COMP 3721
Introduction to Data Communications

12a - Week 12 - Part 1

Learning Outcomes

- By the end of this lecture, you will be able to
  - Explain what is unicast routing.
  - Explain distance-vector and link-state routing algorithms.
  - Explain intra-domain routing protocols, including RIP and OSPF.

Introduction

- The **goal** of the **network layer** is to deliver a datagram from its source to its destination or destinations.
- **Unicast routing** (our focus)
  - If a datagram is destined for **only one destination** (one-to-one delivery).
- Multicast routing
  - If the datagram is destined for several destinations (one-to-many delivery).

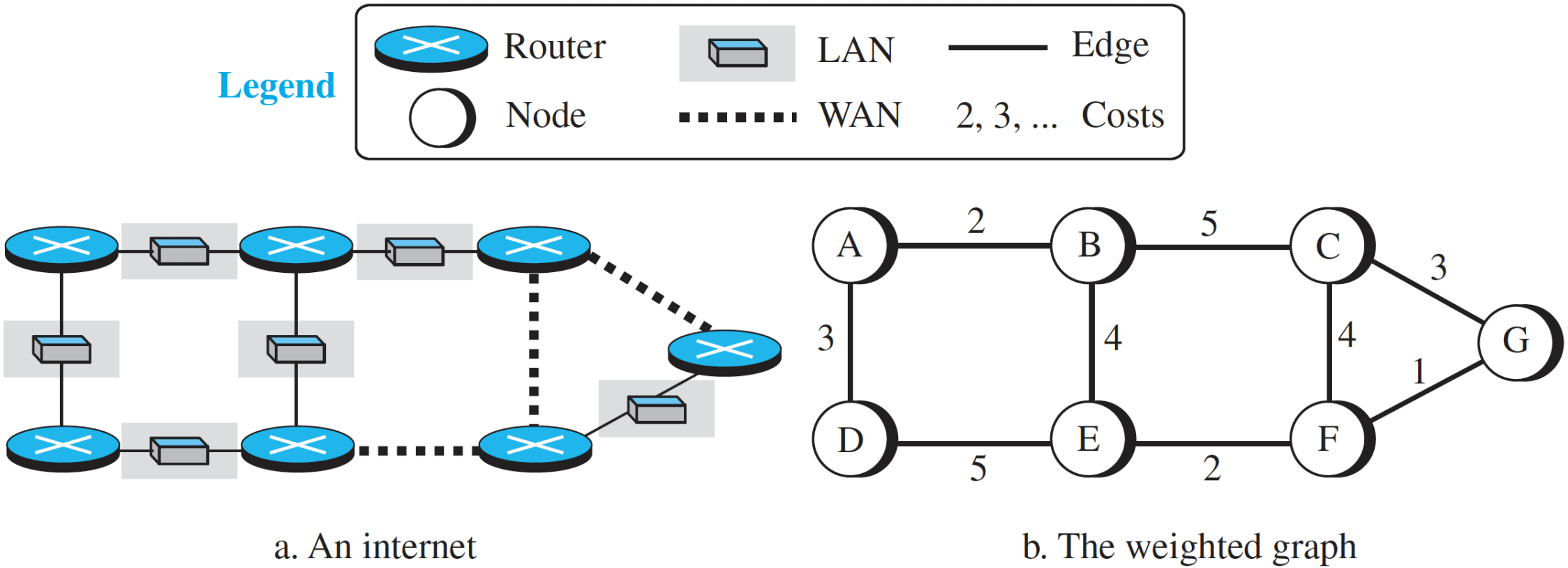
Introduction – Routing Protocols

- Routing is possible if a router has a **forwarding table** to forward a packet to the appropriate next node on its way to the final destination(s).
- To create the forwarding tables of the router, the Internet needs **routing protocols** that will be active all the time in the background and update the forwarding tables.

Unicast Routing – General Idea

- The **source host** needs **no forwarding table** because it delivers its packet to the **default router** in its local network.
- The **destination host** needs **no forwarding table** because it receives the packet from its **default router** in its local network.
  - Only the **routers** need forwarding tables.
- Routing a packet from its source to its destination means routing the packet from a **source router** (the default router of the source host) to a **destination router** (the router connected to the destination network).

An internet as a Weighted Graph



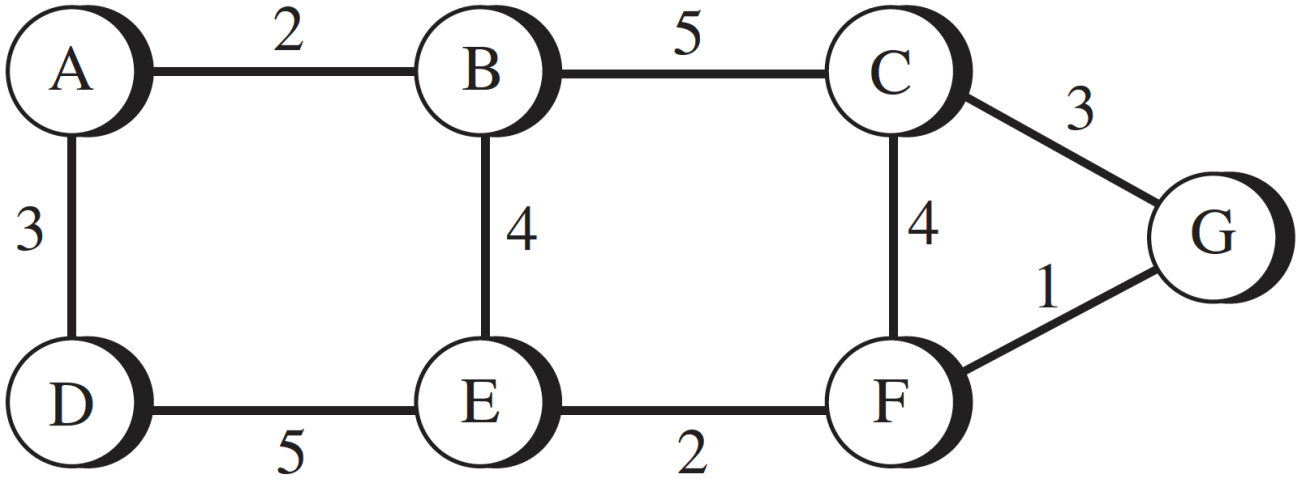
a. An internet

b. The weighted graph

To find the best route, an internet can be modeled as a weighted graph.

Least-Cost Routing

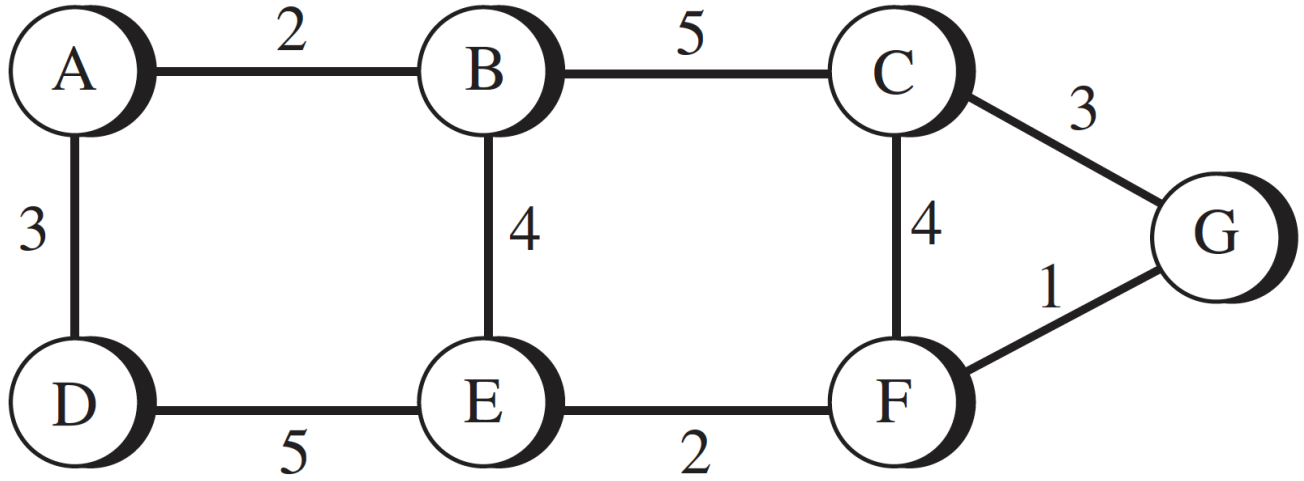
- One of the ways to interpret the **best route** from the source router to the destination router is to find the **least cost path** between the two.



Example:
What is the best route between **A** and **F**? What is its cost?

Least-Cost Routing

- One of the ways to interpret the **best route** from the source router to the destination router is to find the **least cost path** between the two.



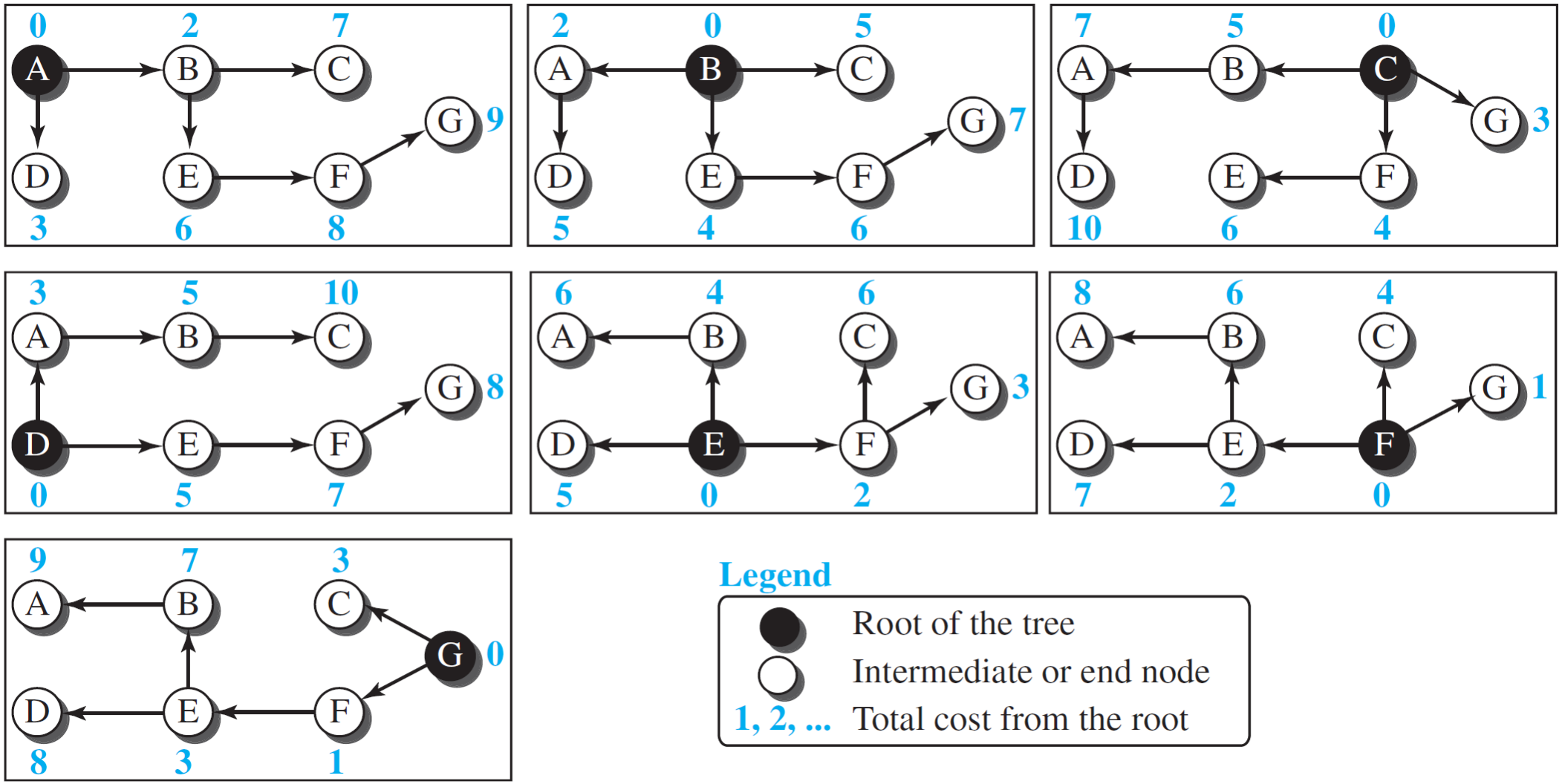
Example:
What is the best route between **A** and **F**? What is its cost?

A, B, E, F
Cost = 8

Least-Cost Trees

- A **least-cost tree** is a **tree** with the source router as the **root** that spans the whole graph (visits all other nodes)
  - in which the path between the root and any other node is the **shortest**.
    → We can have only **one shortest-path** tree for each node.
- If there are *N* routers in an internet, there are (*N* − 1) least-cost paths from each router to any other router.
  - *N* × (*N* − 1) least-cost paths for the whole internet.

