forwarded directly to the destination
  - Remote delivery model is when the destination and the host are on different networks. In this case, the IP packet is inserted into a Layer-2 frame which is forwarded to the next hop router

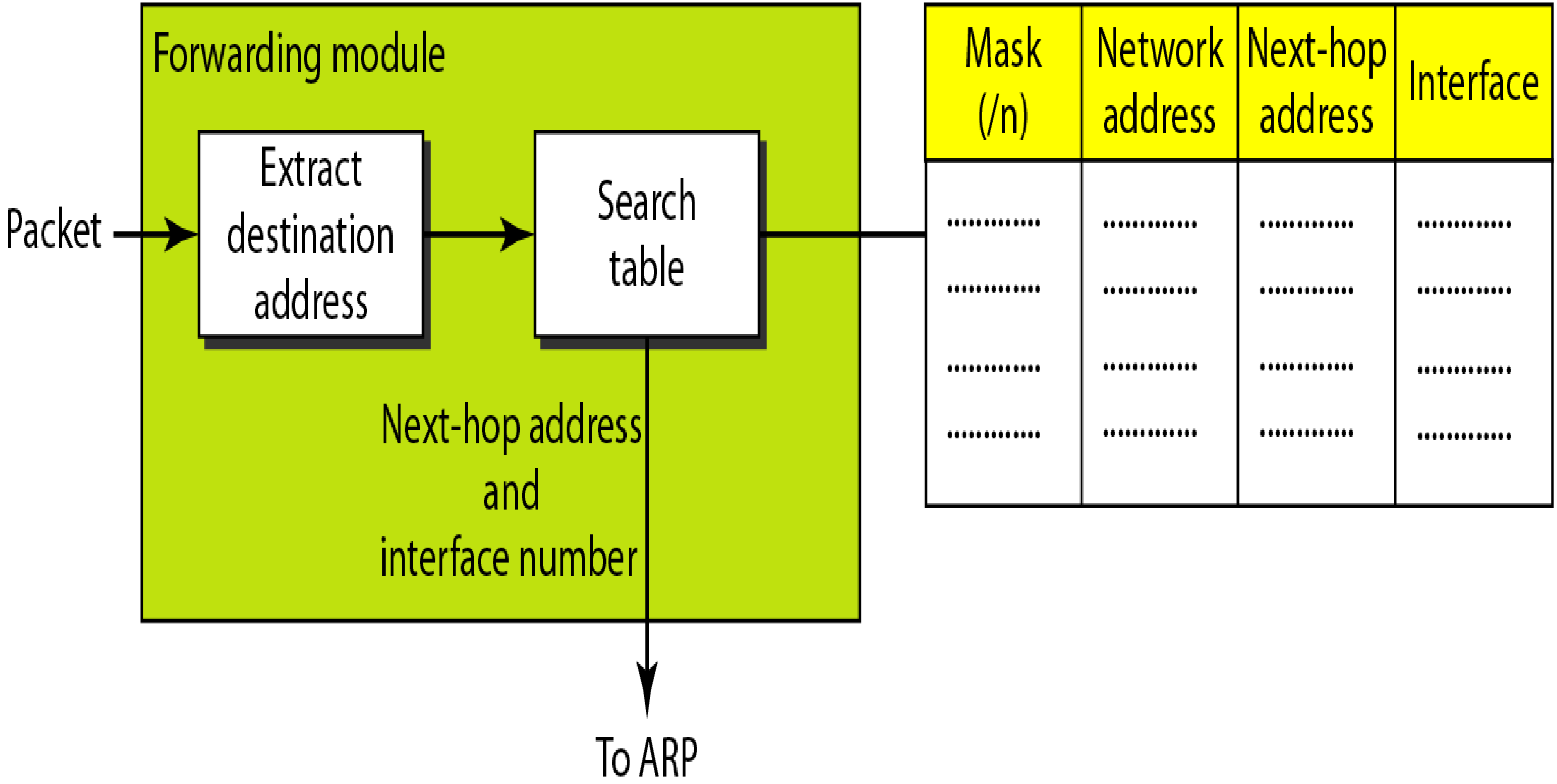
Local (Direct) vs. Remote Delivery



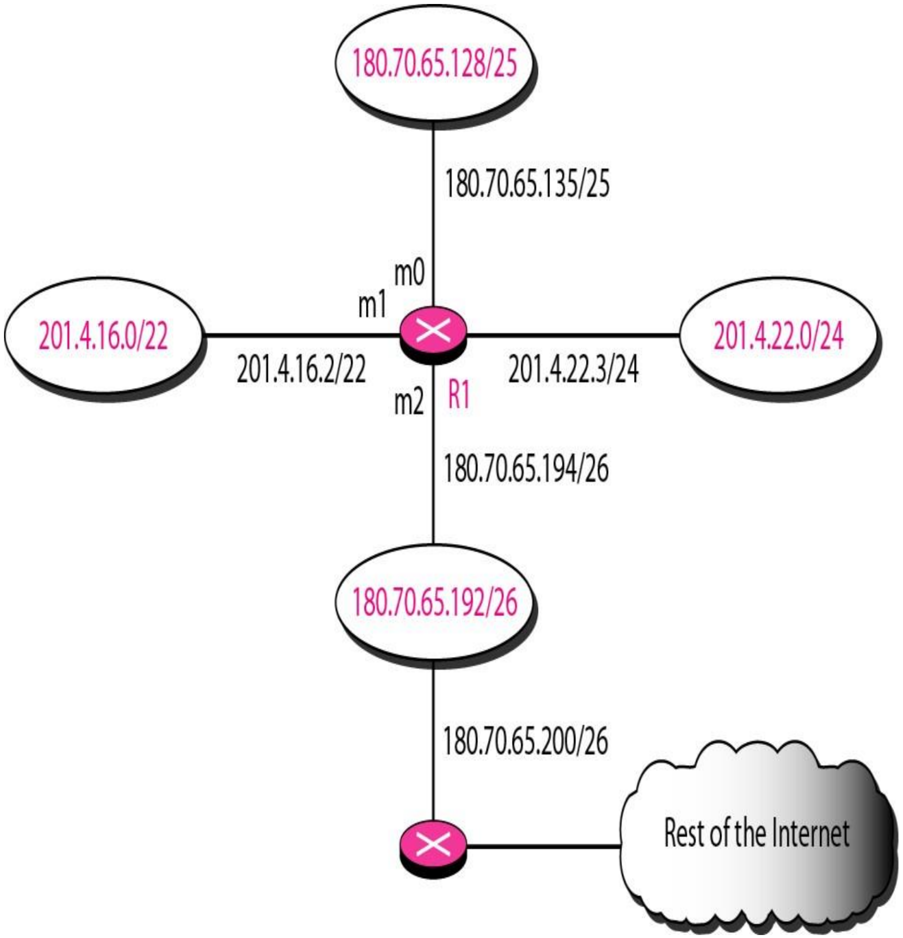
a. Direct delivery

b. Indirect and direct delivery

Forwarding Module



Example: Forwarding Table

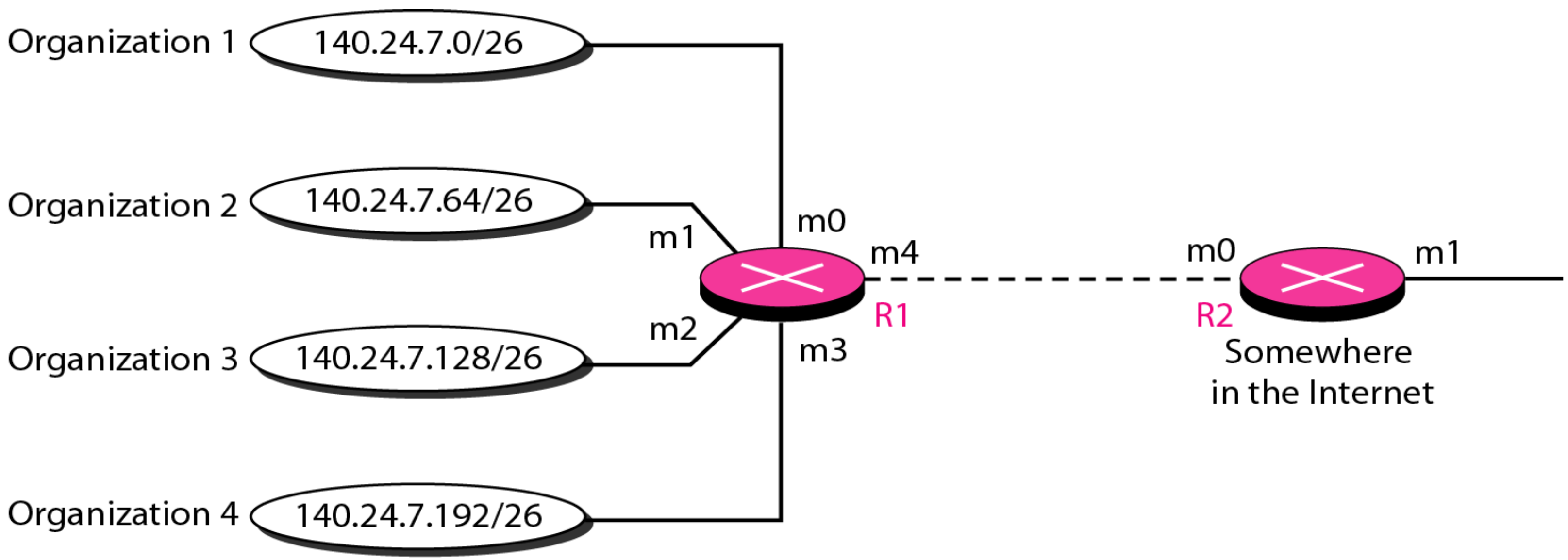


Mask	Network Address	Next Hop	Interface
/26	180.70.65.192	—	m2
/25	180.70.65.128	—	m0
/24	201.4.22.0	—	m3
/22	201.4.16.0	....	m1
Any	Any	180.70.65.200	m2

Example (Continued)

- Suppose that R1 receives a Packet destined to 180.70.65.140. The router performs the following steps:
