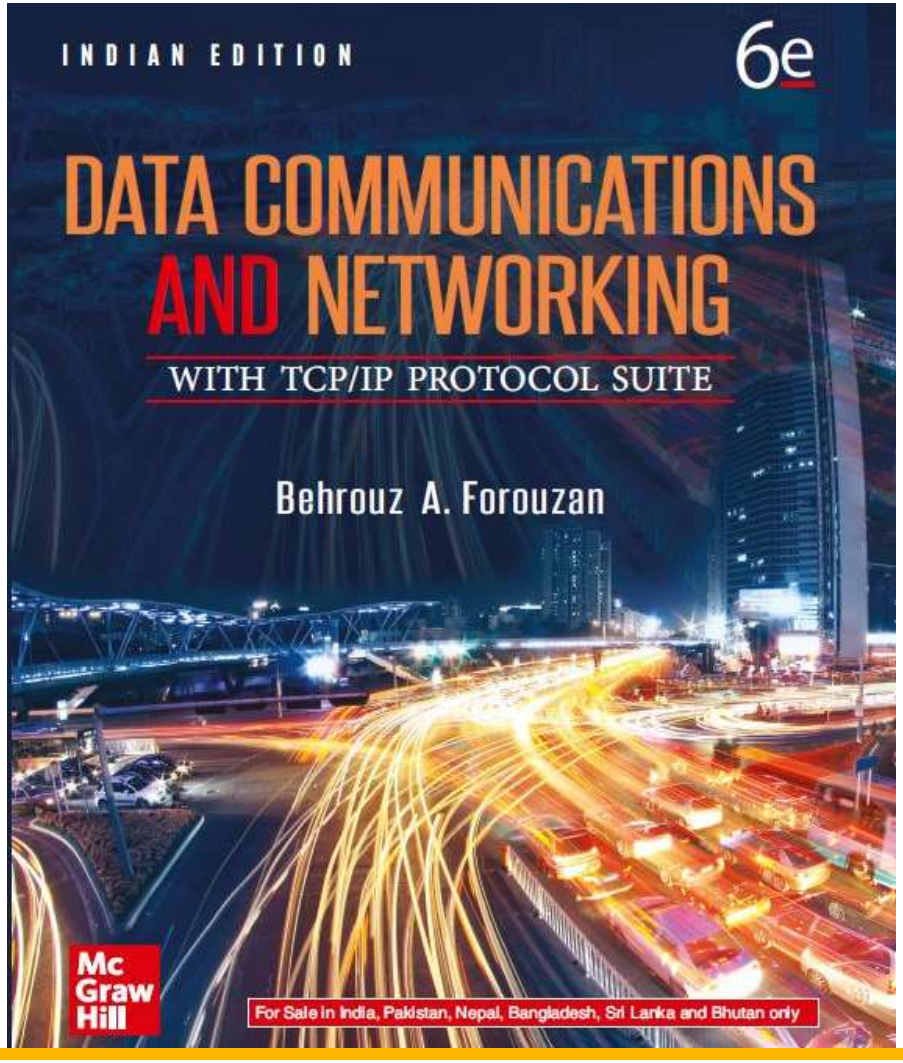
Because learning changes everything.®

Chapter 08

Network Layer: Routing of Packets

Data Communications and Networking, With TCP/IP protocol suite
Sixth Edition
Behrouz A. Forouzan



Chapter 8: Outline

8.1 Introduction

8.2 Routing Algorithms

8.3 Unicast Routing Protocols

8.4 Multicast Routing

8-1 INTRODUCTION

Unicast routing in the Internet, with a large number of routers and a huge number of hosts, can be done only by using hierarchical routing: routing in several steps using different routing algorithms. In this section, we first discuss the general concept of unicast routing in an internet. After the routing concepts and algorithms are understood, we show how we can apply them to the Internet.

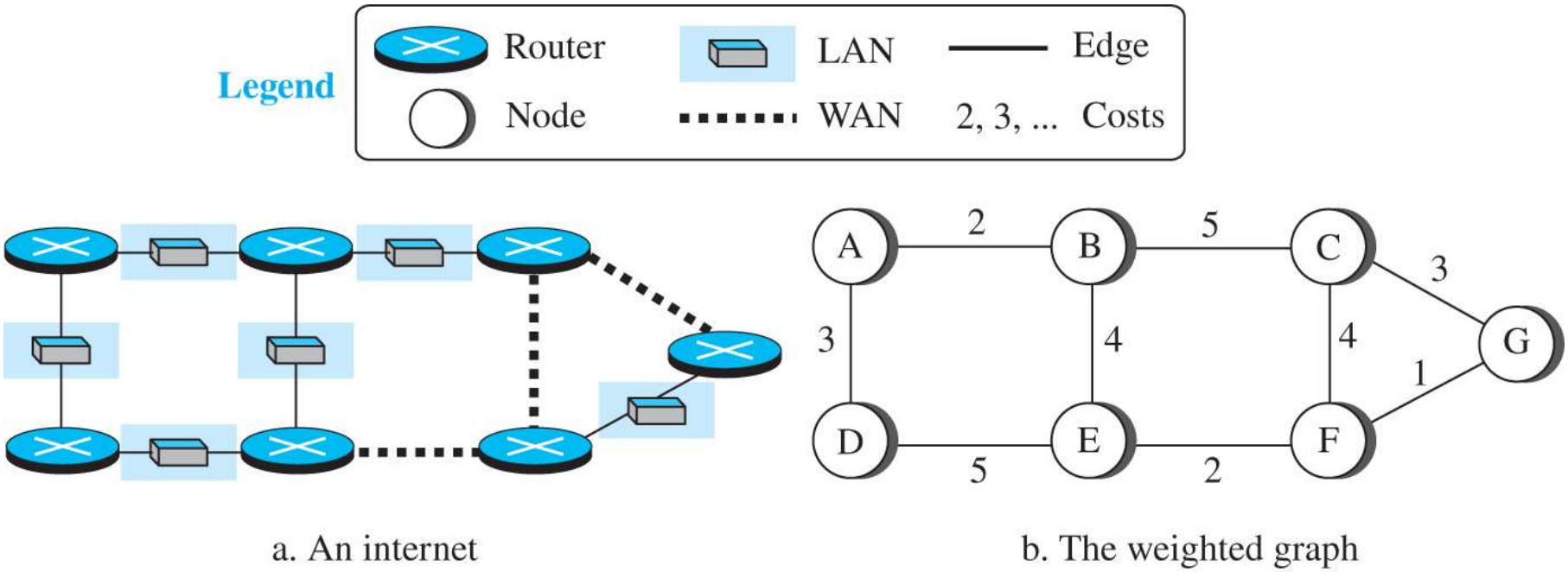
8.1.1 General Idea

In unicast routing, a packet is routed, hop by hop, from its source to its destination by the help of forwarding tables. 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 either because it receives the packet from its default router in its local network. This means that only the routers that glue together the networks in the internet need forwarding tables.

An Internet as a Graph

To find the best route, an internet can be modeled as a graph. A graph in computer science is a set of nodes and edges (lines) that connect the nodes. To model an internet as a graph, we can think of each router as a node and each network between a pair of routers as an edge. An internet is, in fact, modeled as a weighted graph, in which each edge is associated with a cost.

Figure 8.1 An internet and its graphical representation



a. An internet

b. The weighted graph

Access the text alternative for slide images.