Least-Cost Trees

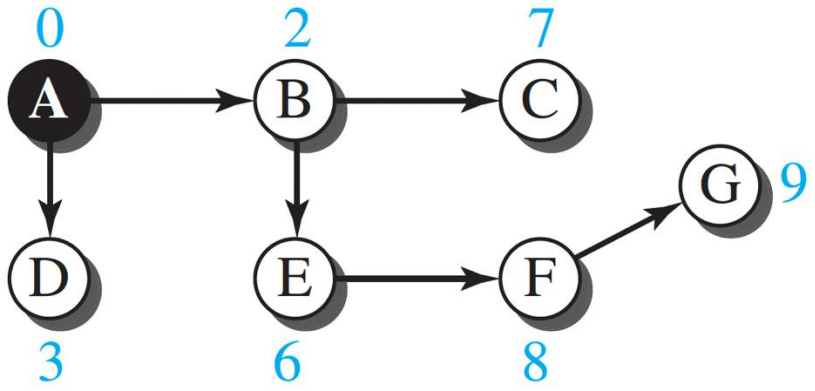


Routing Algorithms

- Two main routing algorithms:
  - **Distance Vector** (DV) **Routing Algorithm**
  - **Link-State** (LS) **Routing Algorithm**

Distance-Vector (DV) Routing Algorithm

- A **decentralized** routing algorithm.
- **Distance vector** is a **one-dimensional array** that contains the **cost** of the least-cost path to all other nodes.



a. Tree for node A

	A
A	0
B	2
C	7
D	3
E	6
F	8
G	9

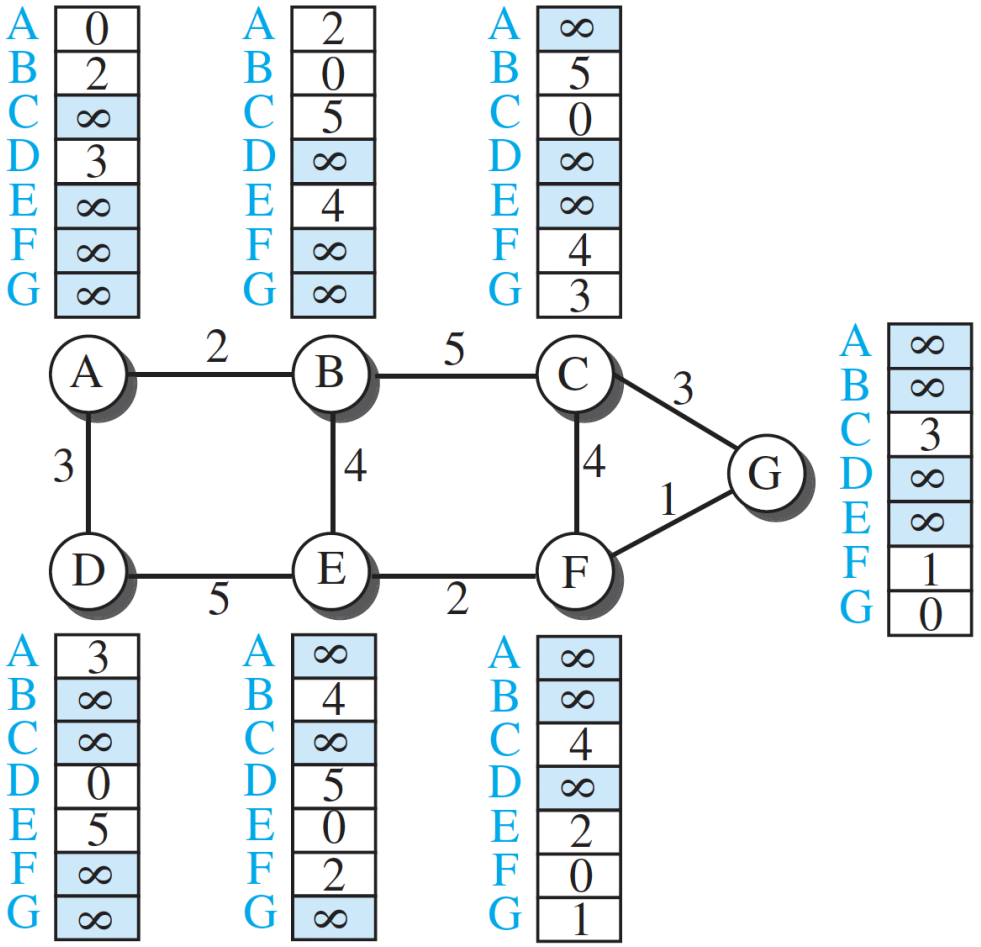
b. Distance vector for node A

Distance-Vector (DV) Routing Algorithm (Cont.)

- A distance vector **does not** give the **path** to the **destinations**; it gives only the **least costs** to the destinations. Also, a distance vector can be changed to a **forwarding table** (we will see in a moment).

First Distance Vectors for the Nodes

- The algorithm is run by each node **independently** and **asynchronously**.
- Each **node** in an internet, when it is **booted**, creates a very rudimentary distance vector with the minimum information the node can obtain **from its neighborhood**.



Updating Distance Vectors

- To improve the first vectors, the nodes in the internet need to help each other by **exchanging information**.
- After each node has created its vector, it **sends a copy of the vector** to **all its immediate neighbors**.
- After a node receives a distance vector from a neighbor, it updates its distance vector using the **Bellman-Ford equation**.
- Exchanging vectors eventually stabilizes the system and allows all nodes to find the ultimate least cost between themselves and any other nodes

Bellman-Ford Equation

- **Bellman-Ford equation** is used to find the least cost (shortest distance) between a source node, ***x***, and a destination node, ***y***, through some intermediary nodes (***a***, ***b***, ***c***, …).
  - Given the **costs** between the **source** and the **intermediary nodes** and the **least costs** between the **intermediary nodes** and the **destination**.
- The following shows the general case in which D_{ij} is the shortest distance and c_{ij} is the cost between nodes ***i*** and ***j***:

$$D_{xy} = \min\{(c_{xa} + D_{ay}), (c_{xb} + D_{by}), (c_{xc} + D_{cy})\}$$

Bellman-Ford Equation

- In distance-vector routing, **normally** we want to update **an existing least cost** with a least cost through an intermediary node, such as ***z***, if the latter is shorter.
- In this case, the equation becomes simpler, as shown below:

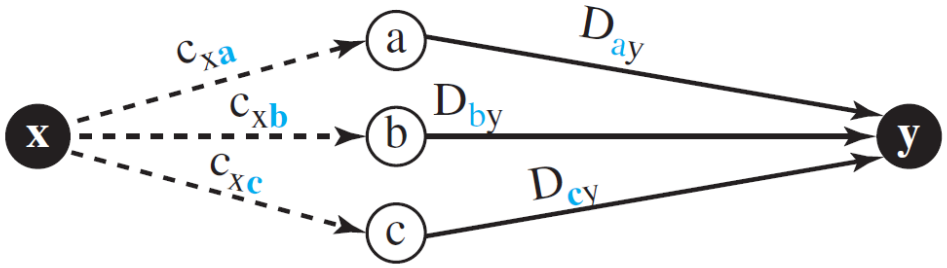
$$D_{xy} = \min\{D_{xy}, (c_{xz} + D_{zy})\}$$

Shortest distance between *x* and *y*

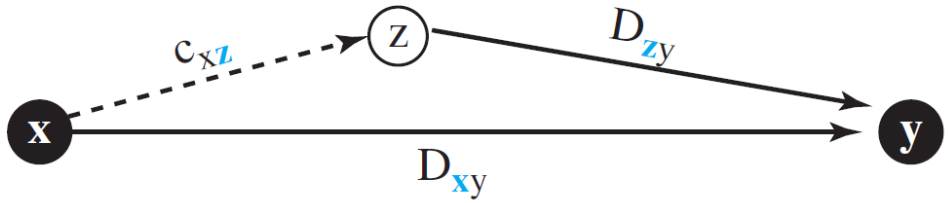
Cost between *x* and *z*

Shortest distance between *x* and *z*

Graphical Idea Behind the Bellman-Ford Equation



a. General case with three intermediate nodes



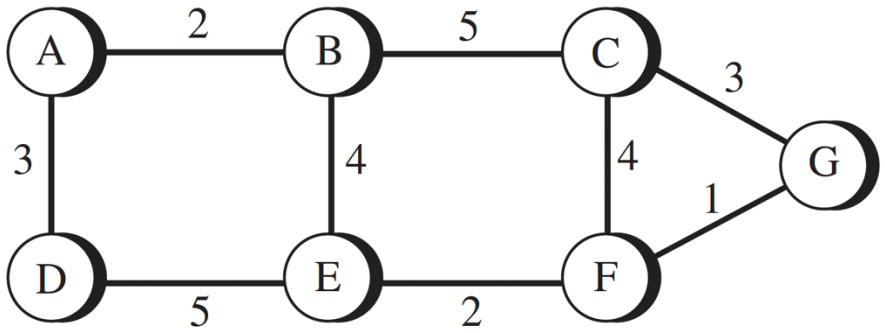
b. Updating a path with a new route

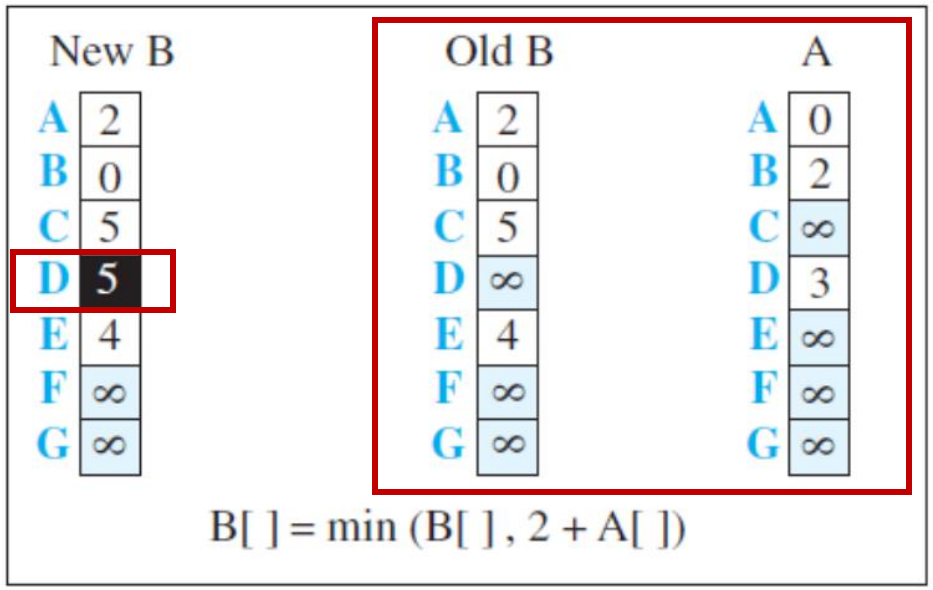
$$D_{xy} = \min\{(c_{xa} + D_{ay}), (c_{xb} + D_{by}), (c_{xc} + D_{cy})\}$$

$$D_{xy} = \min\{D_{xy}, (c_{xz} + D_{zy})\}$$

Updating Distance Vectors

Source: B
Intermediate: A
Destination: All others





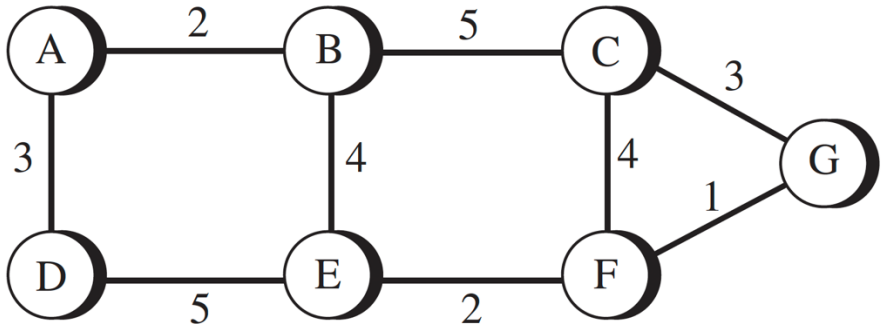
a. First event: B receives a copy of A's vector.

Note:
X[ ]: the whole vector

$$D_{BY} = \min\{D_{BY}, (c_{BA} + D_{AY})\}$$

Updating Distance Vectors

Source: B
Intermediate: E
Destination: All others



	New B	Old B	A
A	2	2	0
B	0	0	2
C	5	5	∞
D	5	∞	3
E	4	4	∞
F	∞	∞	∞
G	∞	∞	∞

B[ ] = min (B[ ] , 2 + A[ ])

a. First event: B receives a copy of A's vector.

Note:
X[ ]: the whole vector

	New B	Old B	E
A	2	2	∞
B	0	0	4
C	5	5	∞
D	5	5	5
E	4	4	0
F	6	∞	2
G	∞	∞	∞

B[ ] = min (B[ ] , 4 + E[ ])

b. Second event: B receives a copy of E's vector.

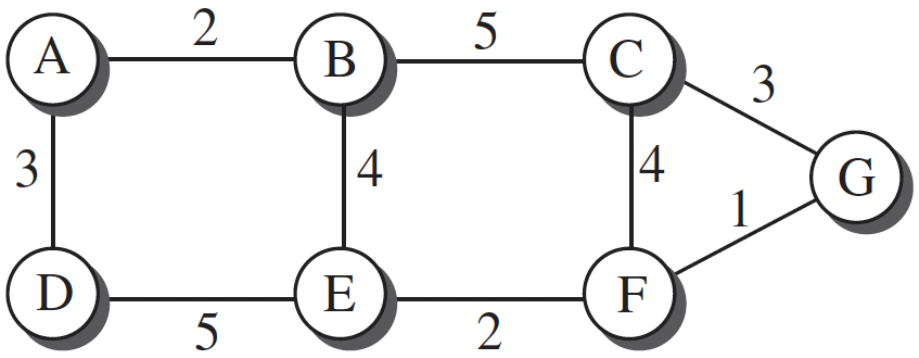
$$D_{BY} = \min\{D_{BY}, (c_{BE} + D_{EY})\}$$

Link-State Routing (LS) Algorithm

- A **centralized** routing algorithm that computes the least-cost path between a source and destination using complete, global knowledge about the network.
- **LS** and **DV** goals are to create the least-cost trees and forwarding tables for each node.
- In LS the cost associated with **an edge** defines the state of the link.
- Links with **lower costs** are **preferred** to links with **higher costs**; if the cost of a link is **infinity**, it means that the **link does not exist** or has been **broken**.
- To create a least-cost tree with this method each node needs to have a complete **map** of the network.