  - The first mask (/26) is applied to the destination address. Result is 180.70.65.128. No match
  - The second mask (/25) is applied to the destination address. Result is 180.70.65.128. A match. The next hop address (in this case it is the destination host address) and the interface m_0 is then passed to the ARP module to get the MAC address

CIDR: Address Aggregation



Mask	Network address	Next-hop address	Interface
/26	140.24.7.0	----------	m0
/26	140.24.7.64	----------	m1
/26	140.24.7.128	----------	m2
/26	140.24.7.192	----------	m3
/0	0.0.0.0	Default	m4

Routing table for R1

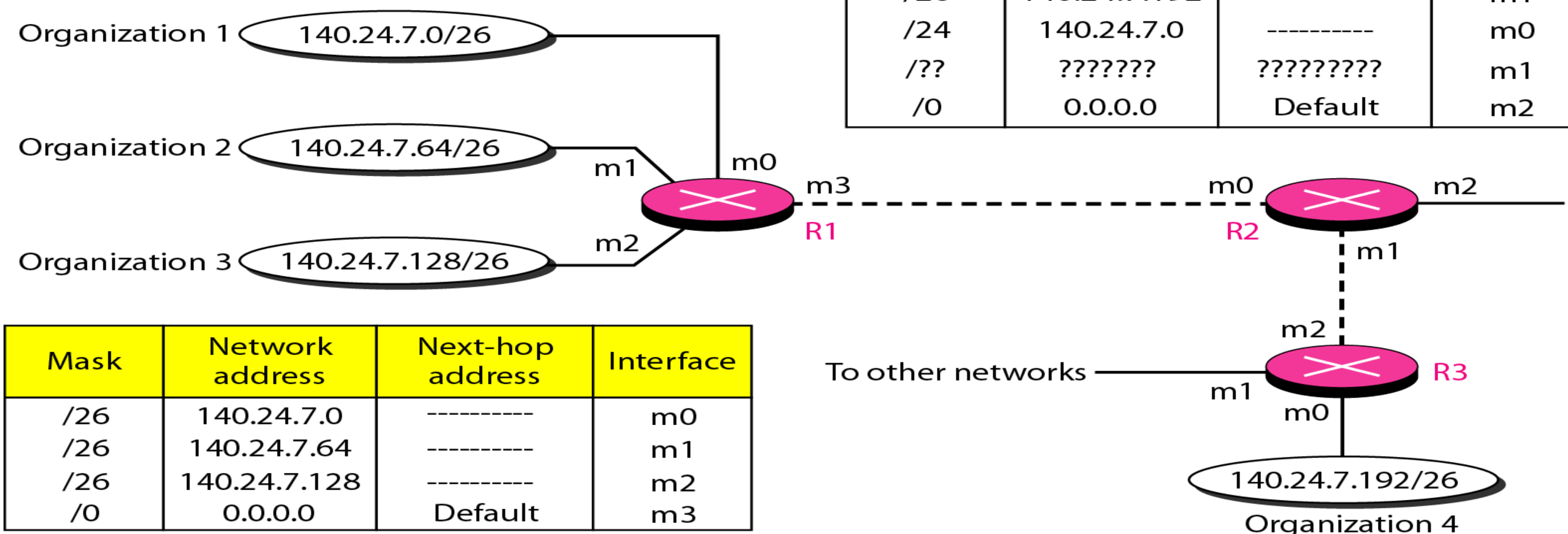
Mask	Network address	Next-hop address	Interface
/24	140.24.7.0	----------	m0
/0	0.0.0.0	Default	m1