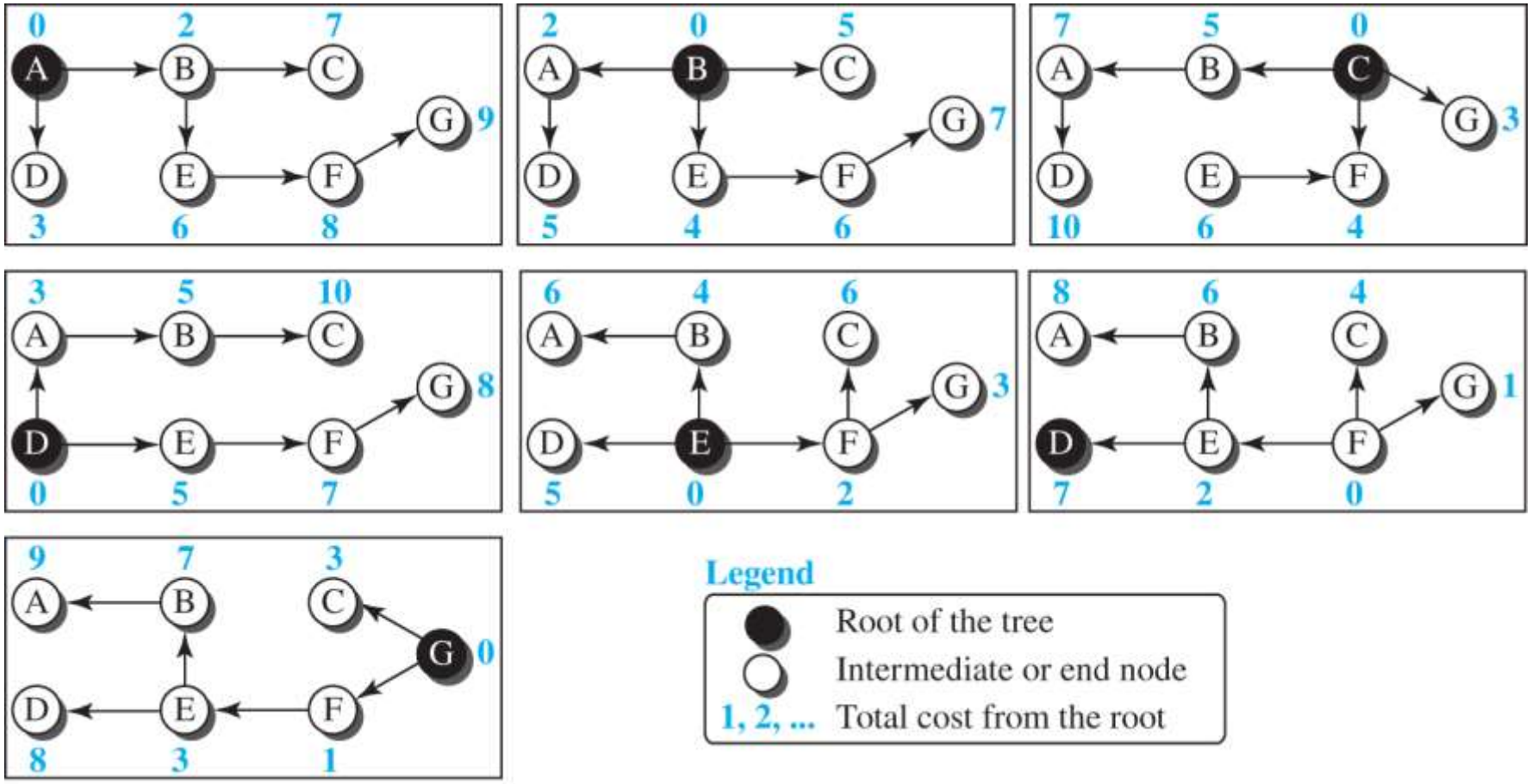
8.1.2 Least-Cost Routing

When an internet is modeled as a weighted graph, one of the ways to interpret the best route from the source router to the destination router is to find the least cost between the two. In other words, the source router chooses a route to the destination router in such a way that the total cost for the route is the least cost among all possible routes.

Least Cost Trees

*If there are N routers in an internet, there are (N - 1) least-cost paths from each router to any other router. This means we need N * (N - 1) least-cost paths for the whole internet. If we have only 10 routers in an internet, we need 90 least-cost paths. A better way to see all of these paths is to combine them in a least-cost tree. A least-cost tree is a tree with the source router as the root that spans the whole graph (visits all other nodes) and in which the path between the root and any other node is the shortest. Figure 8.2 shows the seven least-cost trees for the internet in Figure 8.1.*

Figure 8.2 Least-cost trees for nodes in the internet of Figure 4.56



Access the text alternative for slide images.

8-2 ROUTING ALGORITHMS

Several routing algorithms have been designed in the past. The differences between these methods are in the way they interpret the least cost and the way they create the least-cost tree for each node. In this section, we discuss the common algorithms; later we show how a routing protocol in the Internet implements one of these algorithms.

8.2.1 Distance-Vector Routing

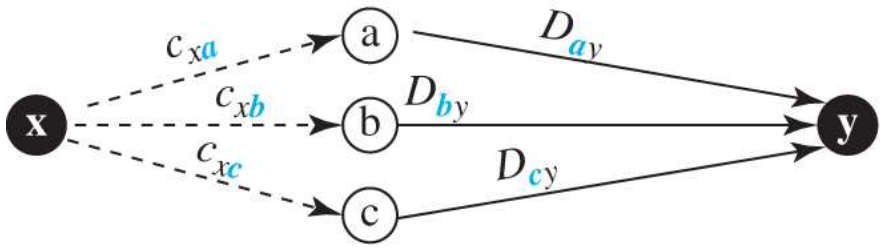
The distance-vector (DV) routing uses the goal we discussed in the introduction, to find the best route. In distance-vector routing, the first thing each node creates is its own least-cost tree with the rudimentary information it has about its immediate neighbors. The incomplete trees are exchanged between immediate neighbors to make the trees more and more complete and to represent the whole internet. We can say that in distance-vector routing, a router continuously tells all of its neighbors what it knows about the whole internet (although the knowledge can be incomplete).

Bellman-Ford Equation

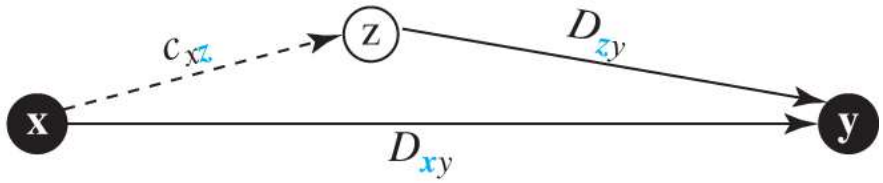
The heart of distance-vector routing is the famous Bellman-Ford equation. This equation is used to find the least cost (shortest distance) between a source node, x, and a destination node, y, through some intermediary nodes (a, b, c, . . .) when the costs between the source and the intermediary nodes and the least costs between the intermediary nodes and the destination are given.

$$Dxy = min\{(cxa + Day), (cxb + Dby), (cxc + Dcy), \ldots\}$$

Figure 8.3 Graphical idea behind Bellman-Ford equation



a. General case with three intermediate nodes



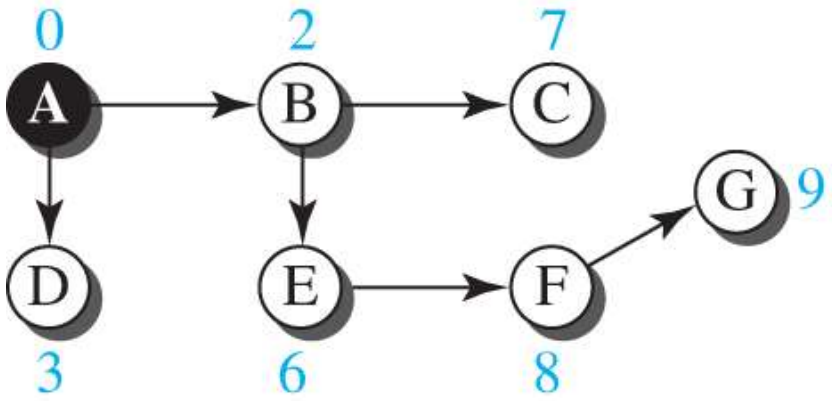
b. Updating a path with a new route

Access the text alternative for slide images.

Distance Vector

The concept of a distance vector is the rationale for the name distance-vector routing. A least-cost tree is a combination of least-cost paths from the root of the tree to all destinations. These paths are graphically glued together to form the tree. Distance-vector routing unglues these paths and creates a distance vector, a one-dimensional array to represent the tree. Figure 8.4 shows the tree for node A in the internet in Figure 8.1 and the corresponding distance vector.