Link-State Database (LSDB)

- The **network topology** and all link costs are known, that is, available as **input to the LS algorithm**.
- **Link-State Database** (**LSDB**): the collection of states for all links



a. The weighted graph

	A	B	C	D	E	F	G
A	0	2	∞	3	∞	∞	∞
B	2	0	5	∞	4	∞	∞
C	∞	5	0	∞	∞	4	3
D	3	∞	∞	0	5	∞	∞
E	∞	4	∞	5	0	2	∞
F	∞	∞	4	∞	2	0	1
G	∞	∞	3	∞	∞	1	0

b. Link state database

Link-State Database (LSDB)

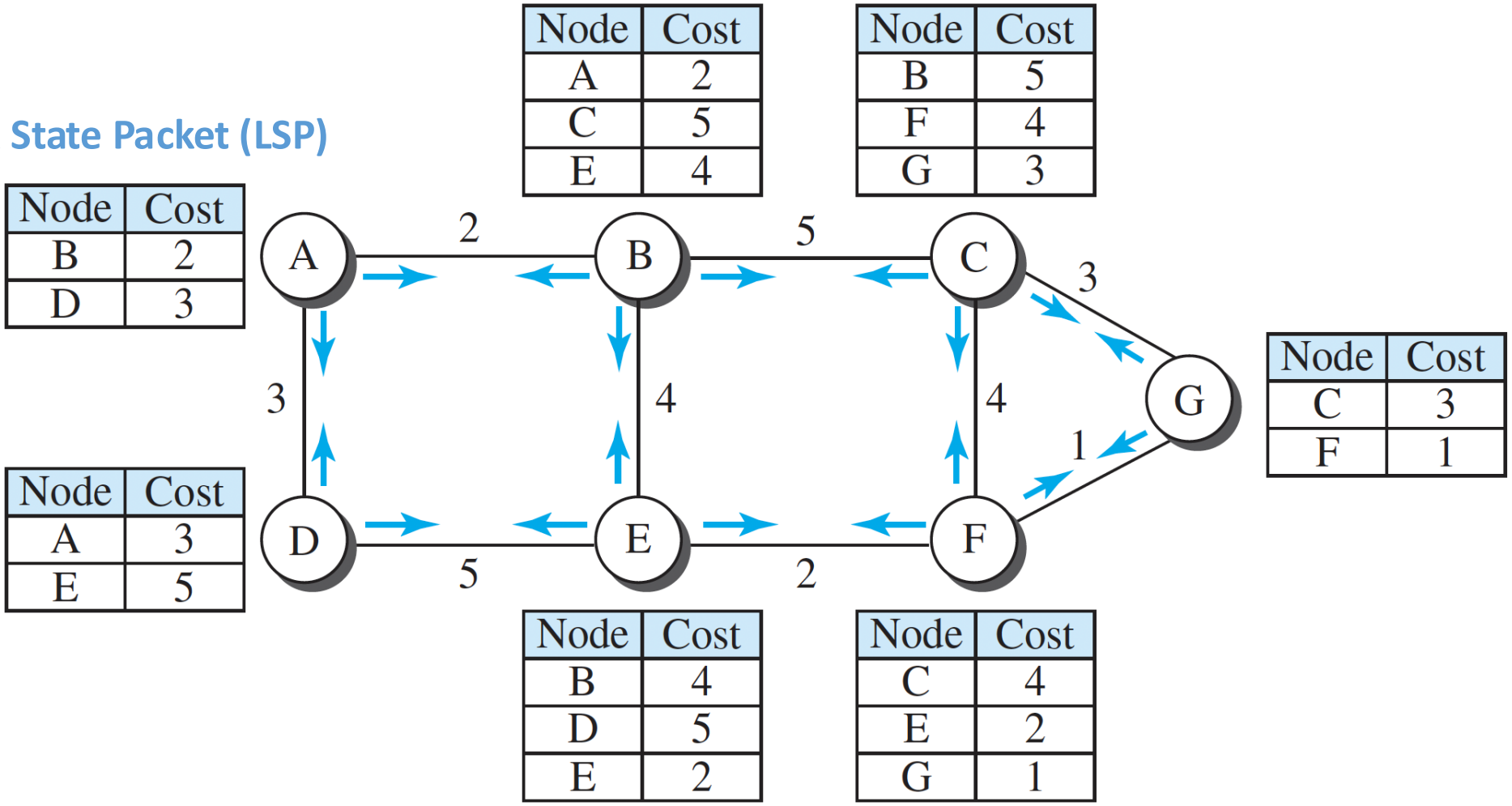
- There is **only one** LSDB for the whole internet; each node needs to have a **duplicate** of it to be able to create the least-cost tree.
- The LSDB can be represented as a **two-dimensional array** (matrix).
  - The value of each cell defines the **cost of the corresponding link**.

Link-State Database (LSDB) (Cont.)

- How can each node create this LSDB that contains information about the whole internet?
  - A process called **flooding** (**broadcasting**).
  - Each node can send some greeting messages to all its immediate neighbors to collect the **identity of the node** and the **cost of the link**.
  - The combination of above two pieces of information is called the **Link State Packet** (**LSP**).
  - If the received LSP is newer or the first one received, the node discards the old LSP (if there is one) and keeps the received one.
  - Then it **sends a copy** of it out of **each interface** except the one from which the packet arrived.

Building LSDB using LSPs

Link State Packet (LSP)



Formation of Least-Cost Trees

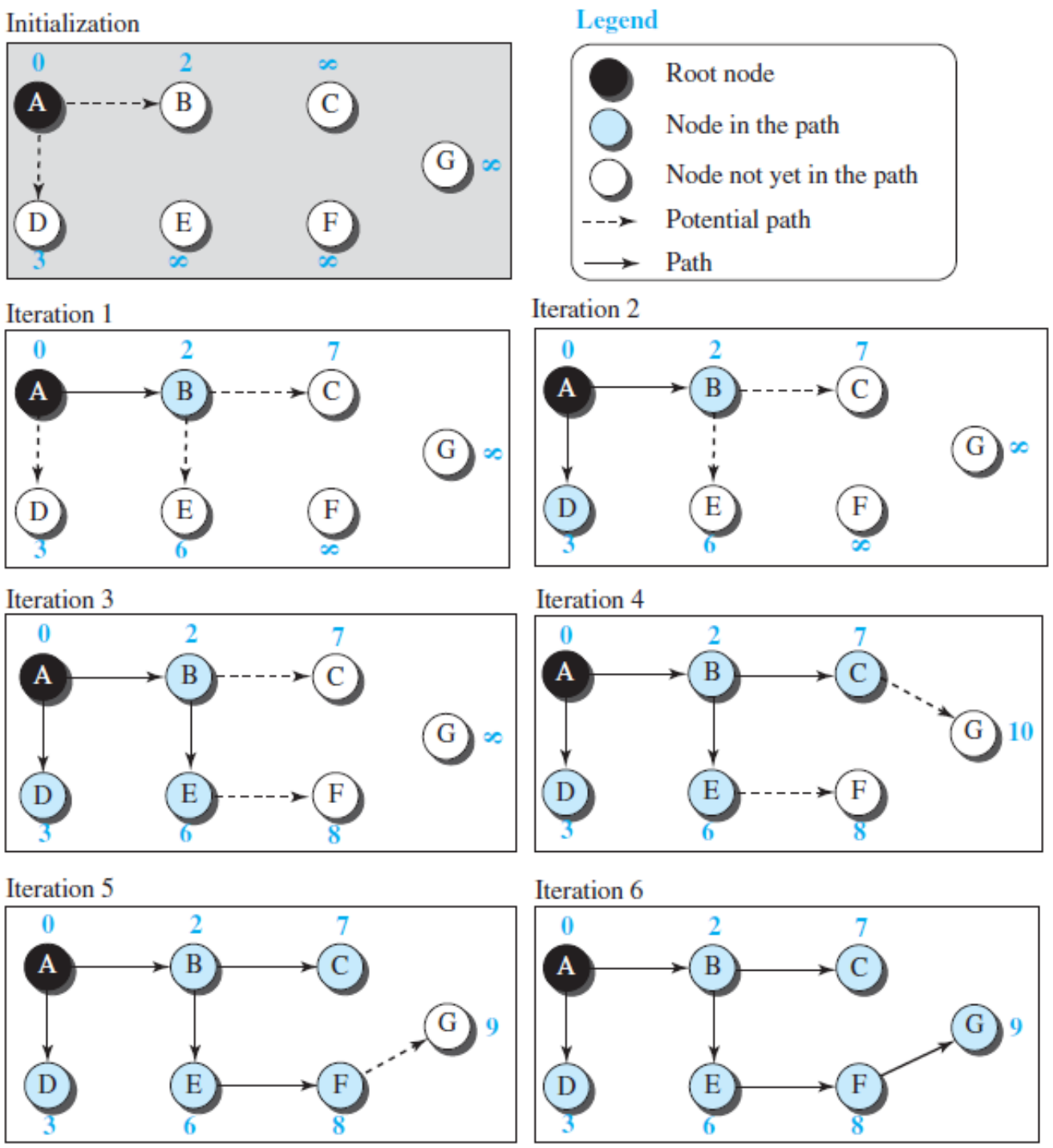
- To create a least-cost tree for itself, using the shared LSDB, **each node** runs the **Dijkstra Algorithm**.
- Dijkstra's algorithm is a solution to the single-source shortest path problem in graph theory.
- Works on both directed and undirected graphs. However, all edges must have nonnegative weights.
- **Approach**: Greedy
- **Input**: Weighted graph $G = \{E, V\}$ and source vertex $v \in V$, such that all edge weights are nonnegative.
- **Output**: Lengths of shortest paths (or the shortest paths themselves) from a given source vertex $v \in V$ to all other vertices.

Running Dijkstra Algorithm at Node A

- In the right-side figure, the number written above each node is the **shortest distance** from the **root** to that **node**.

	A	B	C	D	E	F	G
A	0	2	∞	3	∞	∞	∞
B	2	0	5	∞	4	∞	∞
C	∞	5	0	∞	∞	4	3
D	3	∞	∞	0	5	∞	∞
E	∞	4	∞	5	0	2	∞
F	∞	∞	4	∞	2	0	1
G	∞	∞	3	∞	∞	1	0

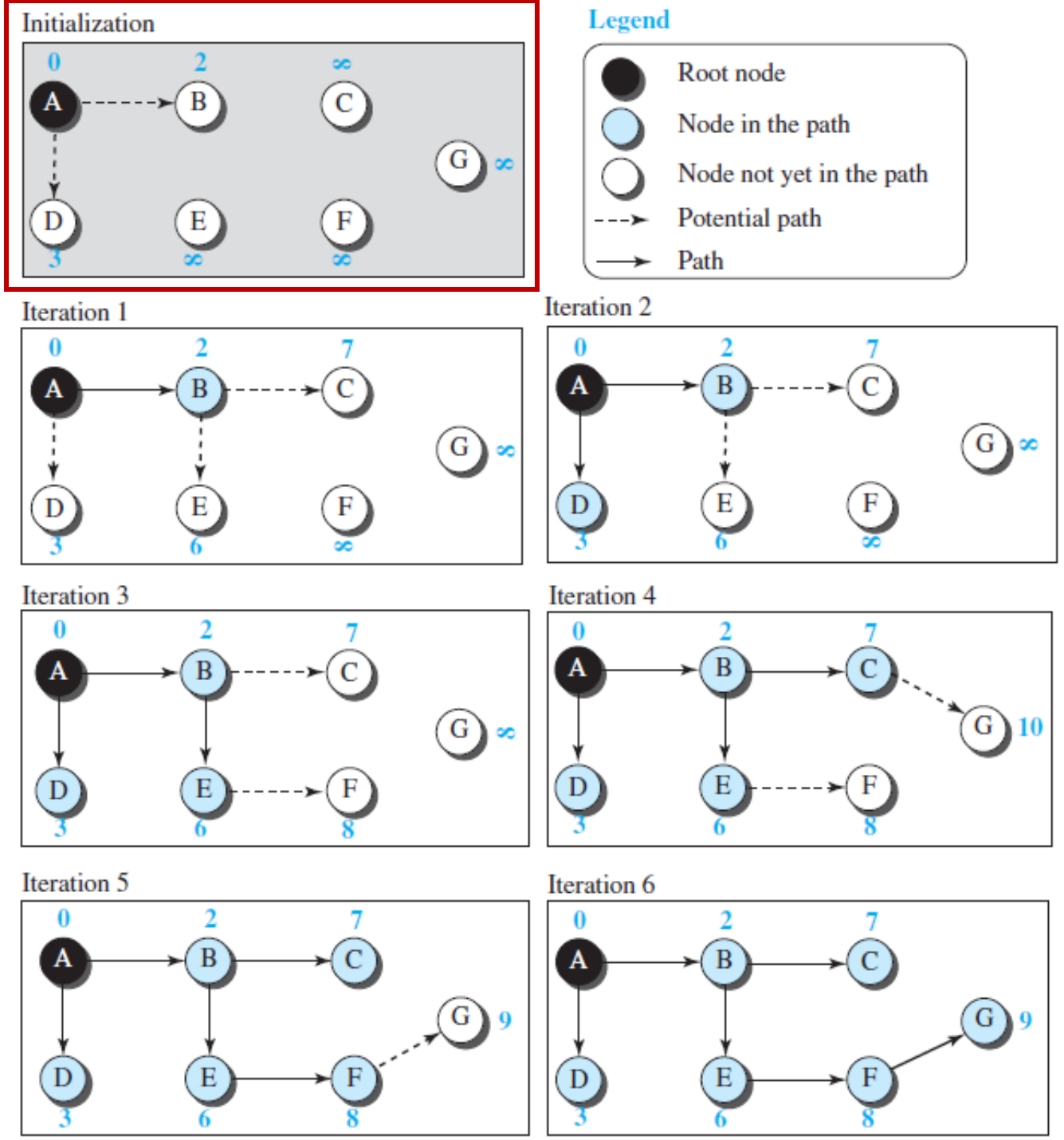
Shared LSDB



Running Dijkstra Algorithm at Node A

	A	B	C	D	E	F	G
A	0	2	∞	3	∞	∞	∞
B	2	0	5	∞	4	∞	∞
C	∞	5	0	∞	∞	4	3
D	3	∞	∞	0	5	∞	∞
E	∞	4	∞	5	0	2	∞
F	∞	∞	4	∞	2	0	1
G	∞	∞	3	∞	∞	1	0