Routing table for R2

Longest Mask (Prefix) Matching

Routing table for R2

Mask	Network address	Next-hop address	Interface
/26	140.24.7.192	----------	m1
/24	140.24.7.0	----------	m0
/??	???????	?????????	m1
/0	0.0.0.0	Default	m2



Mask	Network address	Next-hop address	Interface
/26	140.24.7.0	----------	m0
/26	140.24.7.64	----------	m1
/26	140.24.7.128	----------	m2
/0	0.0.0.0	Default	m3

Routing table for R1

Mask	Network address	Next-hop address	Interface
/26	140.24.7.192	----------	m0
/??	???????	?????????	m1
/0	0.0.0.0	Default	m2

Routing table for R3

Example: Longest Prefix Matching

when looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address.

Destination Address Range	Link interface
11001000 00010111 00010*** *********	0
11001000 00010111 00011000 *********	1
11001000 00010111 00011*** *********	2
otherwise	3

examples:

DA: 11001000 00010111 00010110 10100001 which interface?

DA: 11001000 00010111 00011000 10101010 which interface?

Static vs. Dynamic Routing

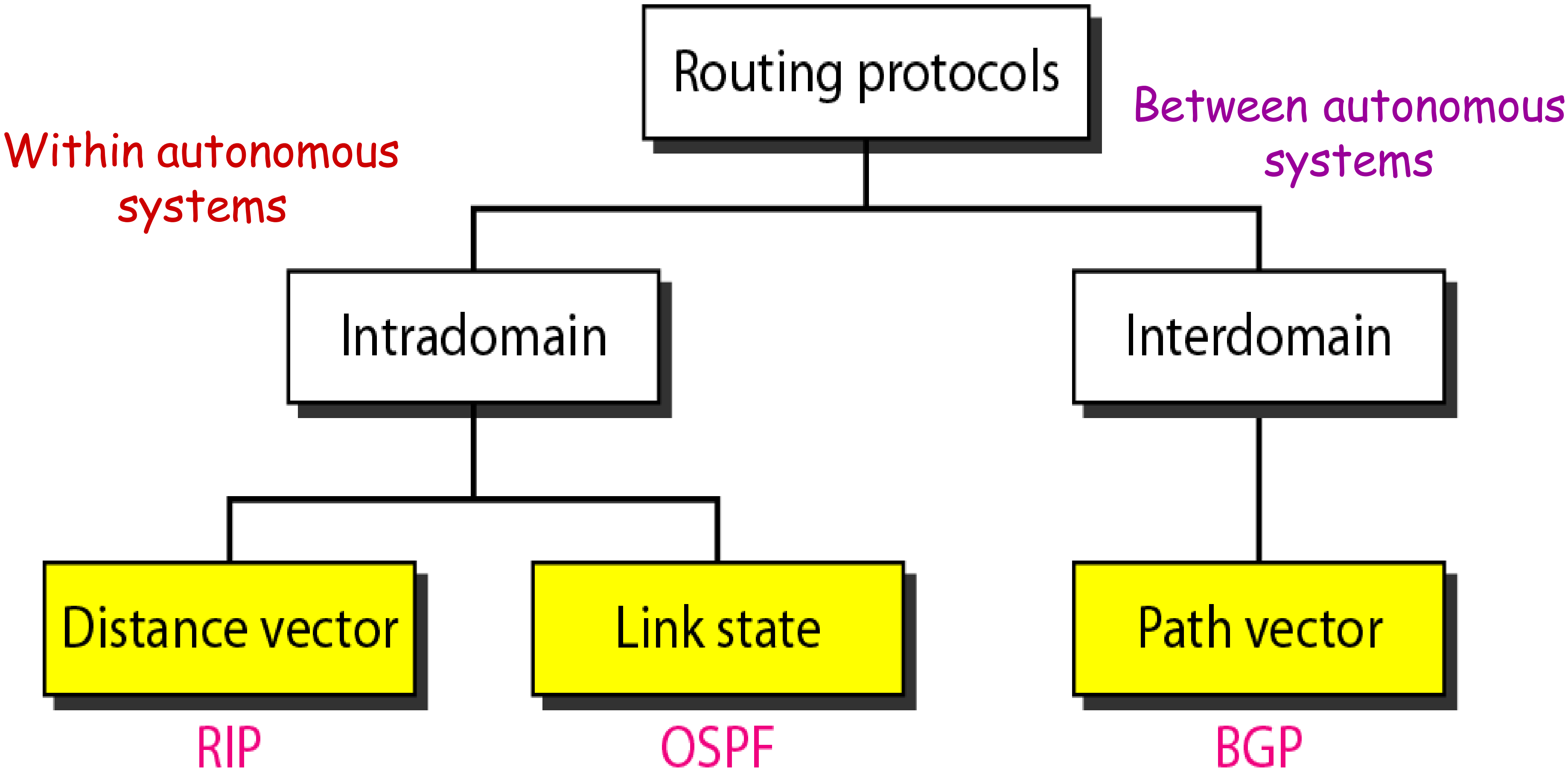
- Static Routing Tables are entered manually
- Strengths of Static Routing
  - Ease of use
  - Reliability
  - Control
  - Security through obscurity
  - Efficiency
- Weaknesses of Static Routing
  - Not Scalable
  - Not adaptable to link failures

- Dynamic Routing Tables are created through the exchange of information between routers on the availability and status of the networks to which an individual router is connected to. Two Types
  - Distance Vector Protocols
    - RIP: Routing Information Protocol
  - Link State Protocols
    - OSPF: Open Shortest Path First