Figure 8.4 The distance vector corresponding to a tree



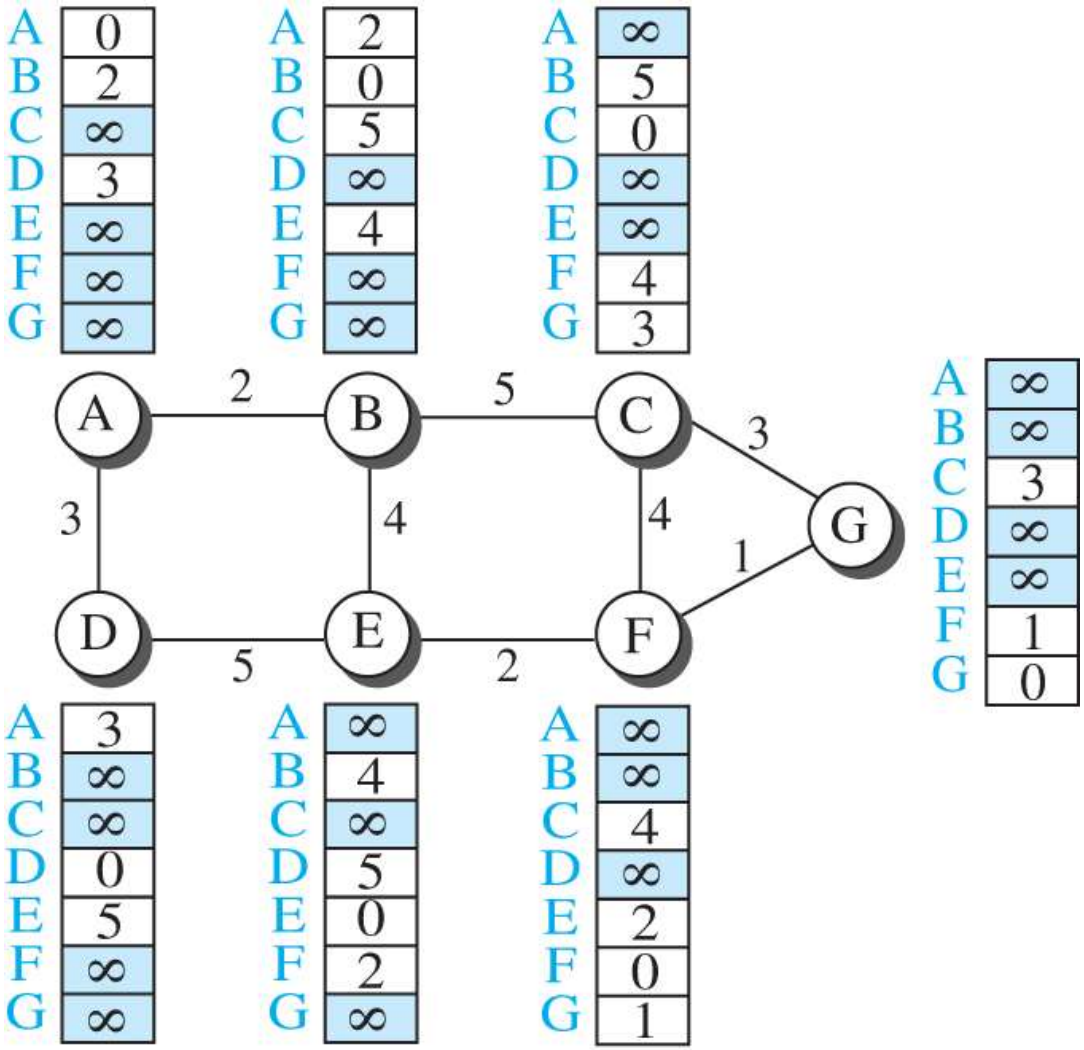
a. Tree for node A

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

b. Distance vector for node A

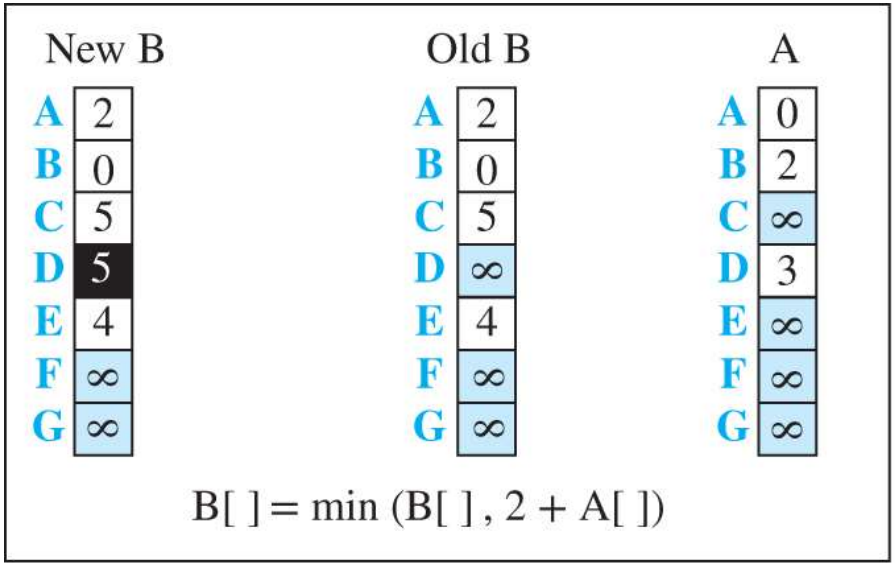
Access the text alternative for slide images.

Figure 8.5 The first distance vector for an internet

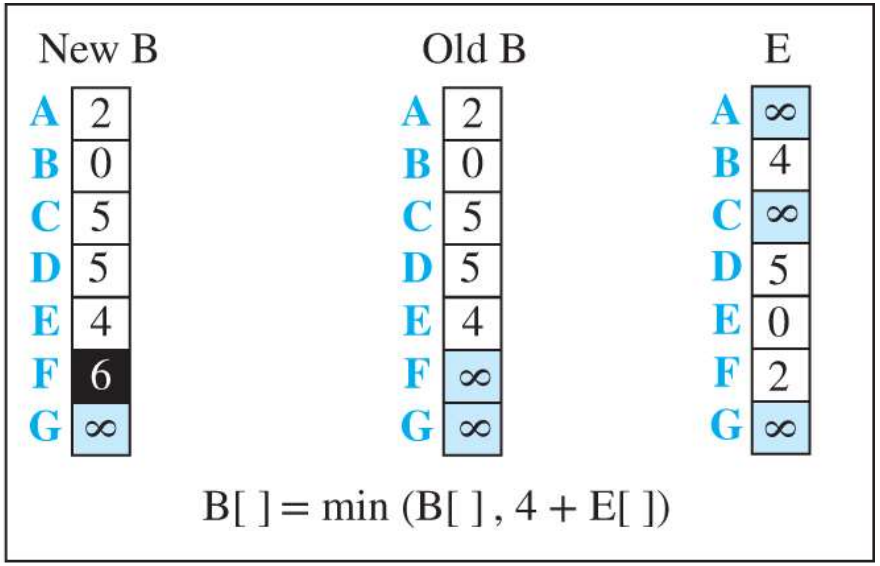


Access the text alternative for slide images.

Figure 8.6 Updating distance vectors



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



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

Note:
X[ ]: the whole vector

Access the text alternative for slide images.

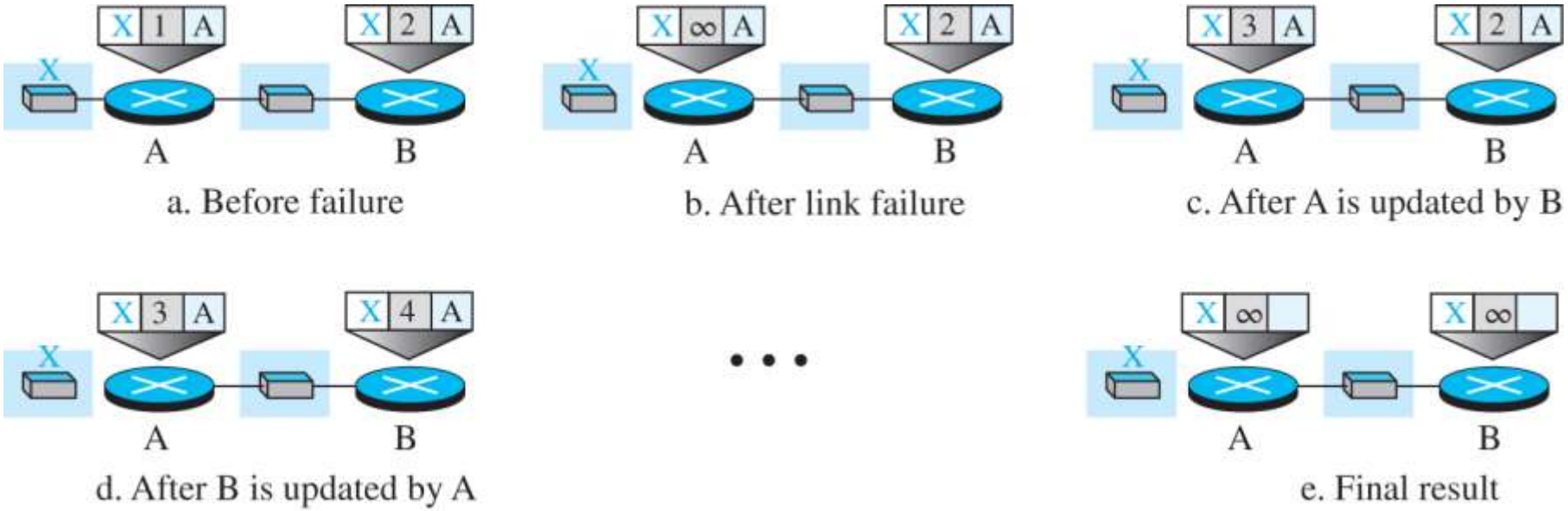
Distance-Vector Routing Algorithm

Now we can give a simplified pseudocode for the distance-vector routing algorithm, as shown in Table 8.1. The algorithm is run by its node independently and asynchronously.

Table 8.1 Distance-vector routing algorithm for a node

```
1   Distance_Vector_Routing ( )
2   {
3       // Initialize (create initial vectors for the node)
4       D[myself ] = 0
5       for (y = 1 to N)
6       {
7           if (y is a neighbor)
8               D[y] = c[myself][y]
9           Else
10              D[y] = ∞
11      }
12      send vector {D[1], D[2], …, D[N]} to all neighbors
13      // Update (improve the vector with the vector received from a neighbor)
14      repeat (forever)
15      {
16          wait (for a vector Dw from a neighbor w or any change in the link)
17          for (y = 1 to N)
18          {
19              D[y] = min [D[y], (c[myself ][w] + Dw[y])] // Bellman-Ford equation
20          }
21          if (any change in the vector)
22              send vector {D[1], D[2], …, D[N]} to all neighbors
23      }
24  }
```

Figure 8.7 Two-node instability



Access the text alternative for slide images.

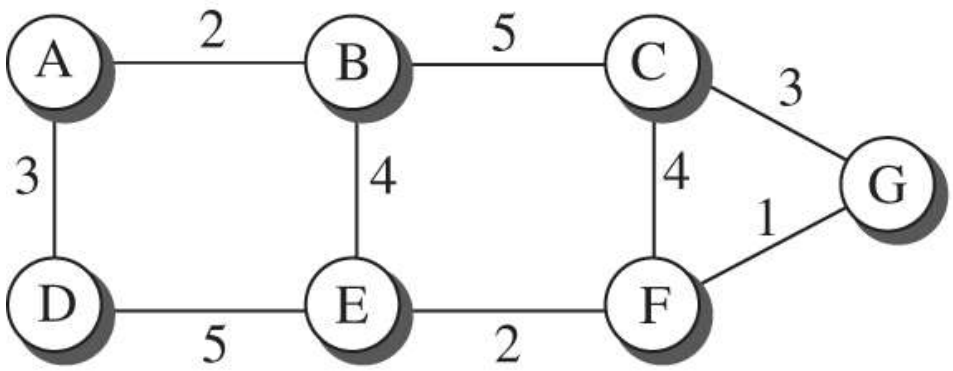
8.2.2 Link-State Routing

A routing algorithm that directly follows our discussion for creating least-cost trees and forwarding tables is link-state (LS) routing. This method uses the term link-state to define the characteristic of a link (an edge) that represents a network in the internet. In this algorithm 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.