Shared LSDB

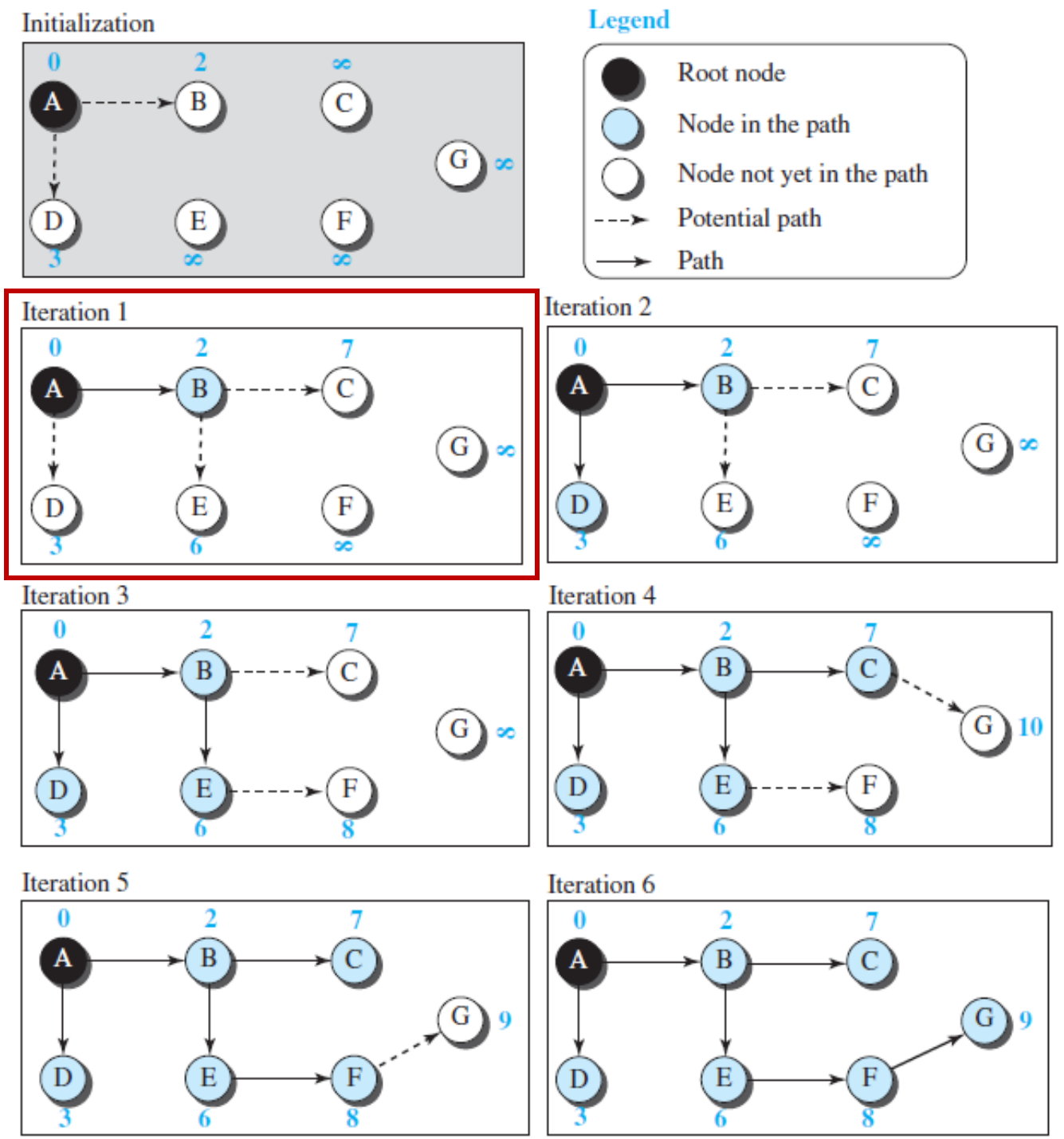


Running Dijkstra Algorithm at Node A

- B is added to the tree. C and E are its neighbors and not in the tree. The distances for C and E are updated.

	A	B	C	D	E	F	G
A	0	2	∞	3	∞	∞	∞
B	2	0	5	∞	4	∞	∞
C	∞	5	0	∞	∞	4	3
D	3	∞	∞	0	5	∞	∞
E	∞	4	∞	5	0	2	∞
F	∞	∞	4	∞	2	0	1
G	∞	∞	3	∞	∞	1	0

Shared LSDB

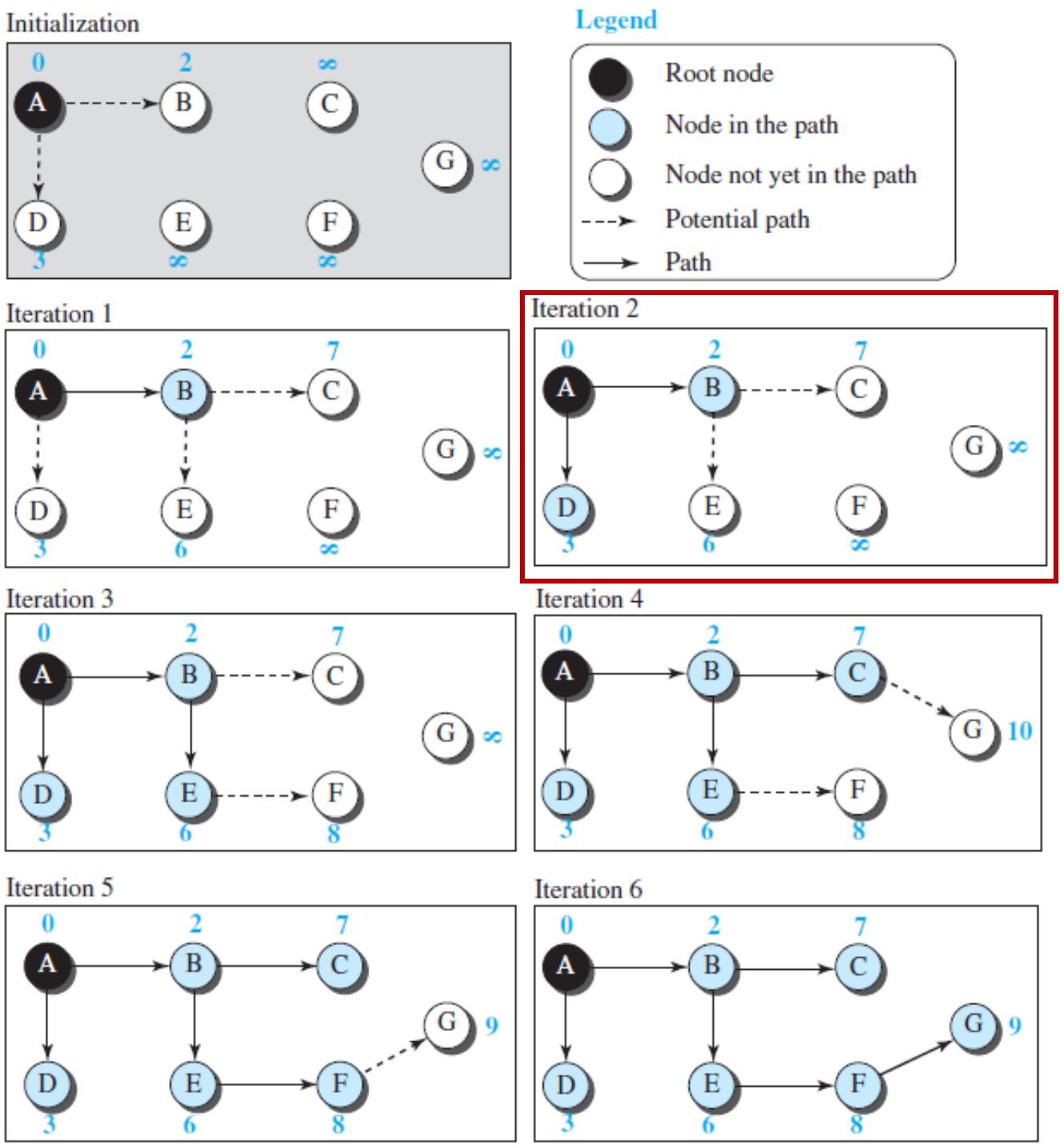


Running Dijkstra Algorithm at Node A

- D is added to the tree. E is its neighbor and not in the tree. No change in the cost of E.

	A	B	C	D	E	F	G
A	0	2	∞	3	∞	∞	∞
B	2	0	5	∞	4	∞	∞
C	∞	5	0	∞	∞	4	3
D	3	∞	∞	0	5	∞	∞
E	∞	4	∞	5	0	2	∞
F	∞	∞	4	∞	2	0	1
G	∞	∞	3	∞	∞	1	0

Shared LSDB

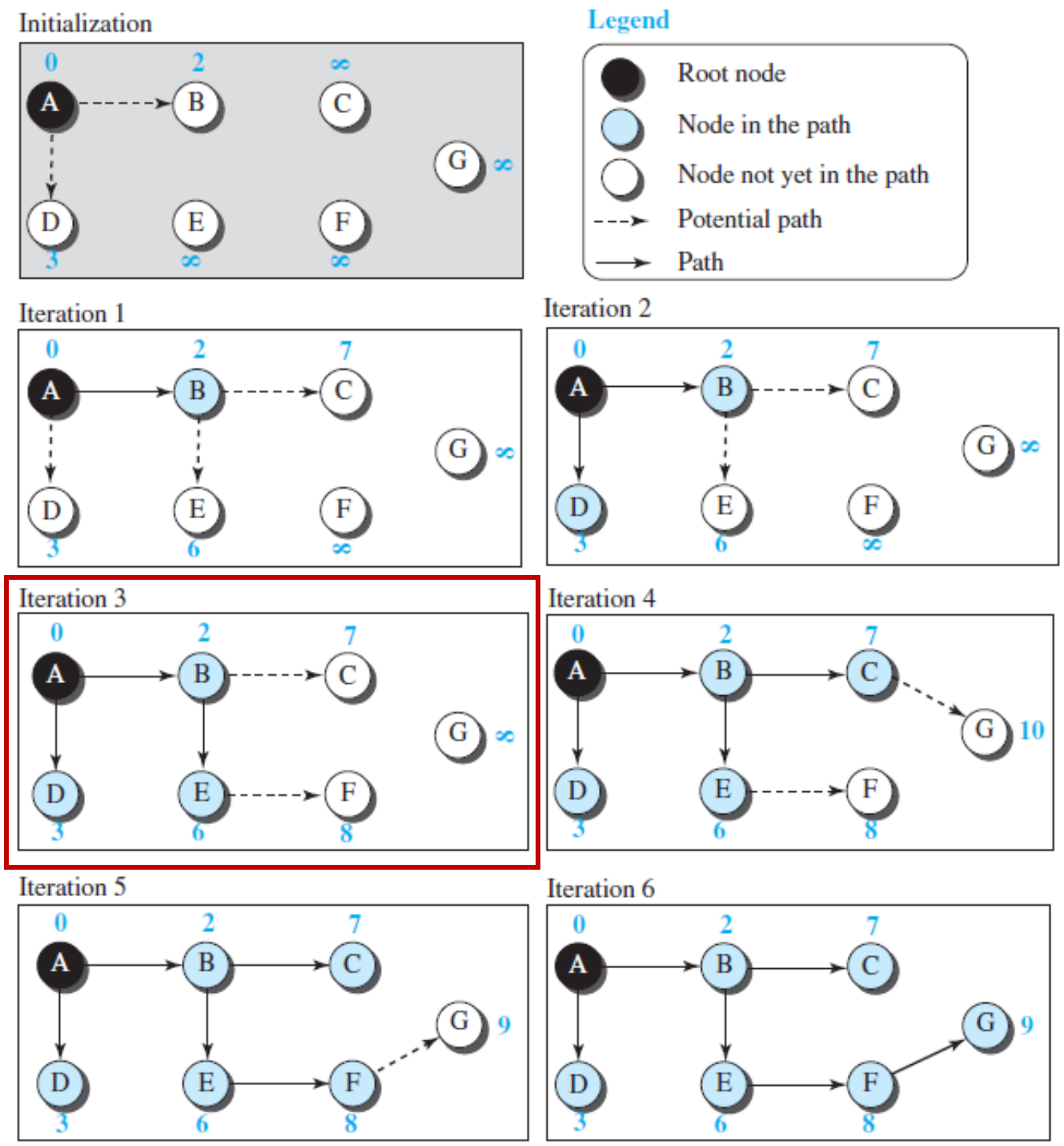


Running Dijkstra Algorithm at Node A

- E is added to the tree. F is its neighbor and not in the tree. The distance for F is updated.