Routing Metrics

- Routing metrics are used by dynamic routing protocols to establish preference for a particular route.
- Goal of routing metrics is to provide the capability to the routing protocol to support Route Diversity and Load Balancing
- Most Common routing metrics include:
  - Hop count (minimum # of hops)
  - Shortest distance
  - Bandwidth/Throughput (maximum throughput)
  - Load (actual usage)
  - Delay (shortest delay)
  - Reliability
  - Cost

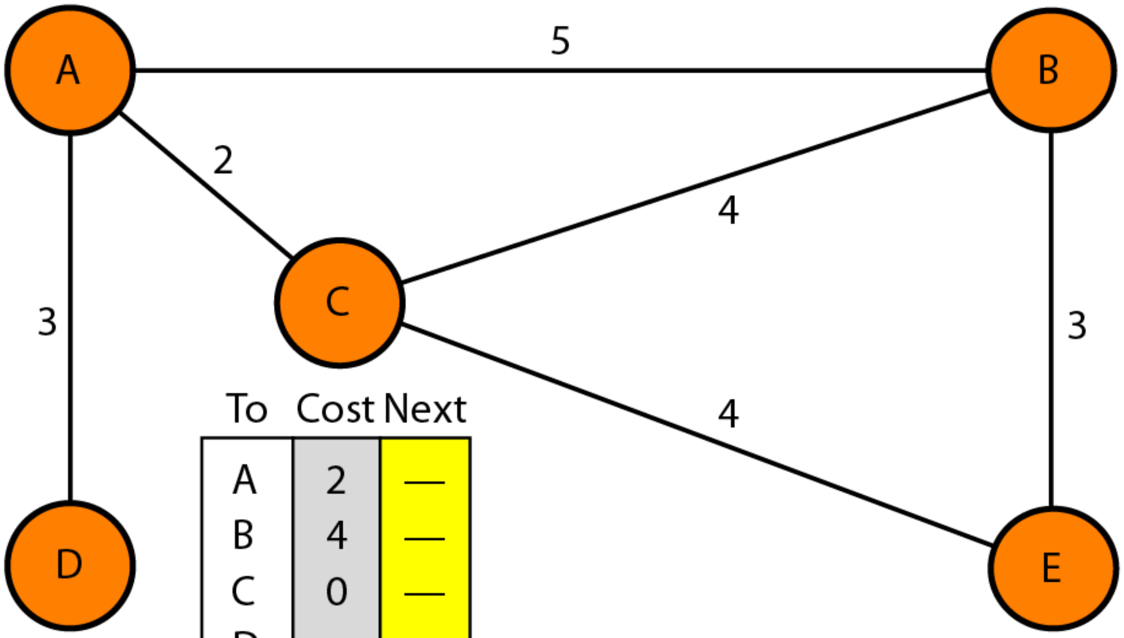
Popular Routing Protocols



Distance Vector (DV) Routing

- Example: RIP: Routing Information Protocol
  - Based on an algorithm by Bellman-Ford (Dynamic Programming)
  - Each router on the network compiles a list of the networks it can reach (in the form of a distance vector) and exchange this list with its neighboring routers only
  - Upon receiving vectors from each of its neighbors, the router computes its own distance to each neighbor. Then, for every network X, router finds that neighbor who is closer to X than to any other neighbor. Router updates its cost to X. After doing this for all X, router goes to the first step.