Link-State Database (LSDB)

To create a least-cost tree with this method, each node needs to have a complete map of the network, which means it needs to know the state of each link. The collection of states for all links is called the 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. Figure 8.8 shows an example of an LSDB for the graph in Figure 8.1. The LSDB can be represented as a two-dimensional array (matrix) in which the value of each cell defines the cost of the corresponding link.

Figure 8.8 Example of a link-state database



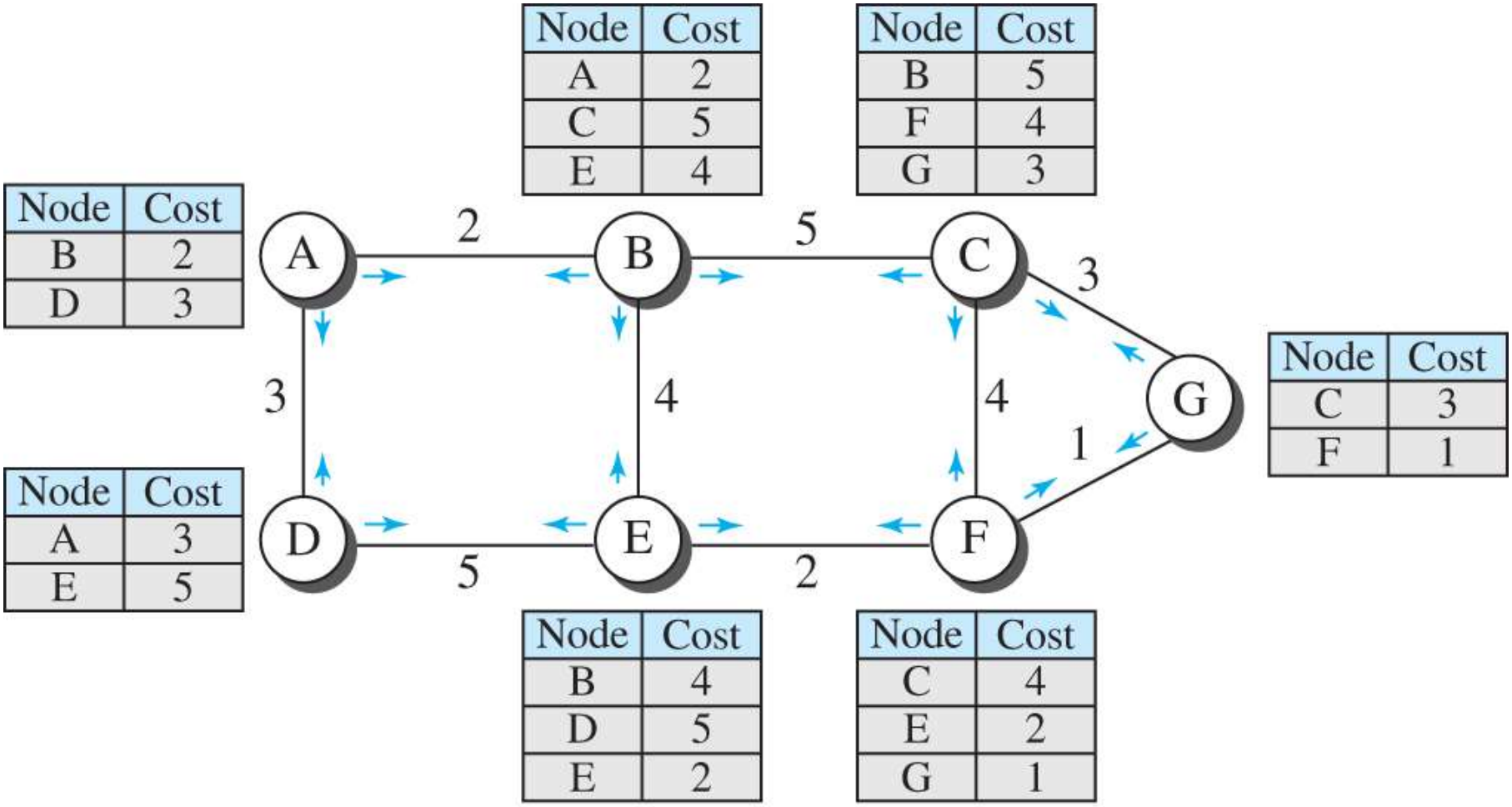
a. The weighted graph

	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

Access the text alternative for slide images.

Figure 8.9 LSPs created and sent out by each node to build LSDB



Access the text alternative for slide images.

Formation of Least-Cost Tree

To create a least-cost tree for itself, using the shared LSDB, each node needs to run the famous Dijkstra Algorithm. This iterative algorithm uses the following steps:

1. *The node chooses itself as the root of the tree, creating a tree with a single node, and sets the total cost of each node based on the information in the LSDB.*

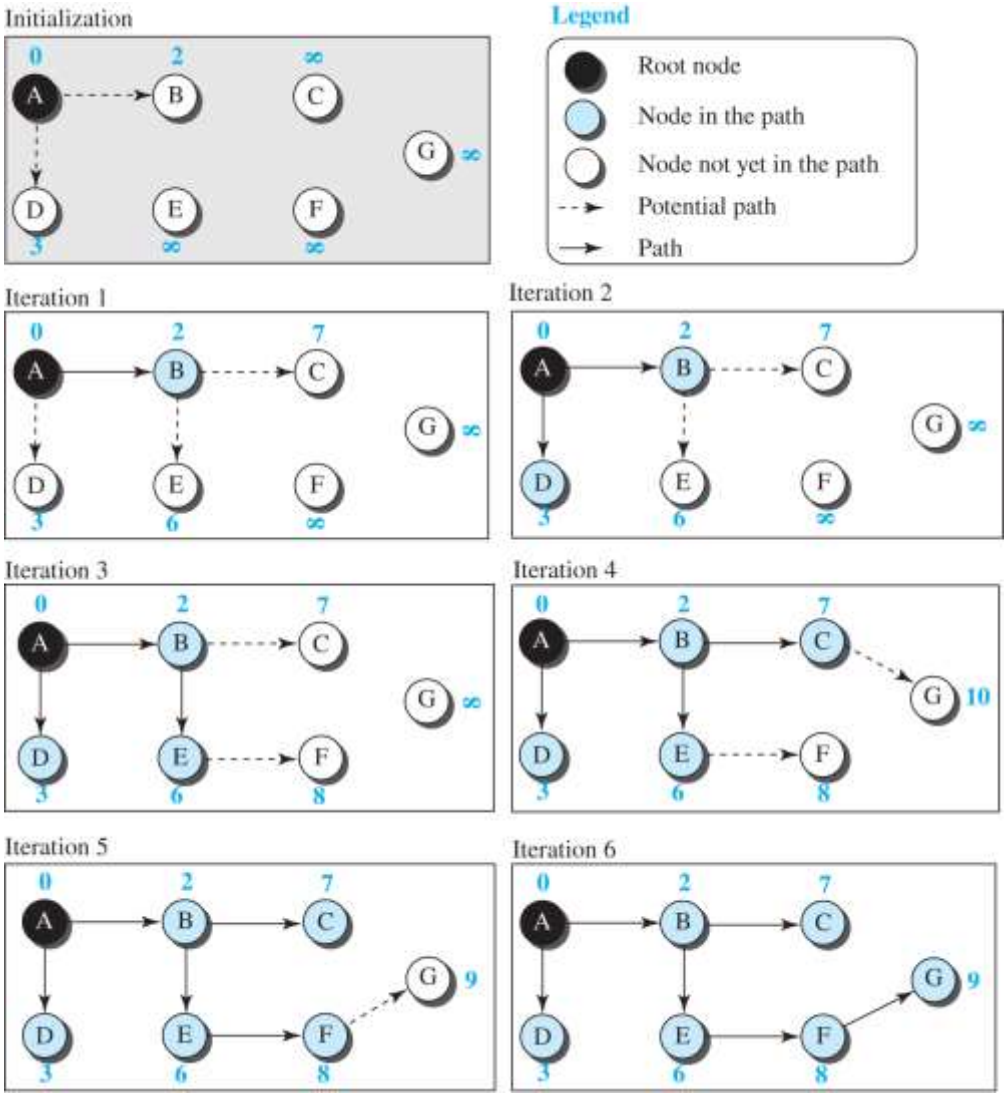
2. *The node selects one node, among all nodes not in the tree, which is closest to the root, and adds this to the tree.*

3. *The node repeats step 2 until all nodes are added to the tree.*

Table 8.2 Dijkstra's algorithm

```
1   Distance_Vector_Routing ( )
2   {
3       // Initialization
4       Tree = {root}        // Tree is made only of the root
5       for (y = 1 to N)     // N is the number of nodes
6       {
7           if (y is the root)
8               D [y] = 0       // D [y] is shortest distance from root to node y
9           else if (y is a neighbor)
10              D [y] = c[root][y]      // c [x] [y] is cost between nodes x and y in LSDB
11          else
12              D [y] = ∞
13      }
14      // Calculation
15      repeat
16      {
17          find a node w, with D [w ] minimum among all nodes not in the Tree
18          Tree = Tree ∪ {w}       // Add w to tree
19          // Update distances for all neighbor of w
20          for (every node x, which is neighbor of w and not in the Tree)
21          {
22              D[x] = min{D[x], (D[w] + c[w][x])}
23          }
24      } until (all nodes included in the Tree)
25  }
```

Figure 8.10 Least-cost tree



Access the text alternative for slide images.