	A	B	C	D	E	F	G
A	0	2	∞	3	∞	∞	∞
B	2	0	5	∞	4	∞	∞
C	∞	5	0	∞	∞	4	3
D	3	∞	∞	0	5	∞	∞
E	∞	4	∞	5	0	2	∞
F	∞	∞	4	∞	2	0	1
G	∞	∞	3	∞	∞	1	0

Shared LSDB

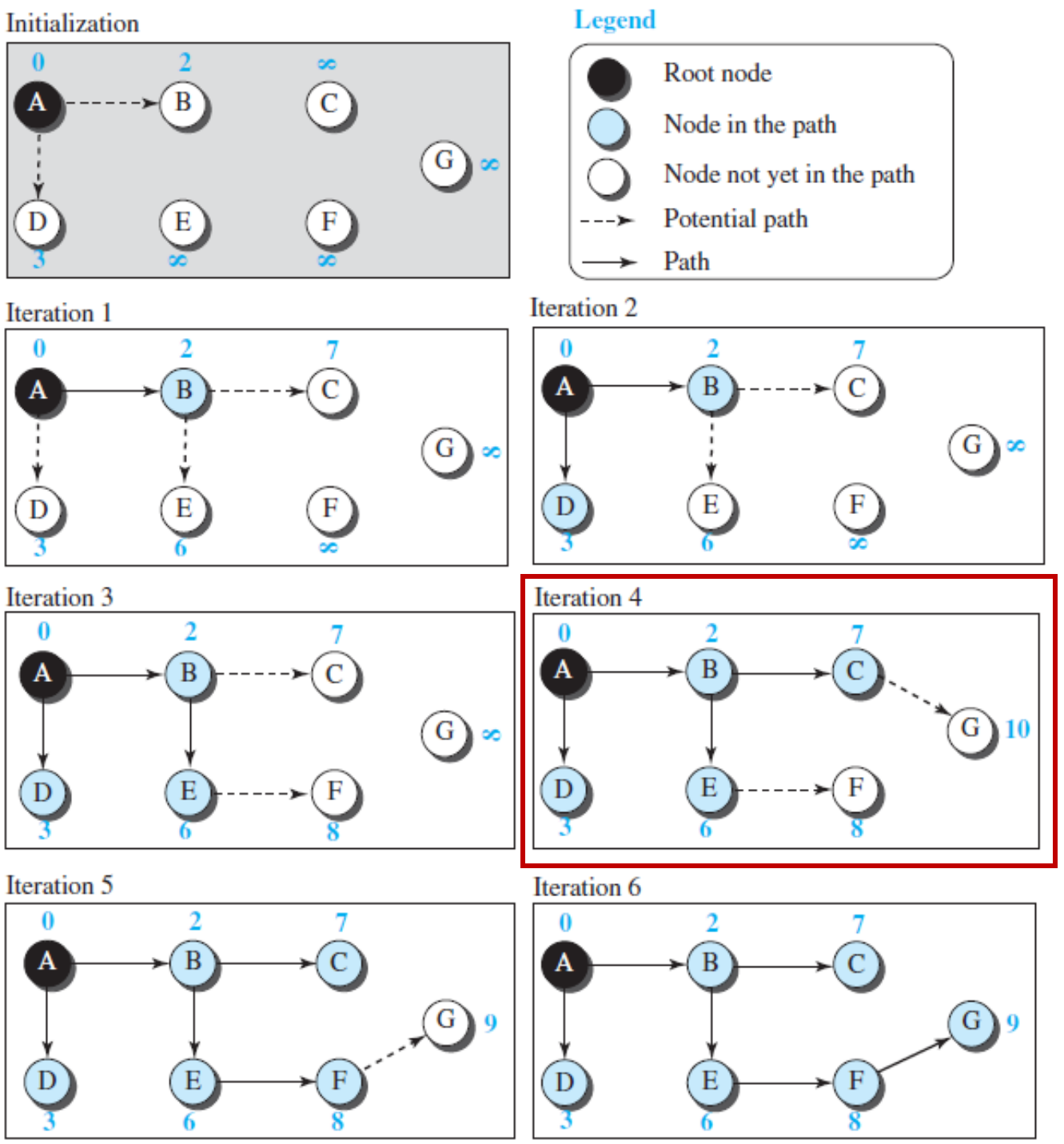


Running Dijkstra Algorithm at Node A

- C is added to the tree. G is its neighbor and not in the tree. The distance for G is updated.

	A	B	C	D	E	F	G
A	0	2	∞	3	∞	∞	∞
B	2	0	5	∞	4	∞	∞
C	∞	5	0	∞	∞	4	3
D	3	∞	∞	0	5	∞	∞
E	∞	4	∞	5	0	2	∞
F	∞	∞	4	∞	2	0	1
G	∞	∞	3	∞	∞	1	0

Shared LSDB

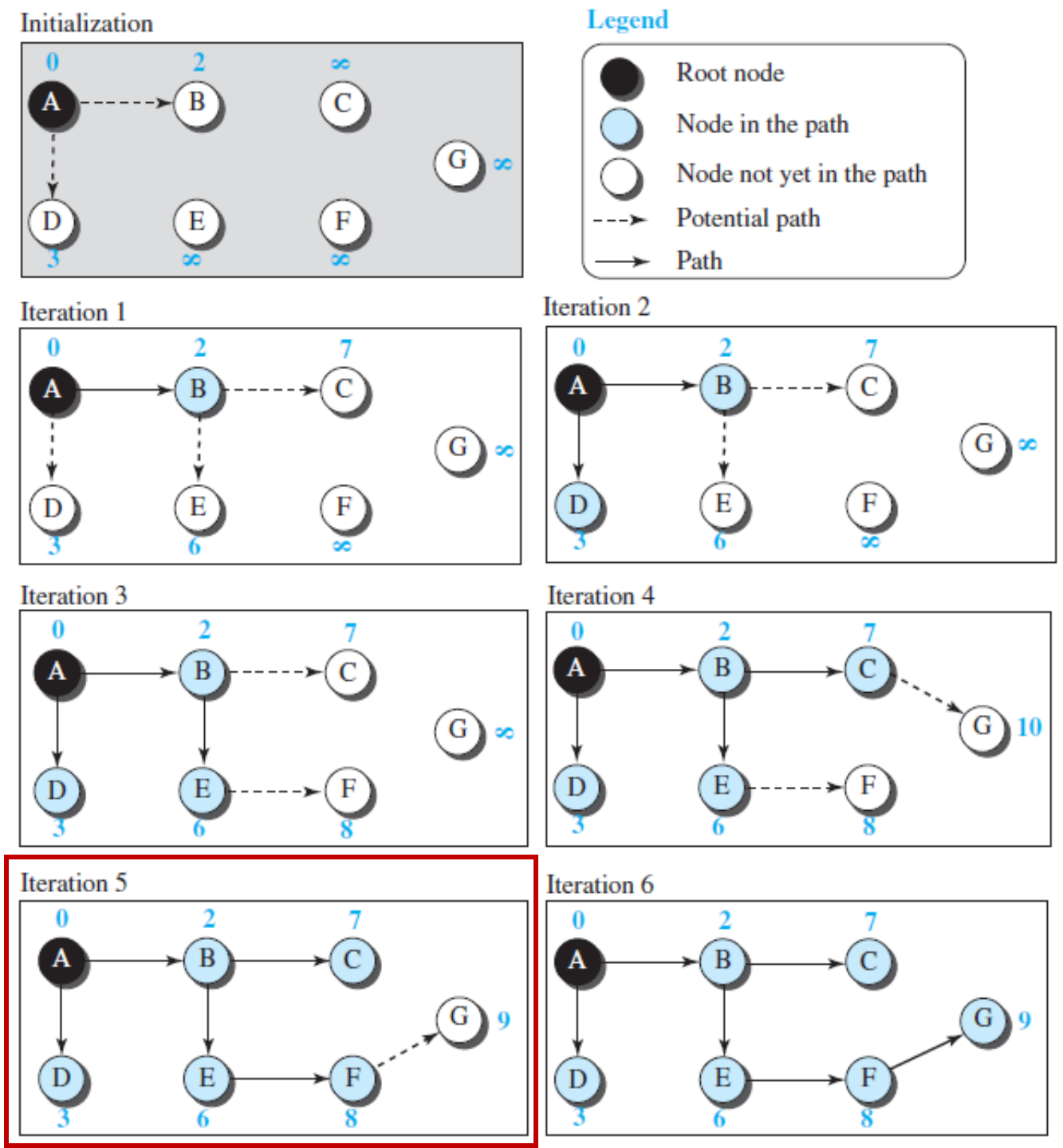


Running Dijkstra Algorithm at Node A

- F is added to the tree. G is its neighbor and not in the tree. The distance for G is updated.

	A	B	C	D	E	F	G
A	0	2	∞	3	∞	∞	∞
B	2	0	5	∞	4	∞	∞
C	∞	5	0	∞	∞	4	3
D	3	∞	∞	0	5	∞	∞
E	∞	4	∞	5	0	2	∞
F	∞	∞	4	∞	2	0	1
G	∞	∞	3	∞	∞	1	0

Shared LSDB

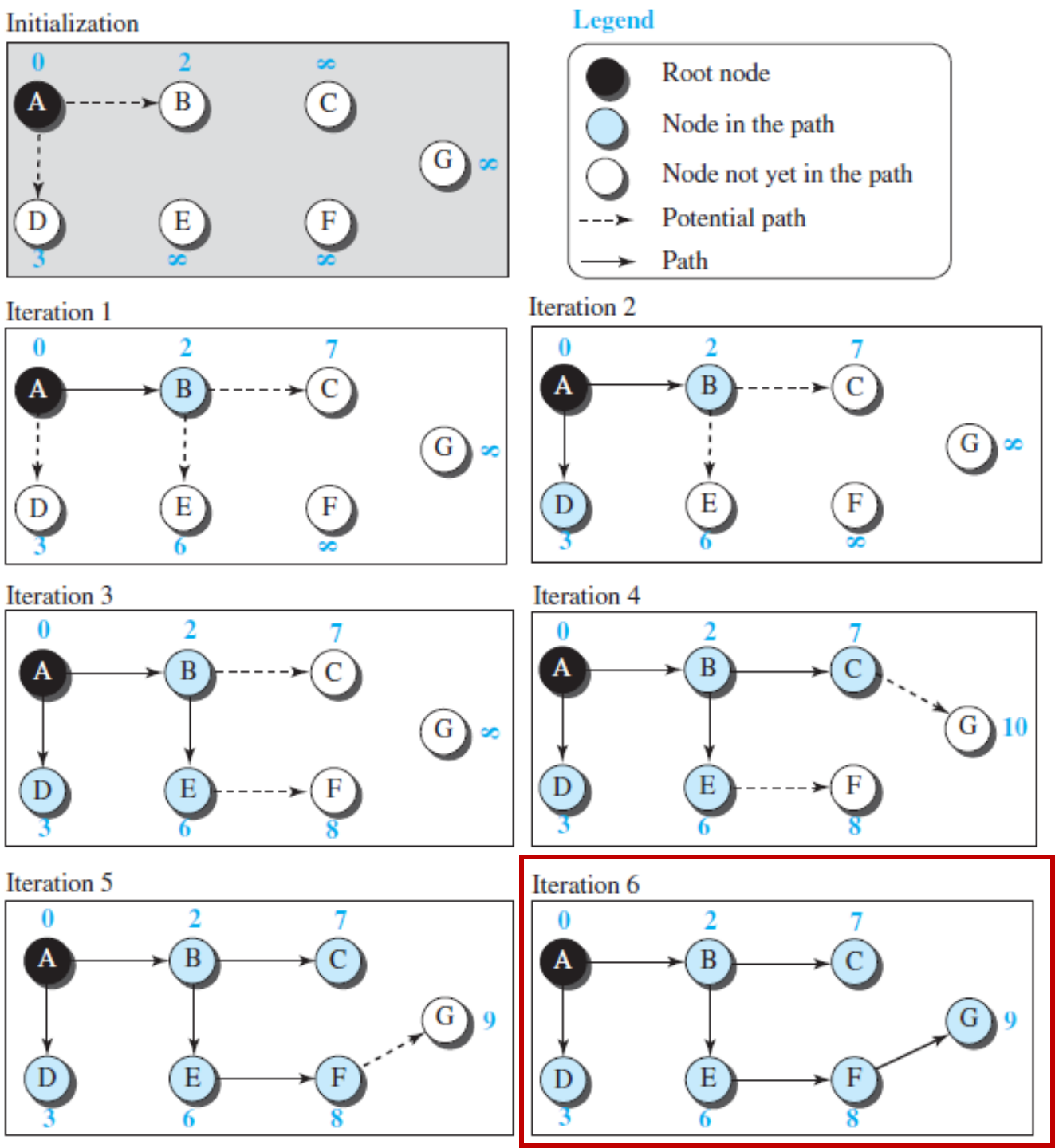


Running Dijkstra Algorithm at Node A

- G is added to the tree.