Initial Distance Vector Tables



A's table

To	Cost	Next
A	0	—
B	5	—
C	2	—
D	3	—
E	∞	

B's table

To	Cost	Next
A	5	—
B	0	—
C	4	—
D	∞	
E	3	—

C's table

To	Cost	Next
A	2	—
B	4	—
C	0	—
D	∞	
E	4	—

D's table

To	Cost	Next
A	3	—
B	∞	
C	∞	
D	0	—
E	∞	

E's table

To	Cost	Next
A	∞	
B	3	B
C	4	C
D	∞	
E	0	D

Updating Distance Vector Tables

To	Cost
A	2
B	4
C	0
D	∞
E	4

Received from C

To	Cost	Next
A	4	C
B	6	C
C	2	C
D	∞	C
E	6	C

A's modified table

To	Cost	Next
A	0	—
B	5	—
C	2	—
D	3	—
E	∞	—

A's old table

Compare

To	Cost	Next
A	0	—
B	5	—
C	2	—
D	3	—
E	6	C

A's new table

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

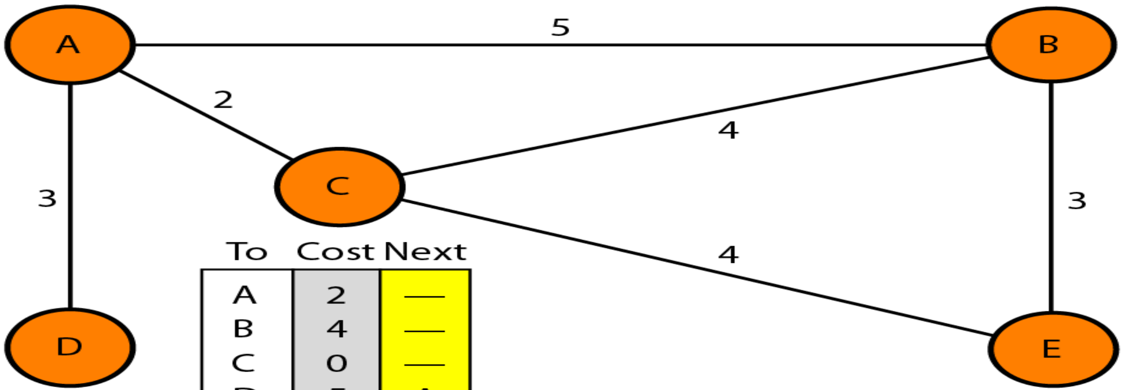
Distance Vector Tables

To	Cost	Next
A	0	—
B	5	—
C	2	—
D	3	—
E	6	C

A's table

To	Cost	Next
A	5	—
B	0	—
C	4	—
D	8	A
E	3	—

B's table



To	Cost	Next
A	3	—
B	8	A
C	5	A
D	0	—
E	9	A

D's table

To	Cost	Next
A	2	—
B	4	—
C	0	—
D	5	A
E	4	—

C's table

To	Cost	Next
A	6	C
B	3	—
C	4	—
D	9	C
E	0	—

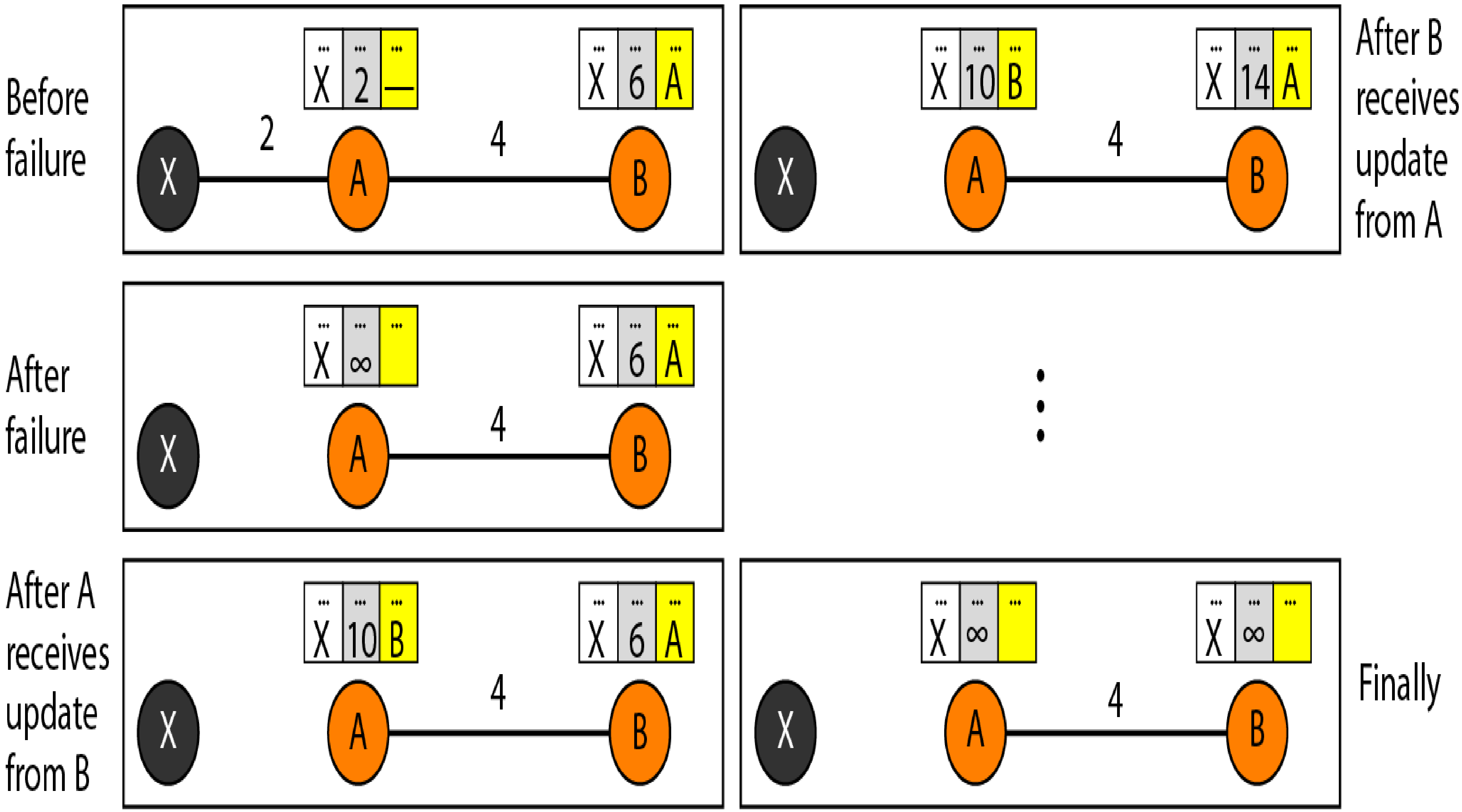
E's table

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:

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

Count-to-∞ Problem (Instability)