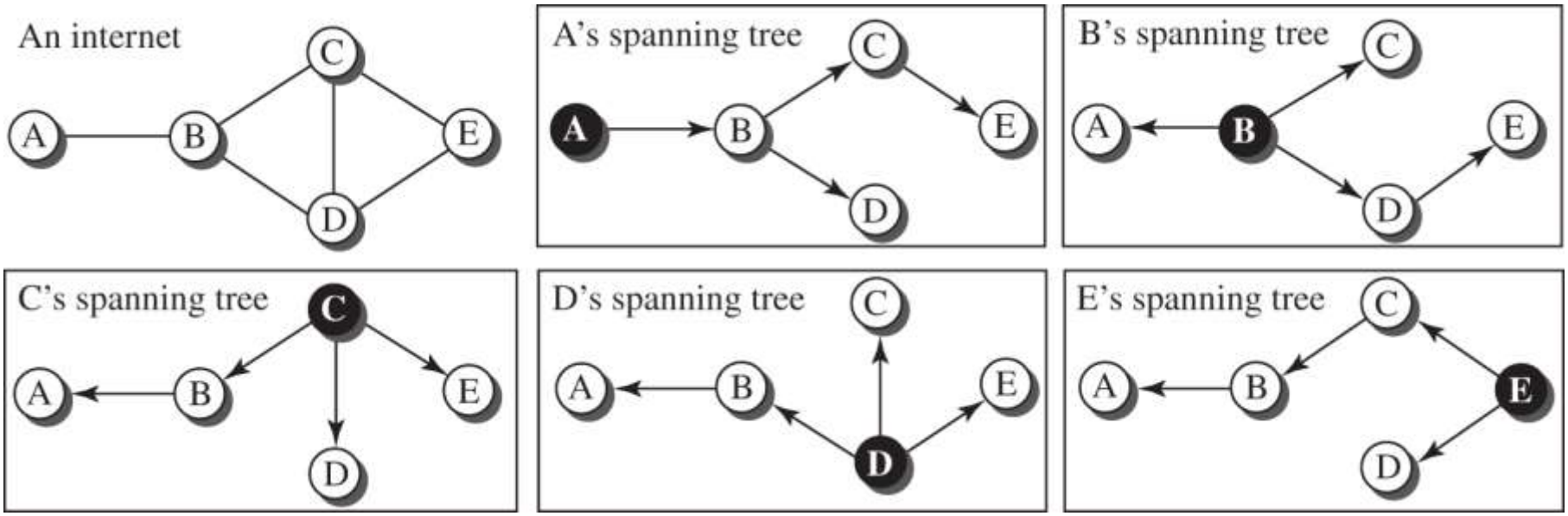
8.2.3 Path-Vector Routing

Both link-state and distance-vector routing are based on the least-cost goal. However, there are instances where this goal is not the priority. For example, assume that there are some routers in the internet that a sender wants to prevent its packets from going through. In other words, the least-cost goal, applied by LS or DV routing, does not allow a sender to apply specific policies to the route a packet may take. To respond to these demands, a third routing algorithm, called path-vector (PV) routing has been devised.

Spanning Tree

In path-vector routing, the path from a source to all destinations is also determined by the best spanning tree. The best spanning tree, however, is not the least-cost tree; it is the tree determined by the source when it imposes its own policy. If there is more than one route to a destination, the source can choose the route that meets its policy best. A source may apply several policies at the same time. One of the common policies uses the minimum number of nodes to be visited (something similar to least-cost). Another common policy is to avoid some nodes as the middle node in a route.

Figure 8.11 Spanning trees in path-vector routing

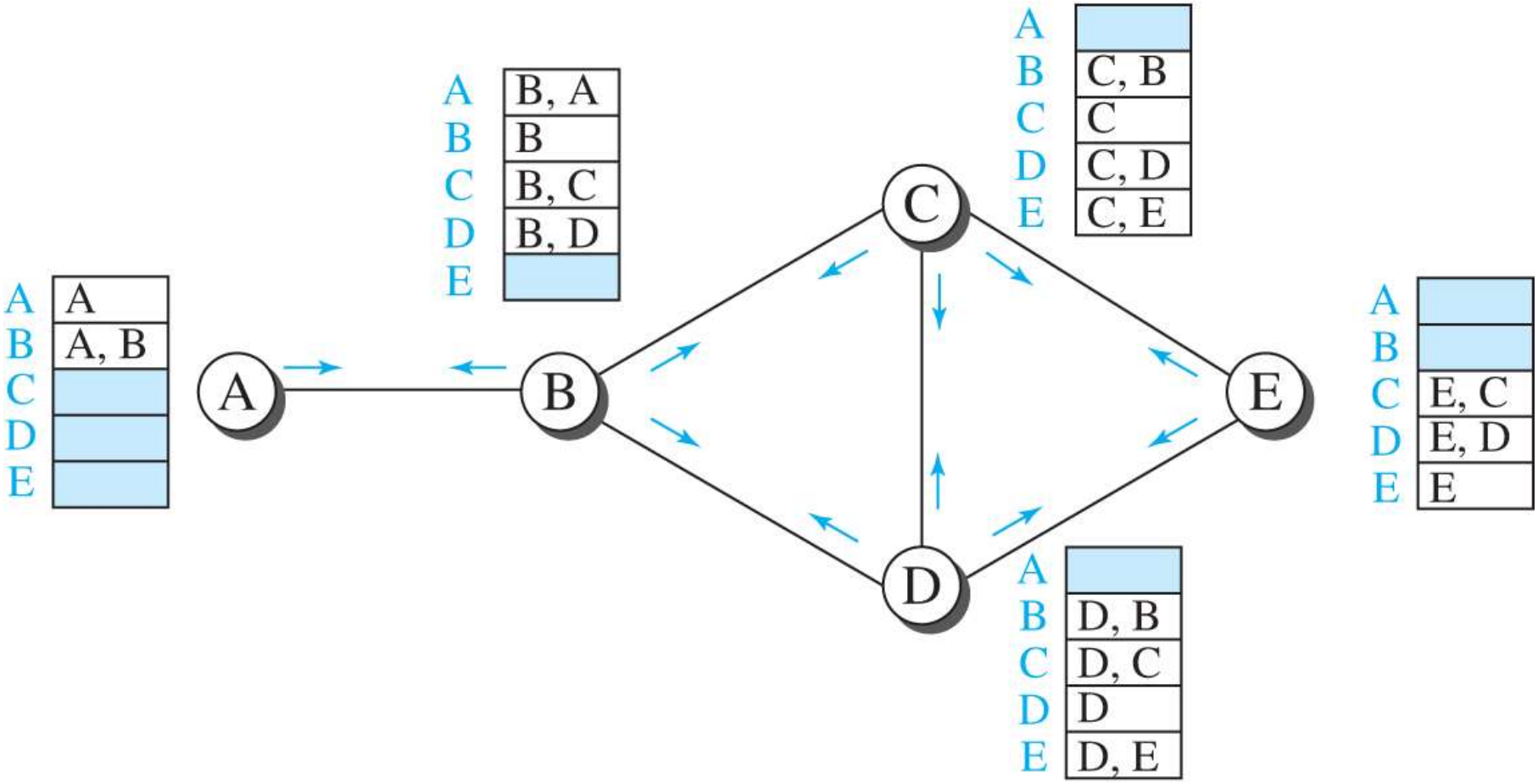


Access the text alternative for slide images.

Creation of Spanning Tree

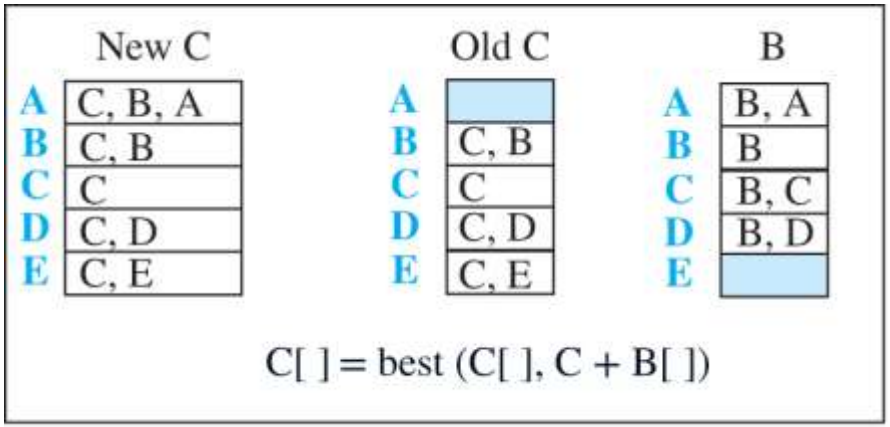
Path-vector routing, like distance-vector routing, is an asynchronous and distributed routing algorithm. The spanning trees are made, gradually and asynchronously, by each node. When a node is booted, it creates a path vector based on the information it can obtain about its immediate neighbor. A node sends greeting messages to its immediate neighbors to collect these pieces of information. Figure 8.12 shows all of these path vectors for our internet in Figure 8.11.