	A	B	C	D	E	F	G
A	0	2	∞	3	∞	∞	∞
B	2	0	5	∞	4	∞	∞
C	∞	5	0	∞	∞	4	3
D	3	∞	∞	0	5	∞	∞
E	∞	4	∞	5	0	2	∞
F	∞	∞	4	∞	2	0	1
G	∞	∞	3	∞	∞	1	0

Shared LSDB



Unicast Routing Protocols

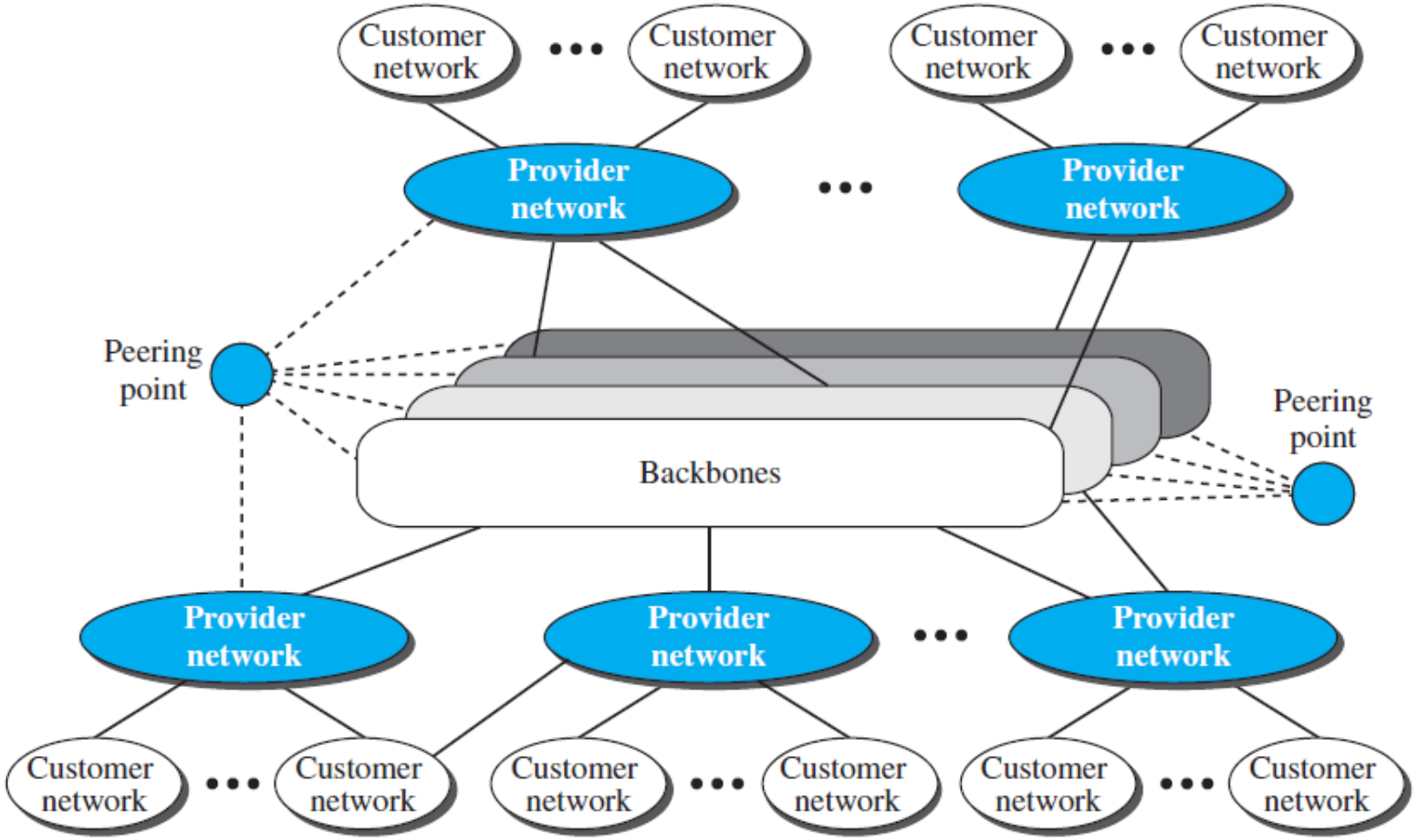
- In the previous section, we discussed unicast routing algorithms
- Now, we discuss two common protocols used in the Internet:
  1. **Routing Information Protocol** (**RIP**), based on the **distance-vector algorithm**.
  2. **Open Shortest Path First** (**OSPF**), based on the **link-state algorithm**.

- We don't discuss Border Gateway Protocol (BGP), which is based on path-vector algorithm.

Internet Structure and Routing Protocols

- A **routing protocol** on the Internet implements one of the routing **algorithms**.
- Although a routing protocol is based on a routing algorithm, it needs to define the following as well (**algorithm ≠ protocol**):
  - Its domain of operation
  - The messages exchanged
  - Communication between routers
  - Interaction with protocols in other domains
- The **structure** of today's Internet.
  - A multi-backbone structure run by different private corporations.

Internet Structure



Any of these three entities (backbone, provider network, or customer network) can be called an Internet Service Provider or ISP. They provide services, but at different levels.

Routing on the Internet

- Routing on the **Internet cannot** be done using a **single protocol**. Why?
  1. **Scalability problem**
     - The size of the **forwarding tables** becomes huge, searching for a destination in a forwarding table becomes time-consuming, and updating creates a huge amount of traffic.
  2. **Administrative issue**
     - The organization must be able to use as many subnets and routers (from any manufacturer) as it needs, may run a specific routing algorithm, etc.

Hierarchical Routing

- **Hierarchical routing** means considering each ISP as an **autonomous system** (**AS**).
  - Each AS is operated by **a single large organization**, such as an Internet service provider (ISP), a large enterprise technology company, a university, or a government agency.
  - Each AS is given an **autonomous system number** (**ASN**) by the ICANN.
    - A 16-bit unsigned integer that uniquely defines an AS (recently expanded to 32-bits as per RFC4893).
- Each AS can run a routing protocol that meets its needs, but the **global Internet** runs a global protocol to glue all ASs together.

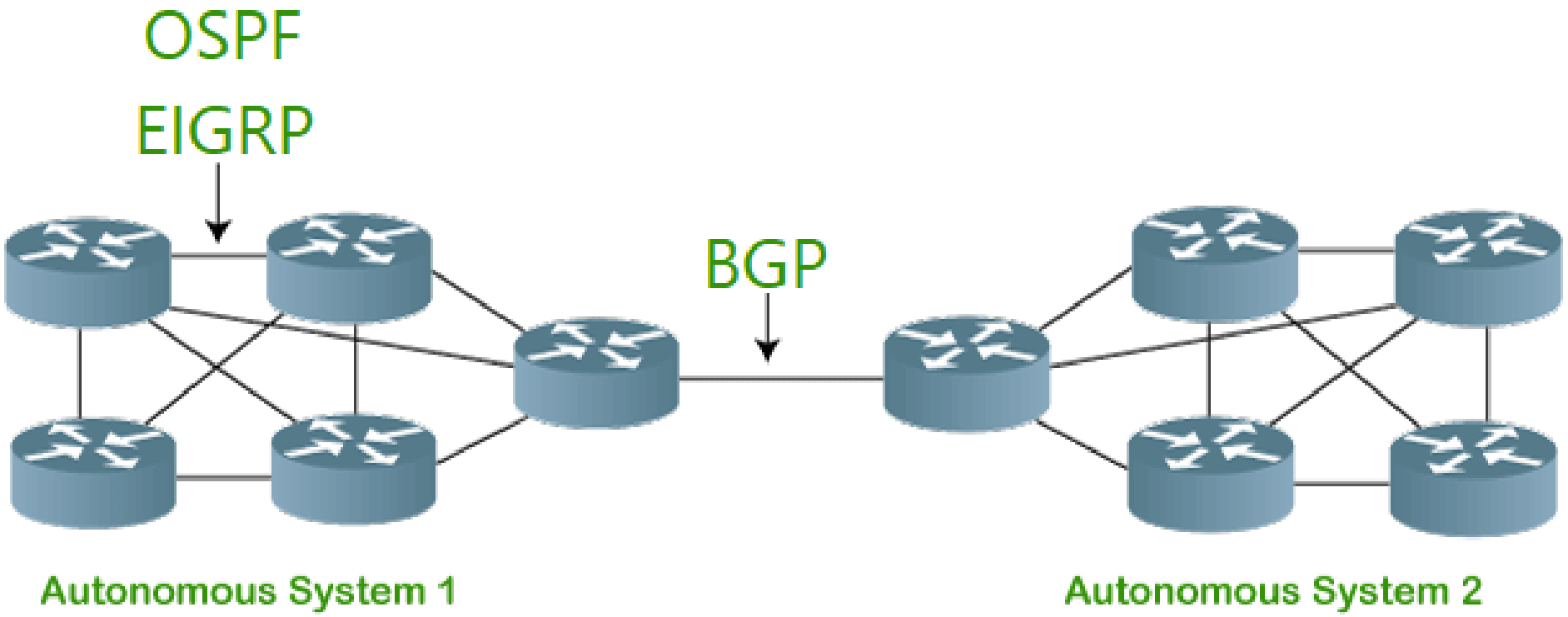
Hierarchical Routing – Intra and Inter ASs

- Intra-AS routing protocol, intradomain routing protocol, or **interior gateway protocol** (**IGP**)
  - The routing protocol run in each AS.
- Inter-AS routing protocol, interdomain routing protocol, or **exterior gateway protocol** (**EGP**)
  - The global routing protocol.

Hierarchical Routing – Intra and Inter ASs

- The two common **intradomain** routing protocols
  - **Routing Information Protocol** (**RIP**) and **Open Shortest Path First** (**OSPF**)
    - Enhanced Interior Gateway Routing Protocol (EIGRP), is also an advanced distance-vector routing protocol developed by Cisco as a proprietary solution exclusively for Cisco routers.
- The only **interdomain** routing protocol is **BGP** (**Border Gateway Protocol**) → out of the scope of this course

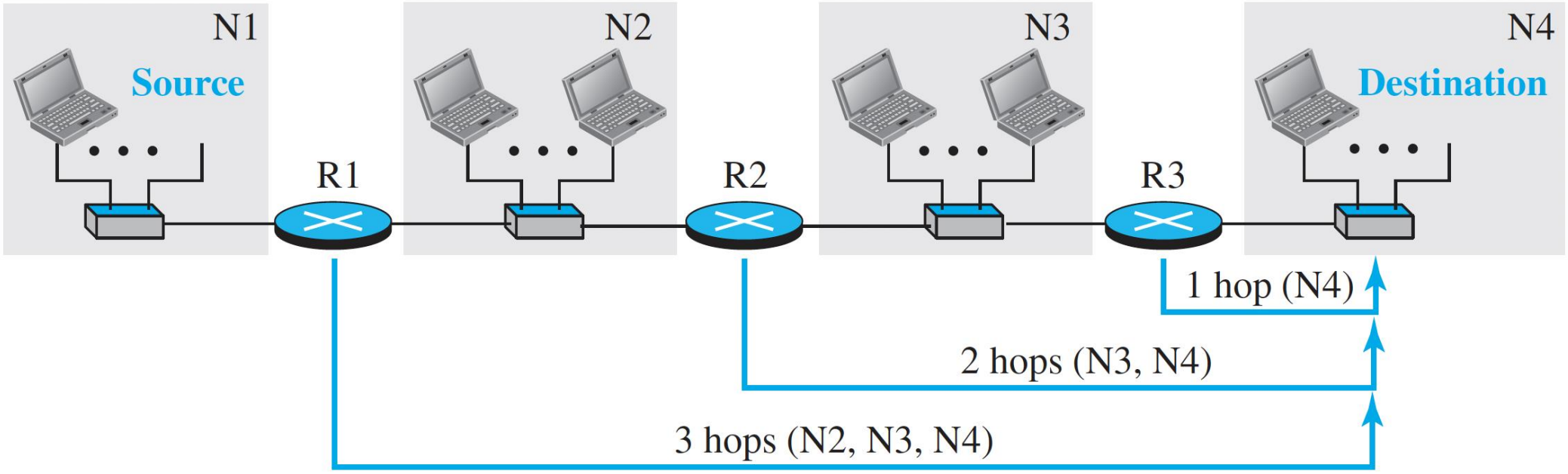
Hierarchical Routing – Intra and Inter ASs



Routing Information Protocol (RIP)

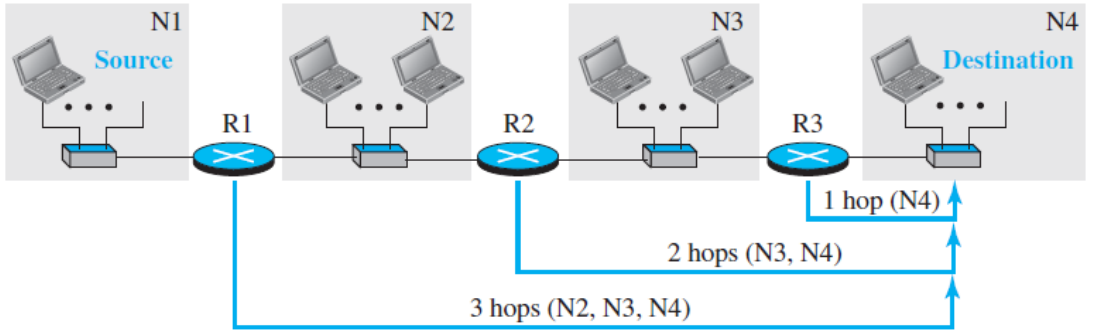
- Based on the **distance-vector routing algorithm**.
- RIP routers **advertise the cost of reaching different networks** instead of reaching other nodes in a theoretical graph.
- Normally used in **small ASs**.
- The **cost** is defined as the **number of hops**.
  - The **number of hops** is the number of networks (subnets) a packet needs to travel through from the **source router** to the **final destination host**.