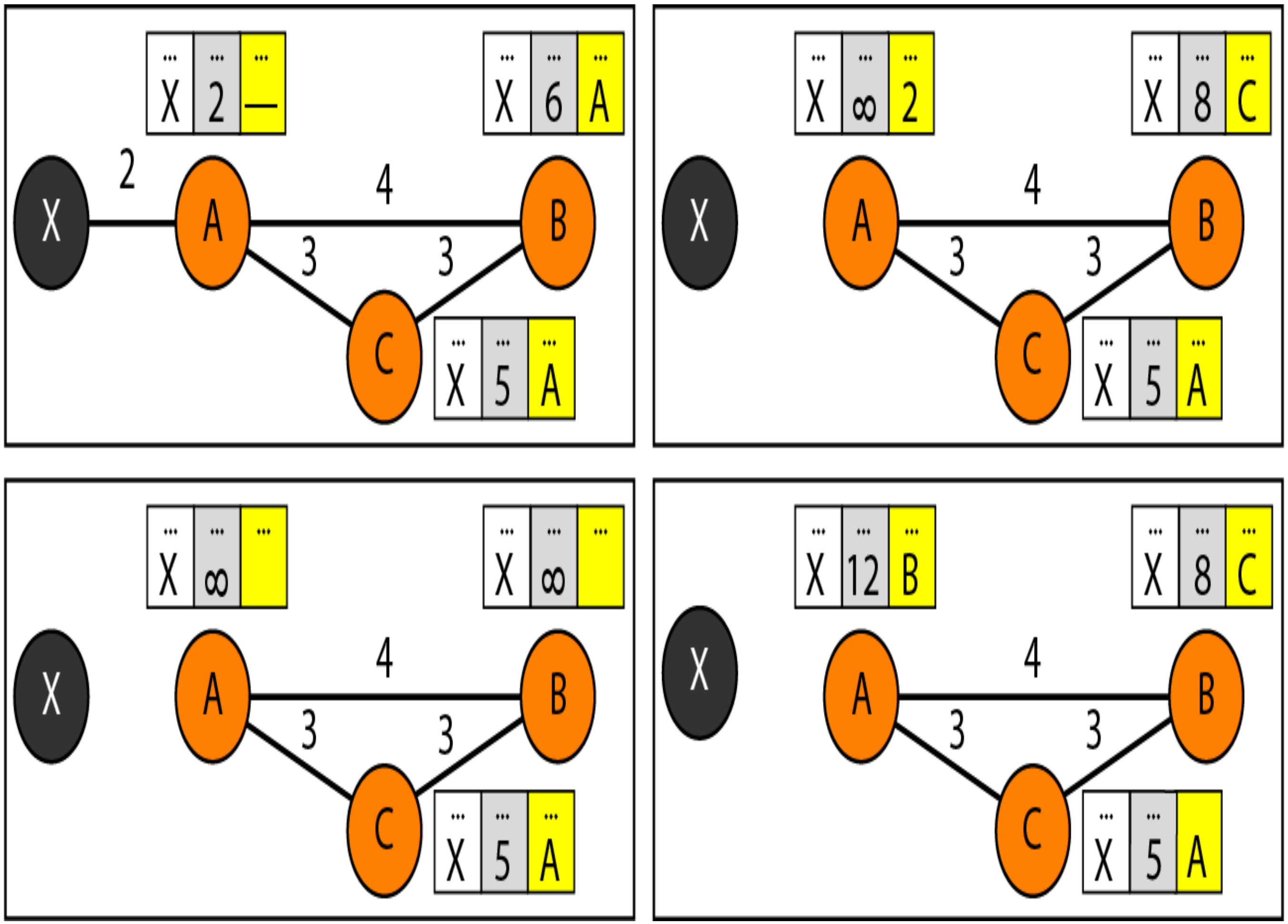
Three-Node Instability

Before failure

After B sends the route to A

After A sends the route to B and C, but the packet to C is lost

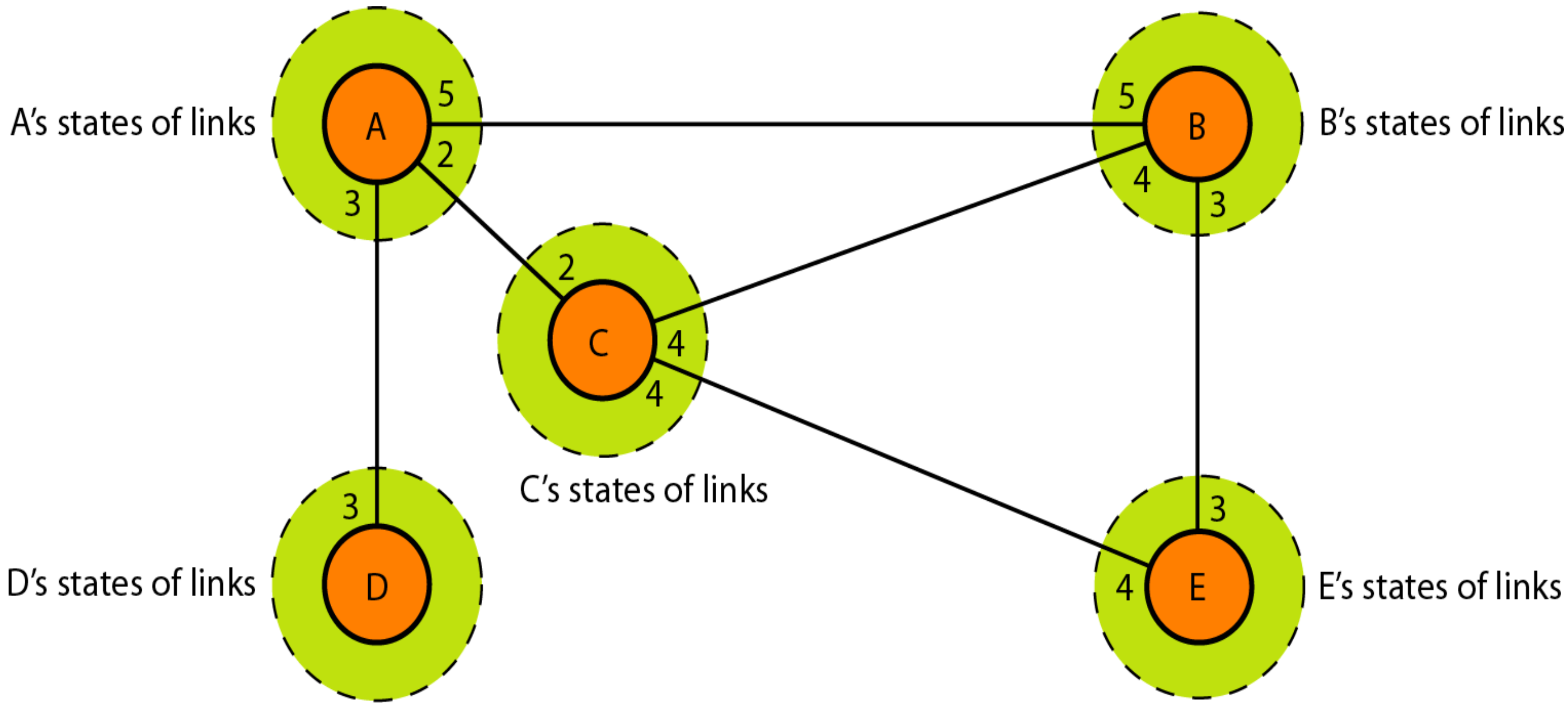
After C sends the route to B



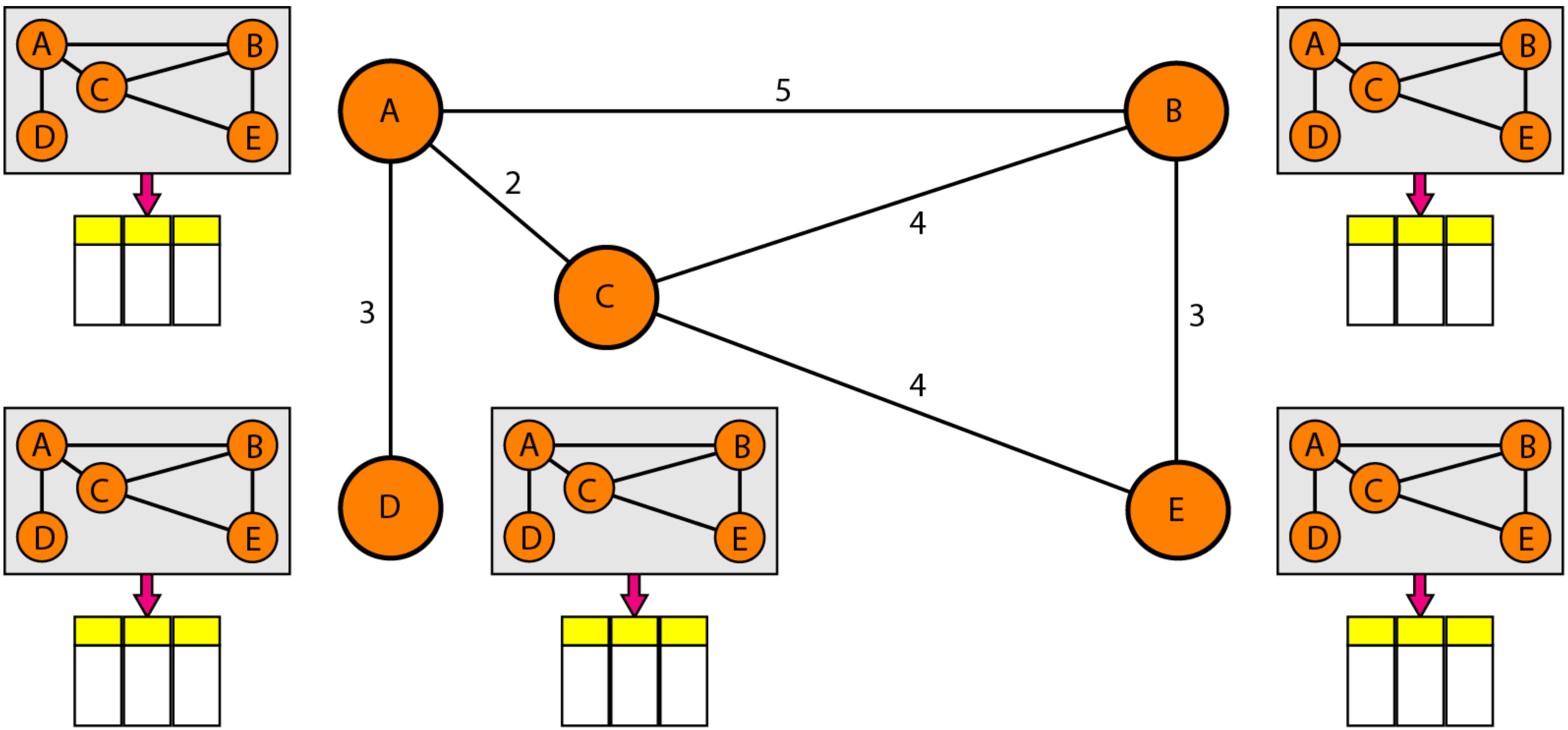
Link State (LS) Routing

- Link-State (LS) Protocols
  - Based on an algorithm by Dijkstra
  - Each router on the network is assumed to know the state of the links to all its neighbors (Cost, Operating Status, Bandwidth, Delay, etc…)
  - Each router will disseminate (via reliable flooding of link state packets, LSPs) the information about its link states to all routers in the network.
  - In this case, every router will have enough information to build a complete map of the network and therefore is able to construct a Shortest Path Spanning Tree from itself to every other router