Figure 8.12 Path vectors made at booting time

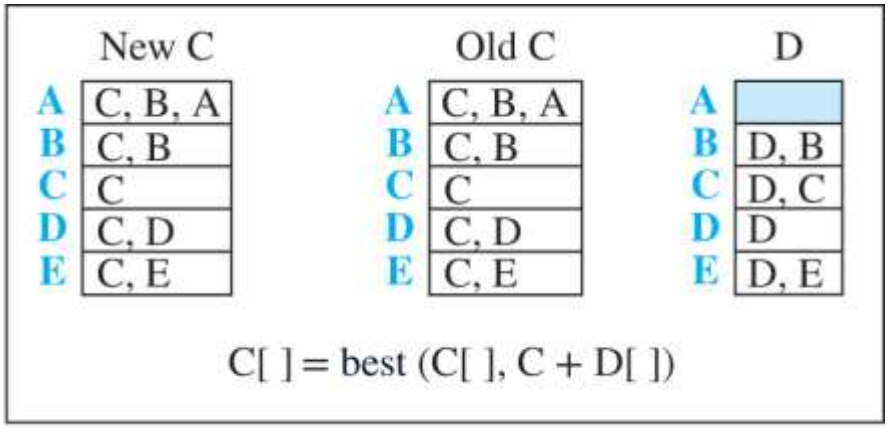


Access the text alternative for slide images.

Figure 8.13 Updating path vectors



Event 1: C receives a copy of B's vector



Event 2: C receives a copy of D's vector

Note:
X [ ]: vector X
Y: node Y

Access the text alternative for slide images.

Table 8.3 Path-vector algorithm for a node

```
1   Table 8.3  Path-vector algorithm for a node
2   {
3       // Initialization
4       for (y = 1 to N)
5       {
6           if ( y is myself)
7               Path[ y] = myself
8           else if (y is a neighbor)
9               Path[ y] = myself + neighbor node
10          else
11              Path[ y] = empty
12      }
13      Send vector {Path[1], Path[2], …, Path[ y]} to all neighbors
14      // Update
15      repeat (forever)
16      {
17          wait (for a vector Path_w from a neighbor w)
18          for ( y = 1 to N)
19      {
20          if (Path_w includes myself)
21              discard the path                    // Avoid any loop
22          else
23              Path[ y] = best {Path[ y], (myself + Path_w[ y])}
24      }
25      If (there is a change in the vector)
26          Send vector {Path[1], Path[2], …, Path[ y]} to all neighbors
27    }
28  } // End of Path Vector
```

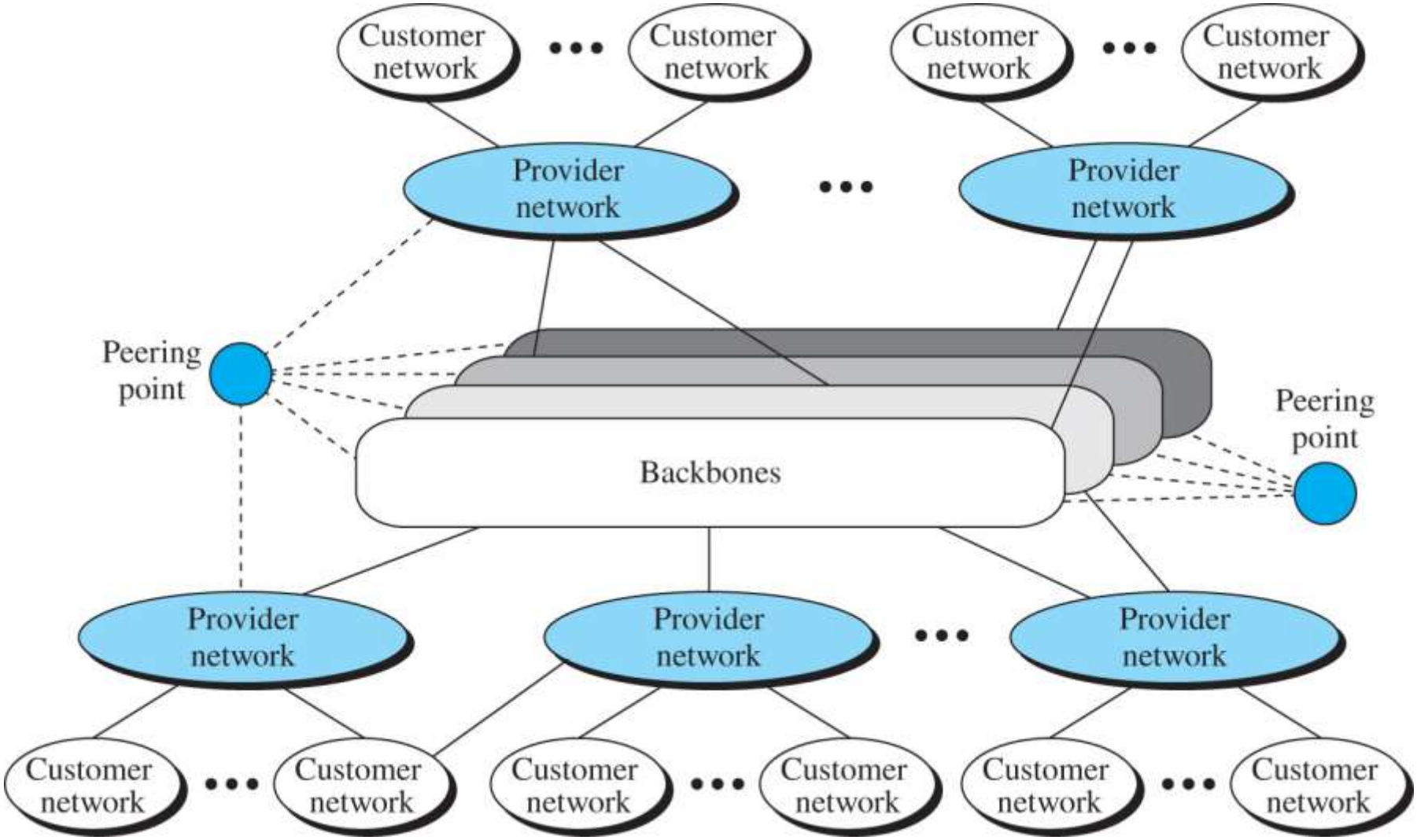
8-3 UNICAST ROUTING PROTOCOLS

After an introduction, we discuss three common protocols used in the Internet: Routing Information Protocol (RIP), based on the distance-vector algorithm; Open Shortest Path First (OSPF), based on the link-state algorithm; and Border Gateway Protocol (BGP), based on the path-vector algorithm.

8.3.1 Internet Structure

Before discussing unicast routing protocols, we need to understand the structure of today's Internet. The Internet has changed from a tree-like structure, with a single backbone, to a multi-backbone structure run by different private corporations today. Although it is difficult to give a general view of the Internet today, we can say that the Internet has a structure similar to what is shown in Figure 8.14.

Figure 8.14 Internet structure



Access the text alternative for slide images.

Hierarchical Routing

The Internet today is made of a huge number of networks and routers that connect them. It is obvious that routing in the Internet cannot be done using one single protocol for two reasons: a scalability problem and an administrative issue. Scalability problem means that 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. The administrative issue is related to the Internet structure described in Figure 8.14. As the figure shows, each ISP is run by an administrative authority.

Autonomous System

As we said before, each ISP is an autonomous system when it comes to managing networks and routers under its control. Although we may have small, medium-size, and large ASs, each AS is given an autonomous number (ASN) by the ICANN. We have stub ASs, multihomed ASs, and transient ASs.

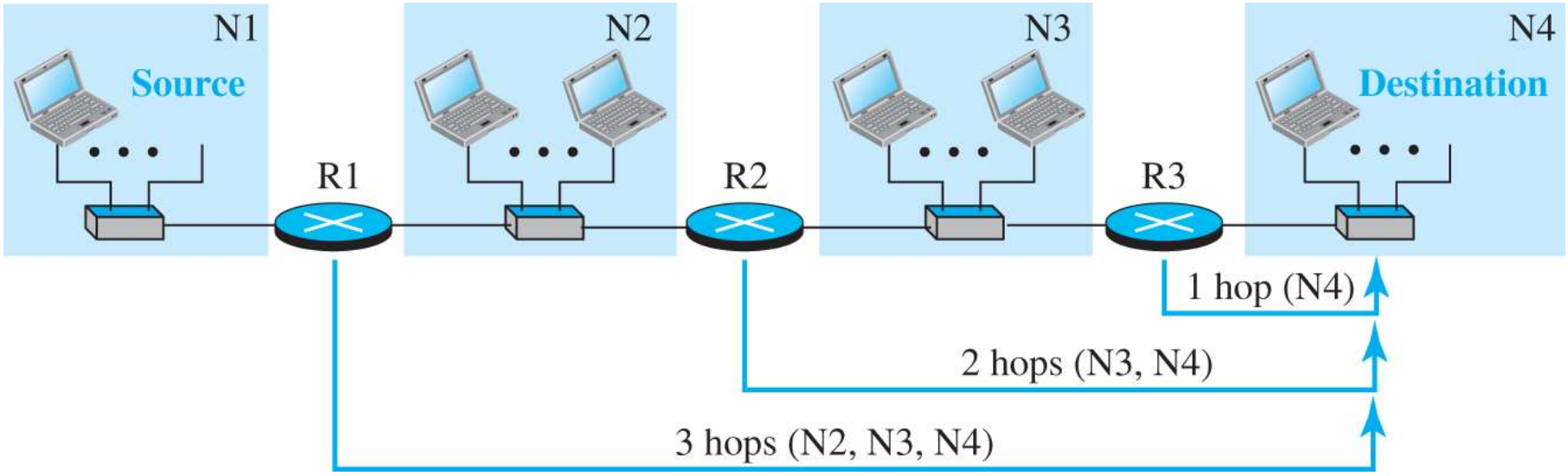
8.3.2 Routing Information Protocol

The Routing Information Protocol (RIP) is one of the most widely used intradomain routing protocols based on the distance-vector routing algorithm we described earlier. RIP was started as part of the Xerox Network System (XNS), but it was the Berkeley Software Distribution (BSD) version of UNIX that helped make the use of RIP widespread.