Hop Counts in RIP



- In RIP, the **maximum cost** of a path can be **15**, which means **16** is considered as **infinity** (no connection). For this reason, RIP can be used only in ASs in which the diameter of the AS is **not more than 15 hops**.

RIP and Forwarding Tables



Forwarding table for R1

Destination network	Next router	Cost in hops
N1	——	1
N2	——	1
N3	R2	2
N4	R2	3

Forwarding table for R2

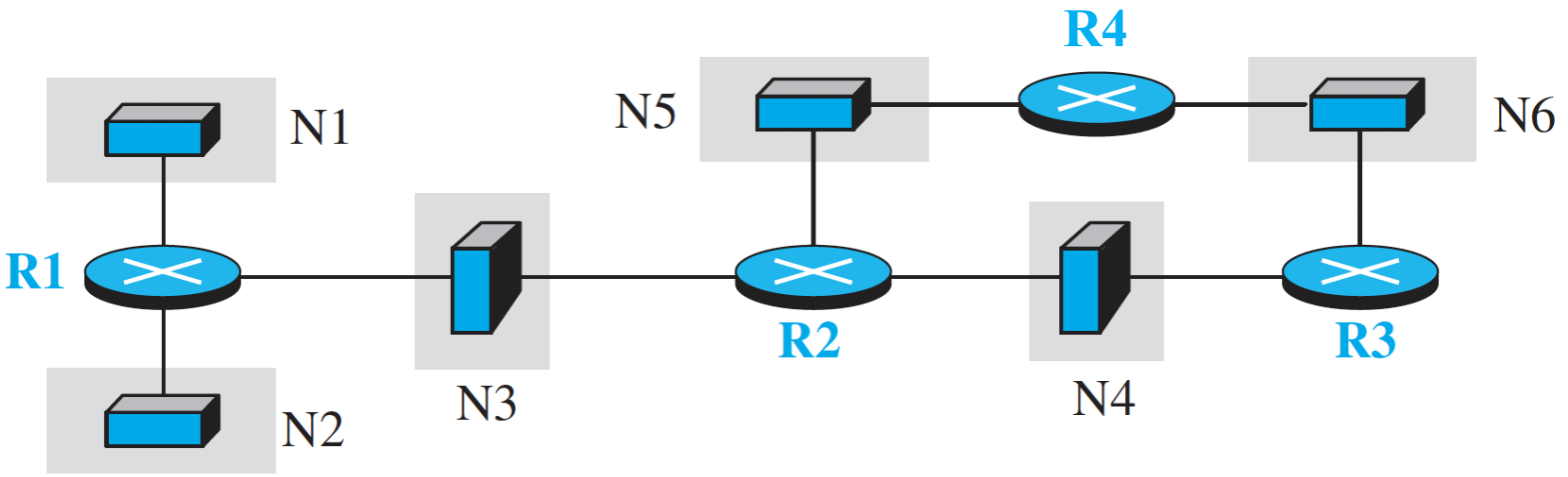
Destination network	Next router	Cost in hops
N1	R1	2
N2	——	1
N3	——	1
N4	R3	2

Forwarding table for R3

Destination network	Next router	Cost in hops
N1	R2	3
N2	R2	2
N3	——	1
N4	——	1

- The third column is not needed for forwarding the packet, but it is needed for updating the **forwarding table** when there is a change in the route.
- Although a forwarding table in RIP defines only the next router in the second column, it gives the information about the **whole least-cost tree**. For example, R1 defines that the next router for the path to N4 is R2; R2 defines that the next router to N4 is R3; R3 defines that there is no next router for this path. The tree is then R1 → R2 → R3 → N4.

Example of an Autonomous System Using RIP



Legend

Des.: Destination network
N. R.: Next router
Cost: Cost in hops
: New route
: Old route

R1

Des.	N. R.	Cost
N1	—	1
N2	—	1
N3	—	1

R2

Des.	N. R.	Cost
N3	—	1
N4	—	1
N5	—	1

R3

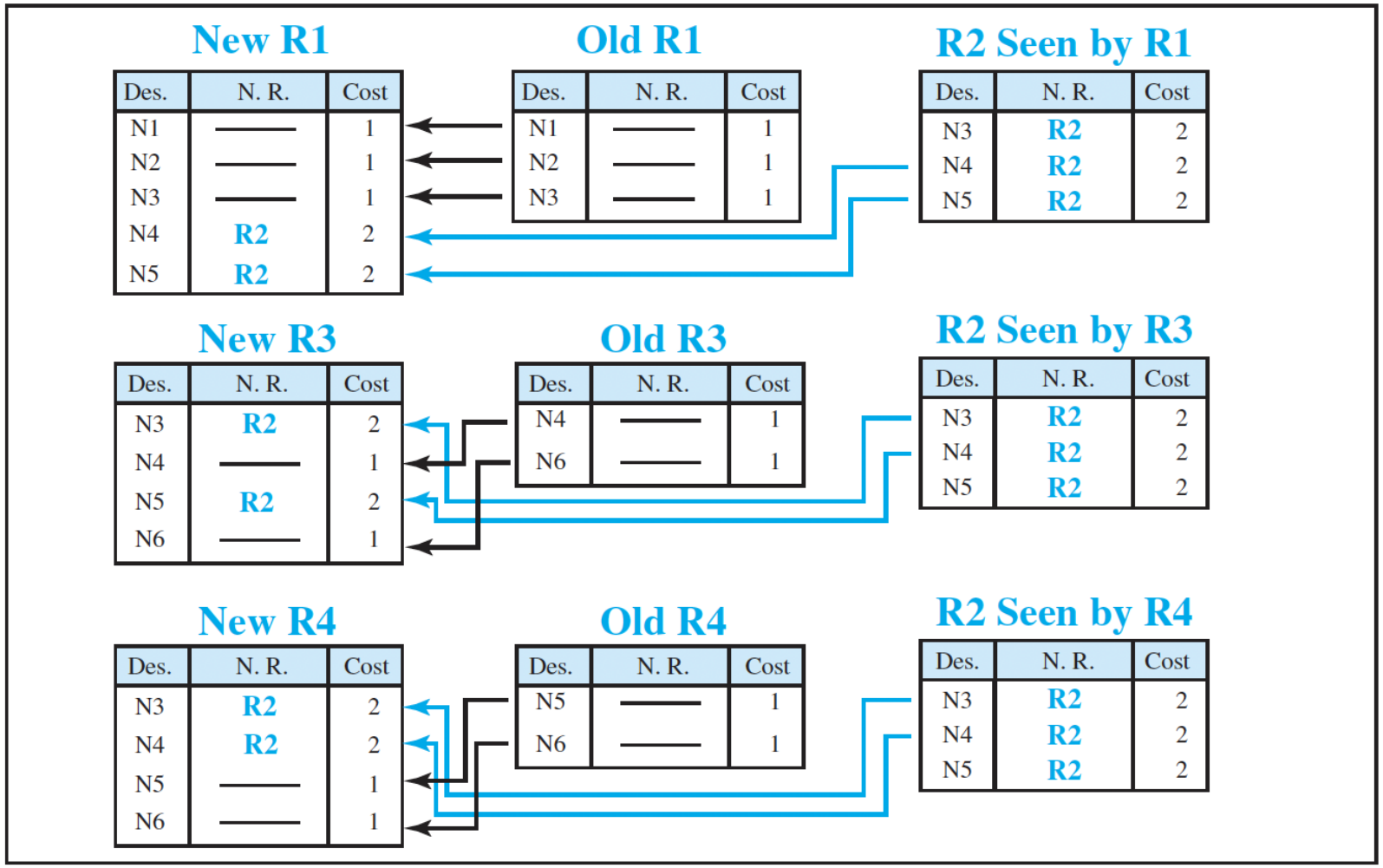
Des.	N. R.	Cost
N4	—	1
N6	—	1

R4

Des.	N. R.	Cost
N5	—	1
N6	—	1

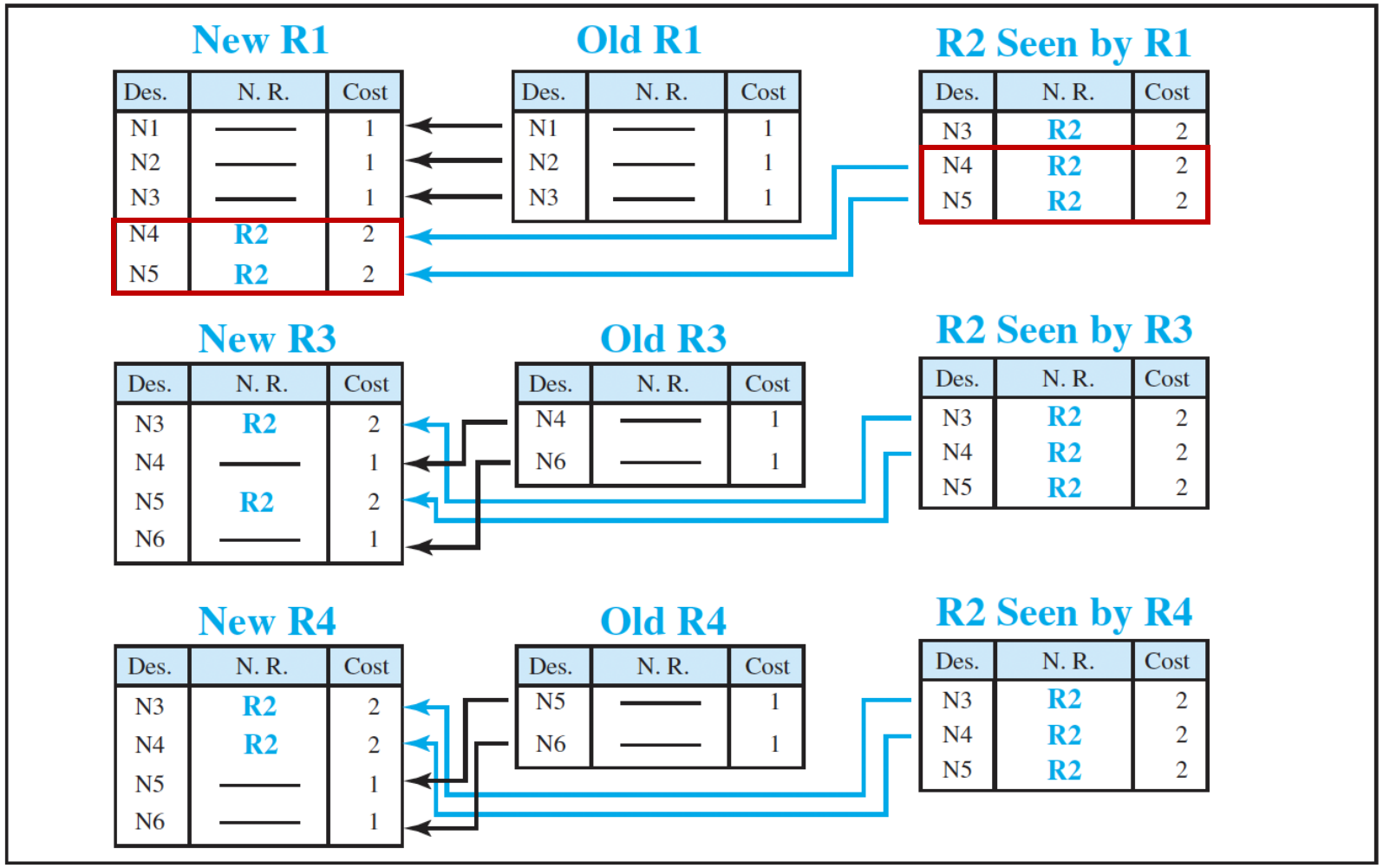
Forwarding tables after all routers booted