Link State Knowledge



After dissemination of Link States



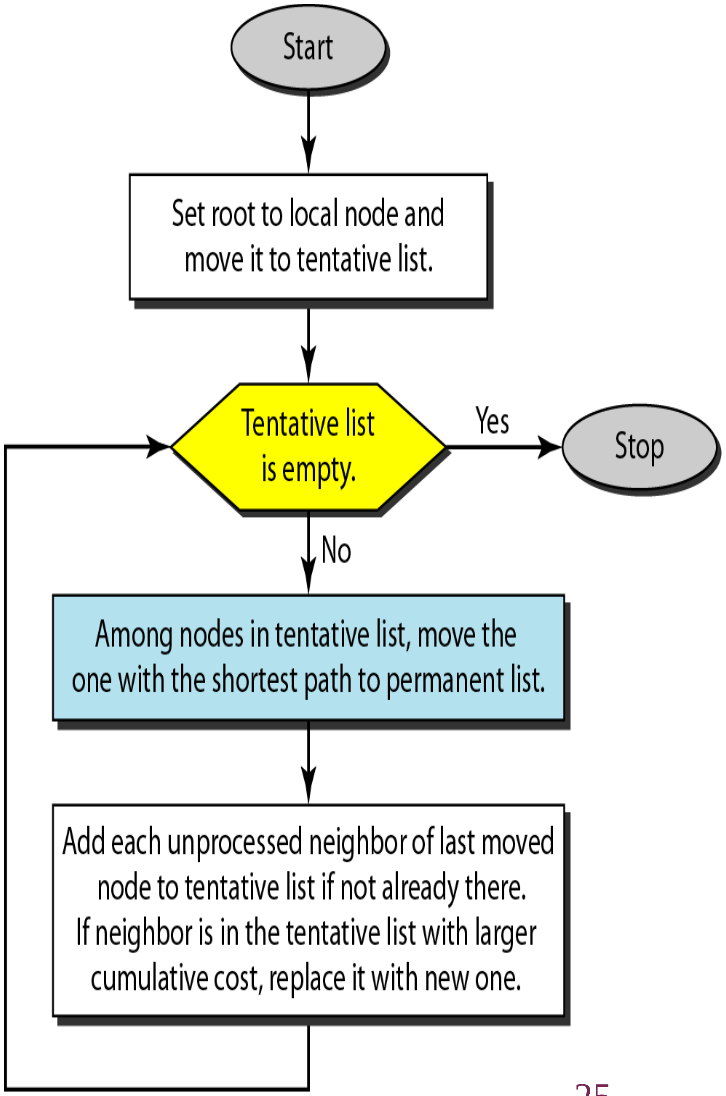
Every Router will create its own table based on the map and does not exchange tables with other routers

Link State Packets

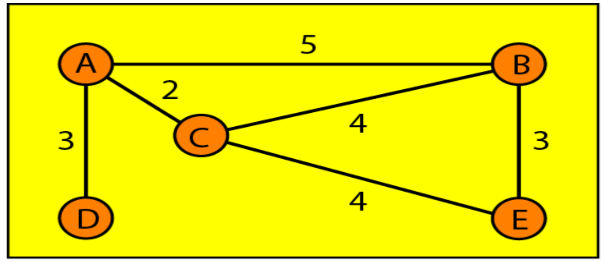
- The link state packets consist of the following information:
  - The address of the node creating the LSP
  - A list of directly connected neighbors to that node with the cost of the link to each neighbor
  - A sequence number to make sure it is the most recent one
  - A time-to-live to insure that an LSP doesn't circulate indefinitely
- A node (router) will only send an LSP if there is a failure (change of status) to some of its links or if a timer expires