Hop Count

A router in this protocol basically implements the distance-vector routing algorithm shown in Table 8.1. However, the algorithm has been modified as described below. First, since a router in an AS needs to know how to forward a packet to different networks (subnets) in an AS, RIP routers advertise the cost of reaching different networks instead of reaching other nodes in a theoretical graph. In other words, the cost is defined between a router and the network in which the destination host is located.

Figure 8.15 Hop counts in RIP



Access the text alternative for slide images.

Forwarding Table 1

Although the distance-vector algorithm we discussed in the previous section is concerned with exchanging distance vectors between neighboring nodes, the routers in an autonomous system need to keep forwarding tables to forward packets to their destination networks.

Figure 8.16 shows the three forwarding tables for the routers in Figure 8.15. Note that the first and the third columns together convey the same information as does a distance vector, but the cost shows the number of hops to the destination networks.

Figure 8.16 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

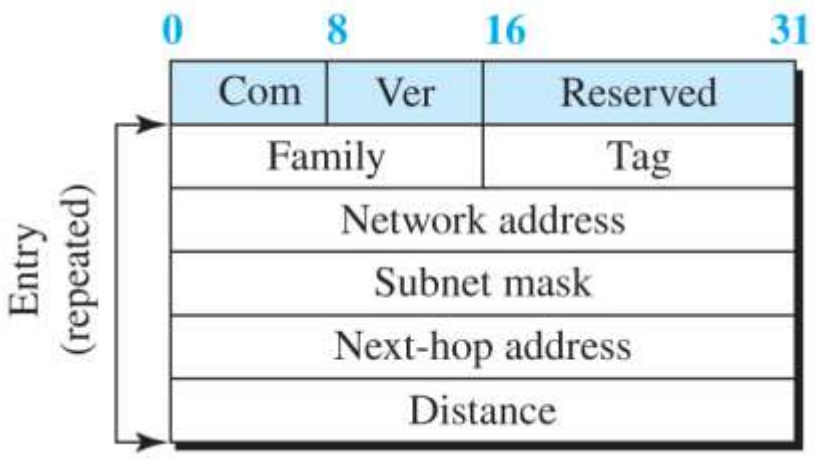
Destination network	Next router	Cost in hops
N1	R2	3
N2	R2	2
N3	——	1
N4	——	1

Access the text alternative for slide images.

RIP Implementation 1

RIP is implemented as a process that uses the service of UDP on the well-known port number 520. In BSD, RIP is a daemon process (a process running at the background), named routed (abbreviation for route daemon and pronounced route-dee). This means that although RIP is a routing protocol to help IP to route its datagrams through the AS, the RIP messages are encapsulated inside UDP user datagrams, which in turn are encapsulated inside IP datagrams. In other words, RIP runs at the application layer, but creates forwarding tables for IP at the network later.

Figure 8.17 RIP message format



Fields

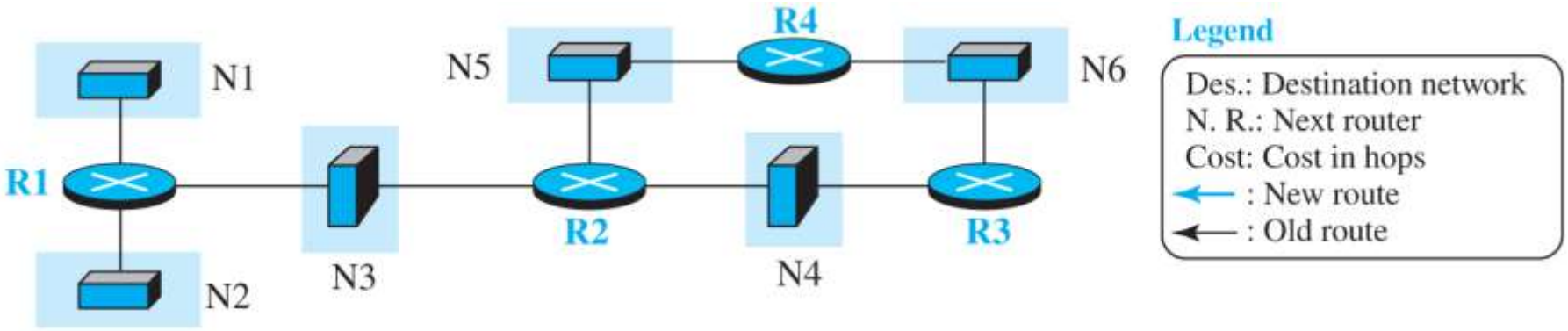
Com: Command, request (1), response (2)
Ver: Version, current version is 2
Family: Family of protocol, for TCP/IP value is 2
Tag: Information about autonomous system
Network address: Destination address
Subnet mask: Prefix length
Next-hop address: Address length
Distance: Number of hops to the destination

Access the text alternative for slide images.

Example 8.1

Figure 8.18 shows a more realistic example of the operation of RIP in an autonomous system. First, the figure shows all forwarding tables after all routers have been booted. Then we show changes in some tables when some update messages have been exchanged. Finally, we show the stabilized forwarding tables when there is no more change.

Figure 8.18 Example of an autonomous system using RIP (Part I)



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

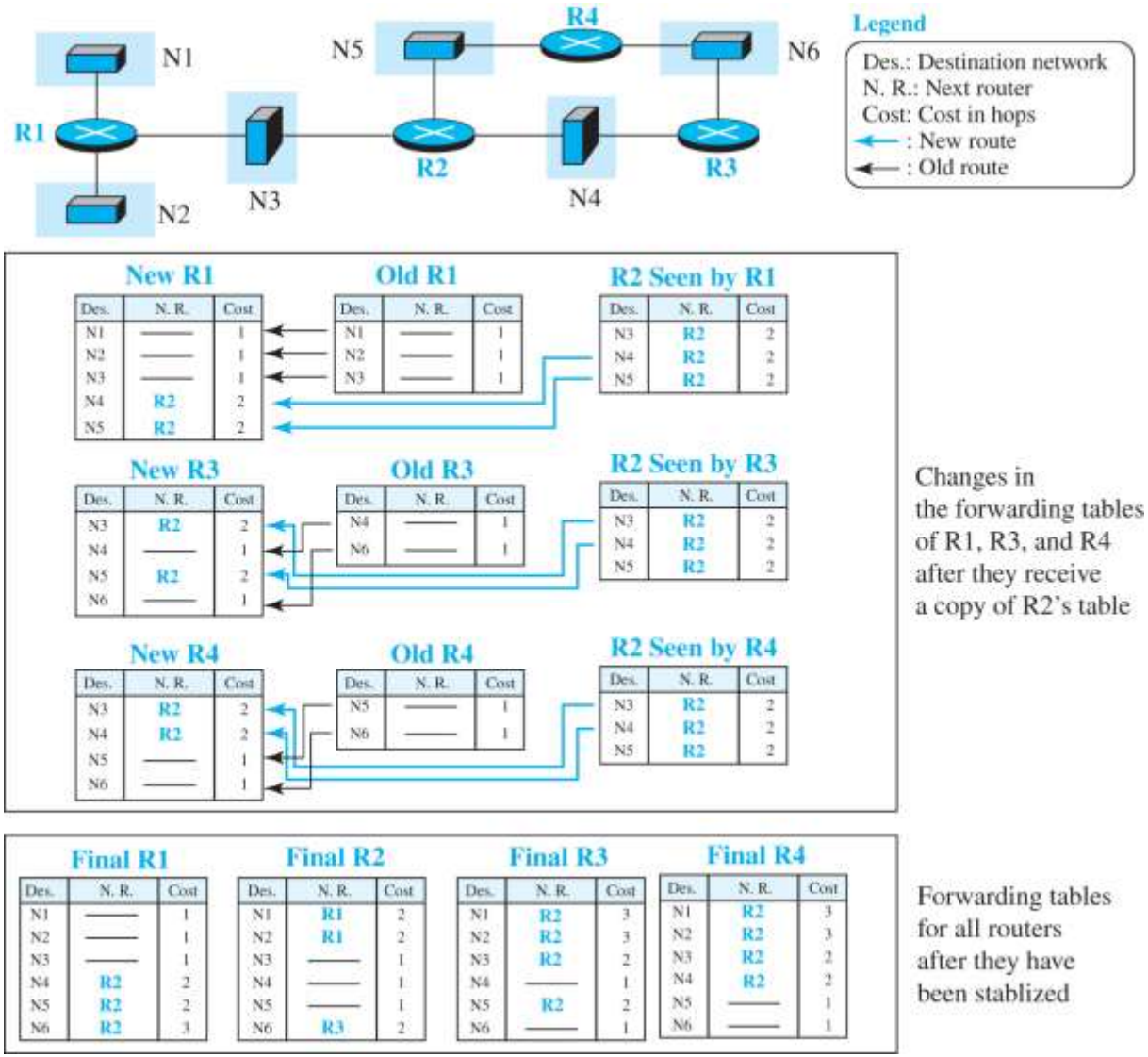
R4

Des.	N. R.	Cost
N5	—	1
N6	—	1

Forwarding tables after all routers booted

Access the text alternative for slide images.