Example of an Autonomous System Using RIP



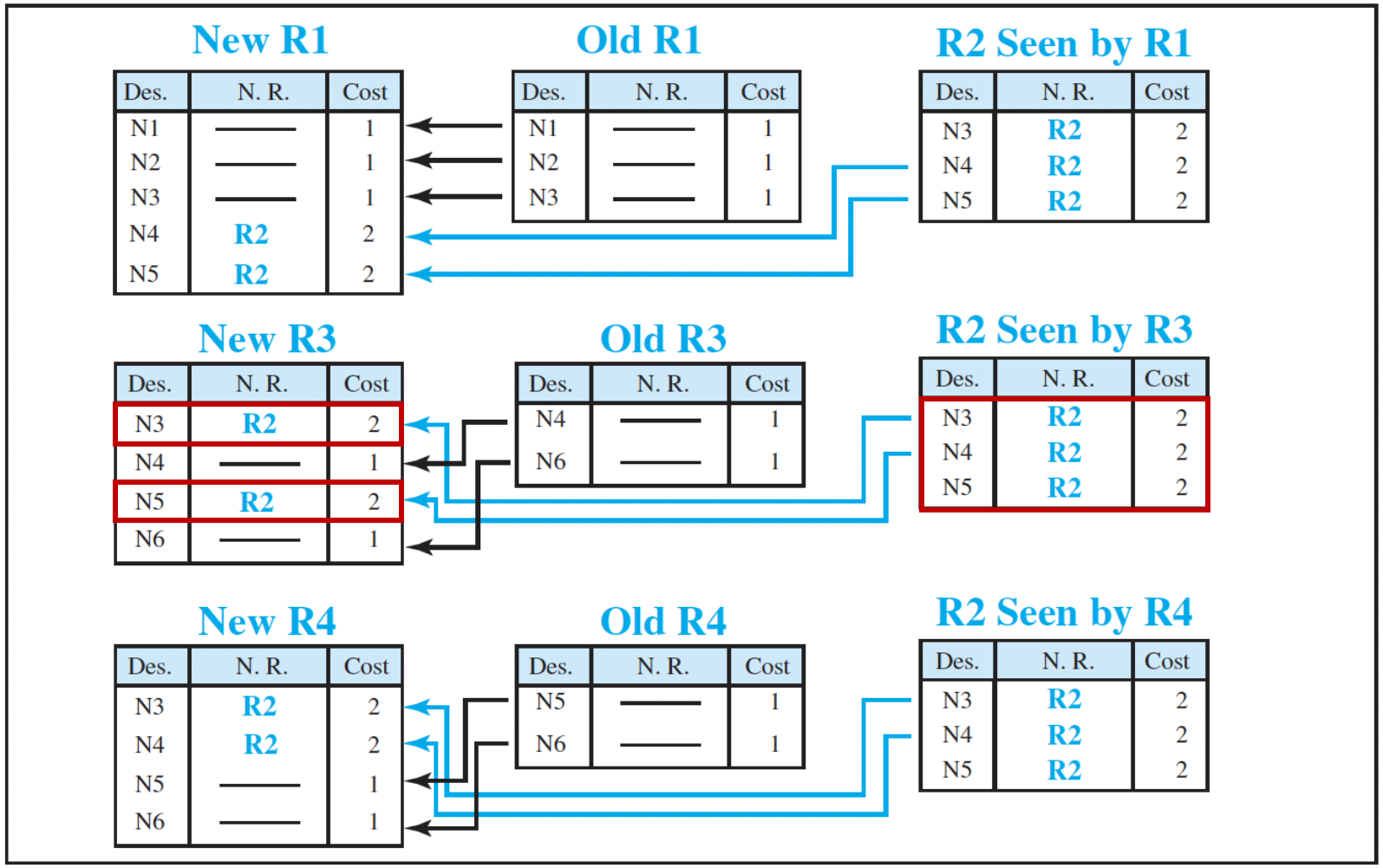
Changes in the forwarding tables of R1, R3, and R4 after they receive a copy of R2's table

Example of an Autonomous System Using RIP



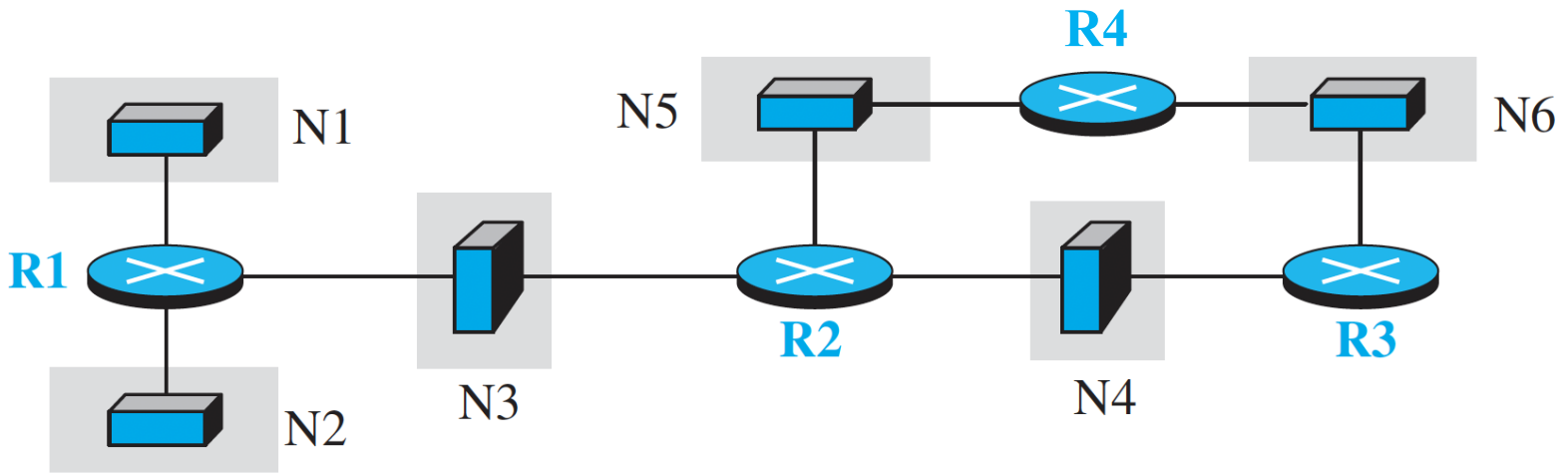
Changes in the forwarding tables of R1, R3, and R4 after they receive a copy of R2's table

Example of an Autonomous System Using RIP



Changes in the forwarding tables of R1, R3, and R4 after they receive a copy of R2's table

Example of an Autonomous System Using RIP



Legend

Des.: Destination network
N. R.: Next router
Cost: Cost in hops
: New route
: Old route

Final R1

Des.	N. R.	Cost
N1	———	1
N2	———	1
N3	———	1
N4	R2	2
N5	R2	2
N6	R2	3

Final R2

Des.	N. R.	Cost
N1	R1	2
N2	R1	2
N3	———	1
N4	———	1
N5	———	1
N6	R3	2

Final R3

Des.	N. R.	Cost
N1	R2	3
N2	R2	3
N3	R2	2
N4	———	1
N5	R2	2
N6	———	1

Final R4

Des.	N. R.	Cost
N1	R2	3
N2	R2	3
N3	R2	2
N4	R2	2
N5	———	1
N6	———	1

Forwarding tables for all routers after they have been stablized

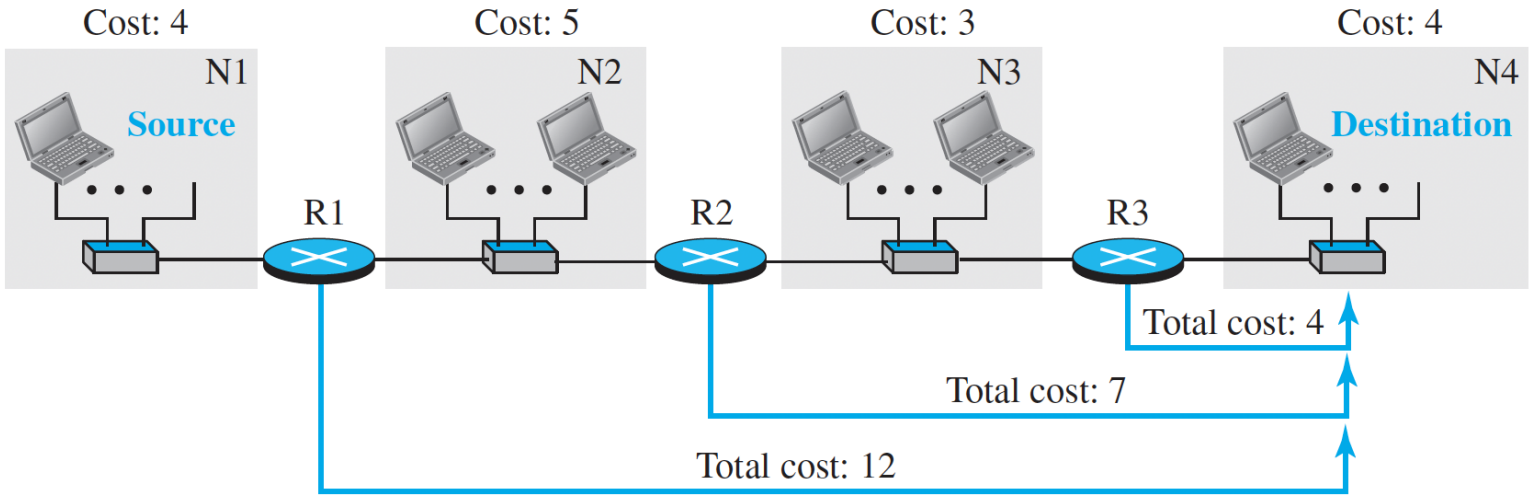
Open Shortest Path First (OSPF)

- Based on the **link-state routing algorithm**.
- The cost of reaching a destination from the host is calculated from the source router to the destination network
- **Each link** (network) can be assigned a **weight** based on the throughput, round-trip time, reliability, and so on.
  - **OSPF does not mandate a policy** for how **link weights** are set (that is the job of the network administrator).
  - Administration can also decide to use the hop count as the cost.
- Used to handle routing in **small or large ASs**.

Open Shortest Path First (OSPF)

- A router **broadcasts routing information** to all other routers in the AS, not just to its neighboring routers.
  - When there is a change in a link's state (e.g., a change in cost or a change in up/down status).
  - Periodically (at least once every 30 minutes).
- The OSPF protocol also checks that links are operational (via a HELLO message that is sent to an attached neighbor).

OSPF – Forwarding Tables



Forwarding table for R1

Destination network	Next router	Cost
N1	——	4
N2	——	5
N3	R2	8
N4	R2	12

Forwarding table for R2

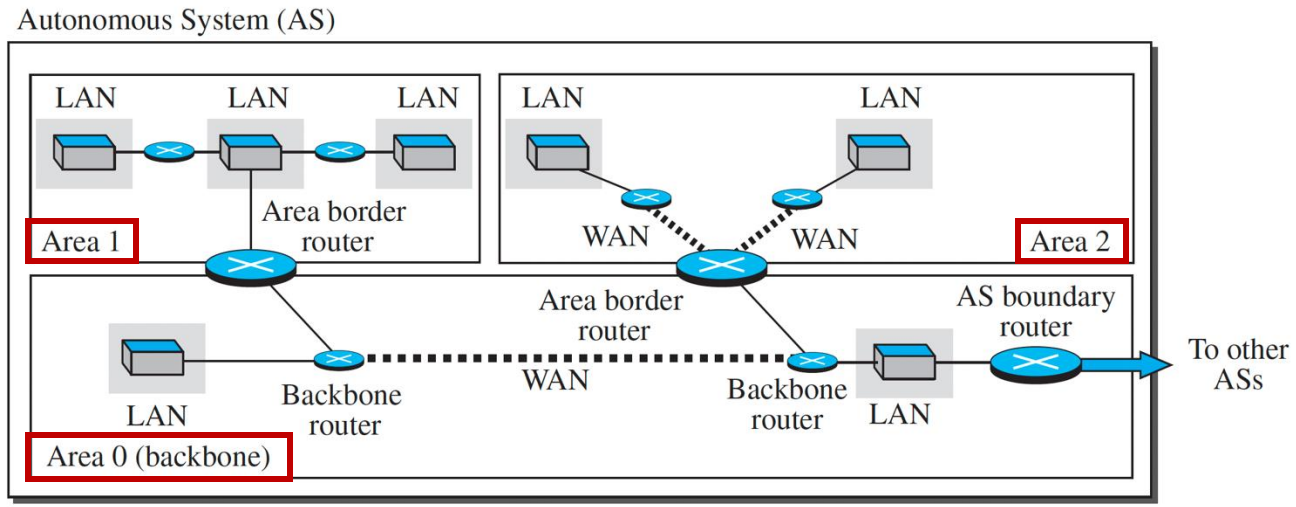
Destination network	Next router	Cost
N1	R1	9
N2	——	5
N3	——	3
N4	R3	7

Forwarding table for R3

Destination network	Next router	Cost
N1	R2	12
N2	R2	8
N3	——	3
N4	——	4

OSPF – Areas

- The AS is divided into small sections called **areas**.
  - Each area acts as a small independent domain for flooding LSPs (LS Packets).
- Why using areas?
  - OSPF requires that all routers flood the whole AS with their LSPs to create the global LSDB → It may create a huge volume of traffic in a large AS.



Summary

- Although there are several routes that a packet can travel from the source to the destination, the question is which should be the **best**.
- The interpretation of the term best depends on the **cost and policy** imposed on the trip and routing algorithms help to find the best route.
- Two most commonly used **intra-domain routing protocols**:
  - **RIP**: base on the distance-vector (DV) routing algorithm.
  - **OSPF**: based on link-state (LS) routing algorithms
- An inter-domain routing protocol: **BGP**.

References

[1] Behrouz A.Forouzan, Data Communications & Networking with TCP/IP Protocol Suite, 6th Ed, 2022, McGraw-Hill companies.

[2] J.F. Kurose, K.W. Ross, Computer Networking: A Top-Down Approach, 7th Ed, 2017, Pearson Education, Inc.

Reading

- Chapter 8 of the textbook, sections 8.1, 8.2 (up to 8.2.3), and 8.3 (up to 8.3.4)
- Chapter 8 of the textbook, section 8.7 (Practice Test)