Dijkstra Algorithm

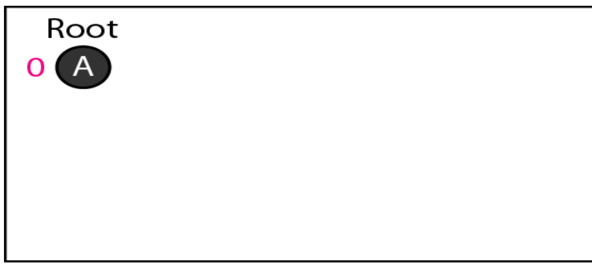
- SPT = {a}
- for all nodes v
  - if v adjacent to a then D(v) = cost (a, v)
  - else D(v) = infinity
- Loop
  - find w not in SPT, where D(w) is min
  - add w in SPT
  - for all v adjacent to w and not in SPT
    - D(v) = min (D(v), D(w) + C(w, v))
- until all nodes are in SPT



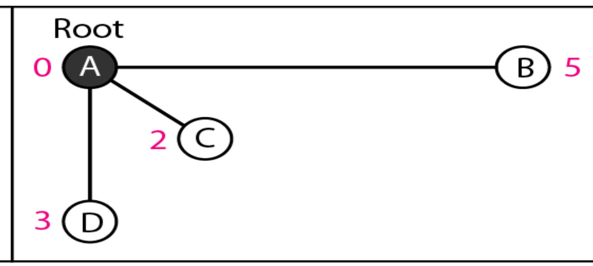
Example on Dijkstra Algorithm



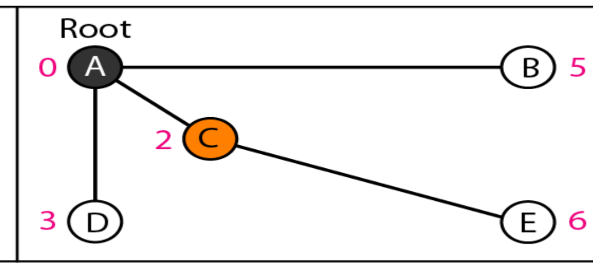
Topology



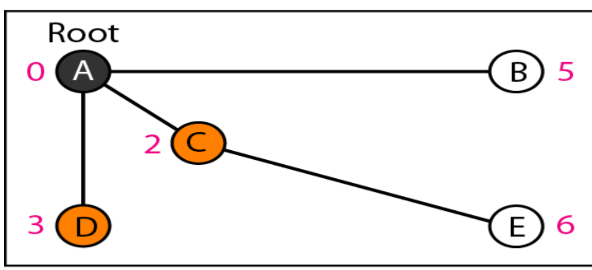
1. Set root to A and move A to tentative list.



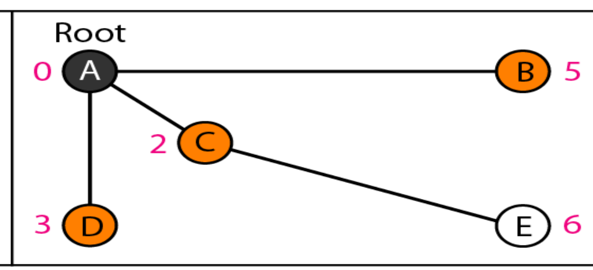
2. Move A to permanent list and add B, C, and D to tentative list.



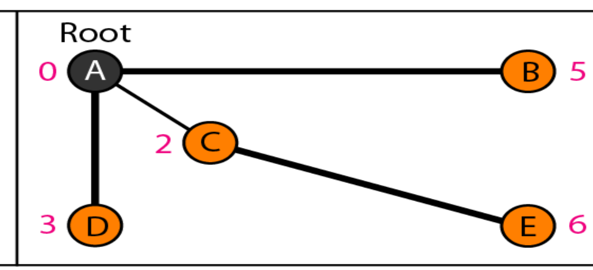
3. Move C to permanent and add E to tentative list.



4. Move D to permanent list.

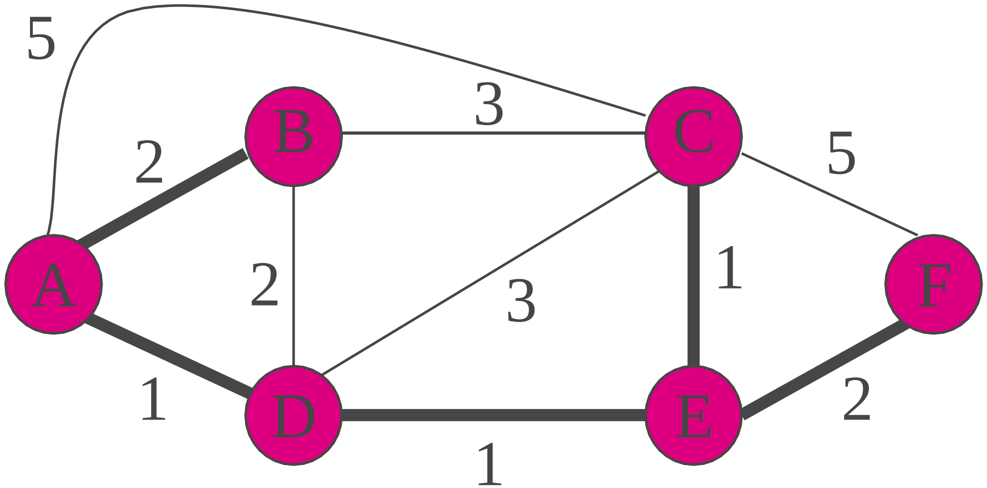


5. Move B to permanent list.



6. Move E to permanent list (tentative list is empty).

Another Example



		B	C	D	E	F
step	SPT	D(b), P(b)	D(c), P(c)	D(d), P(d)	D(e), P(e)	D(f), P(f)
0	A	2, A	5, A	1, A	~	~
1	AD	2, A	4, D		2, D	~
2	ADE	2, A	3, E			4, E
3	ADEB		3, E			4, E
4	ADEBC					4, E
5	ADEBCF					