Figure 8.18 Example of an autonomous system using RIP (Part II)



RIP Implementation 2

Before ending this section, let us mention the performance of RIP:

- *Update Messages*
- *Convergence of Forwarding Tables*
- *Robustness*

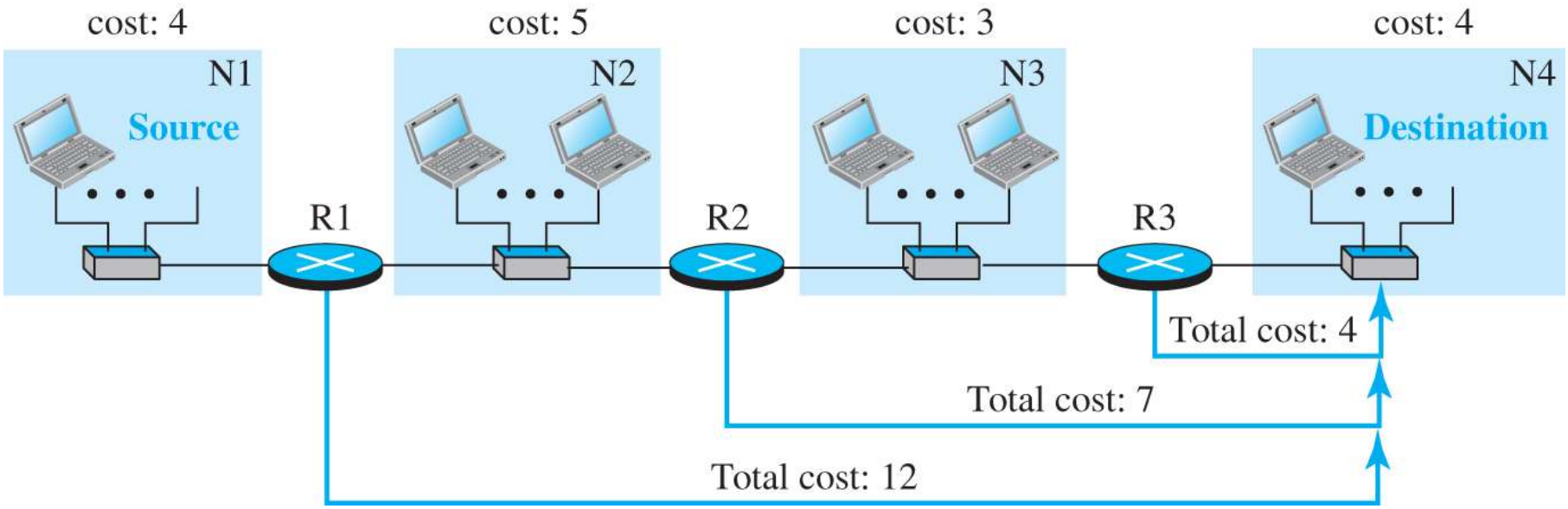
8.3.3 Open Shortest Path First

Open Shortest Path First (OSPF) is also an intradomain routing protocol like RIP, but it is based on the link-state routing protocol we described earlier in the chapter. OSPF is an open protocol, which means that the specification is a public document.

Metric

In OSPF, like RIP, the cost of reaching a destination from the host is calculated from the source router to the destination network. However, each link (network) can be assigned a weight based on the throughput, round-trip time, reliability, and so on.

Figure 8.19 Metric in OSPF



Access the text alternative for slide images.

Forwarding Table 2

Each OSPF router can create a forwarding table after finding the shortest-path tree between itself and the destination using Dijkstra's algorithm, described earlier in the chapter. Figure 8.20 shows the forwarding tables for the simple AS in Figure 8.19.

Figure 8.20 Forwarding tables in OSPF

Forwarding table for R1

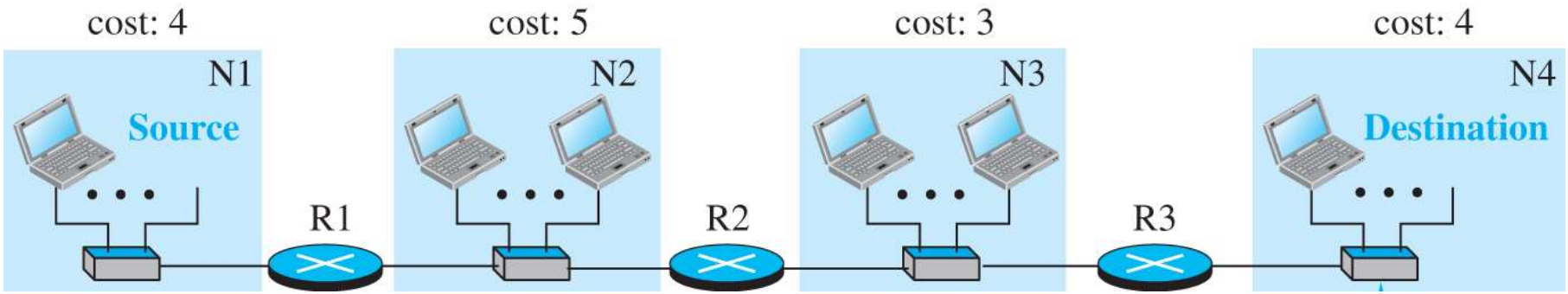
Destination network	Next router	Cost
N1	——	4
N2	——	5
N3	R2	8
N4	R2	12

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

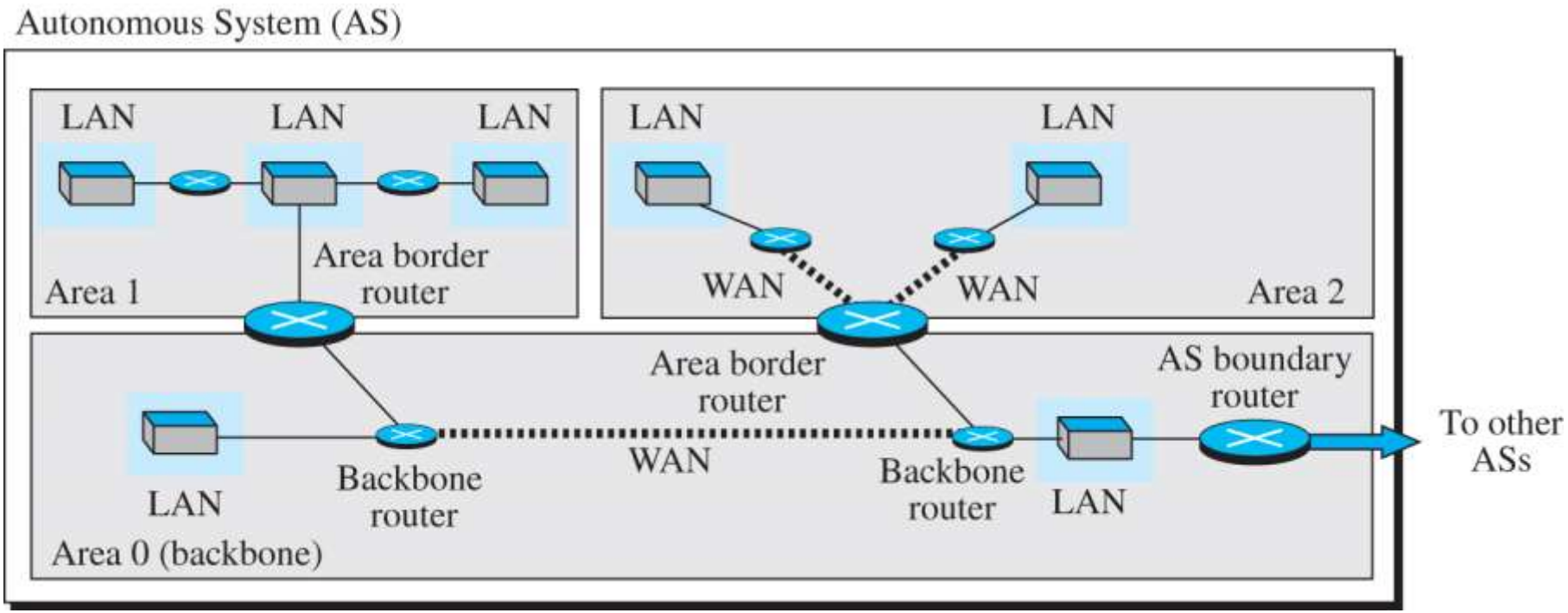


Access the text alternative for slide images.

Areas

Each OSPF router can create a forwarding table after finding the shortest-path tree between itself and the destination using Dijkstra's algorithm, described earlier in the chapter. Figure 8.20 shows the forwarding tables for the simple AS in Figure 8.19.

Figure 8.21 Areas in an autonomous system

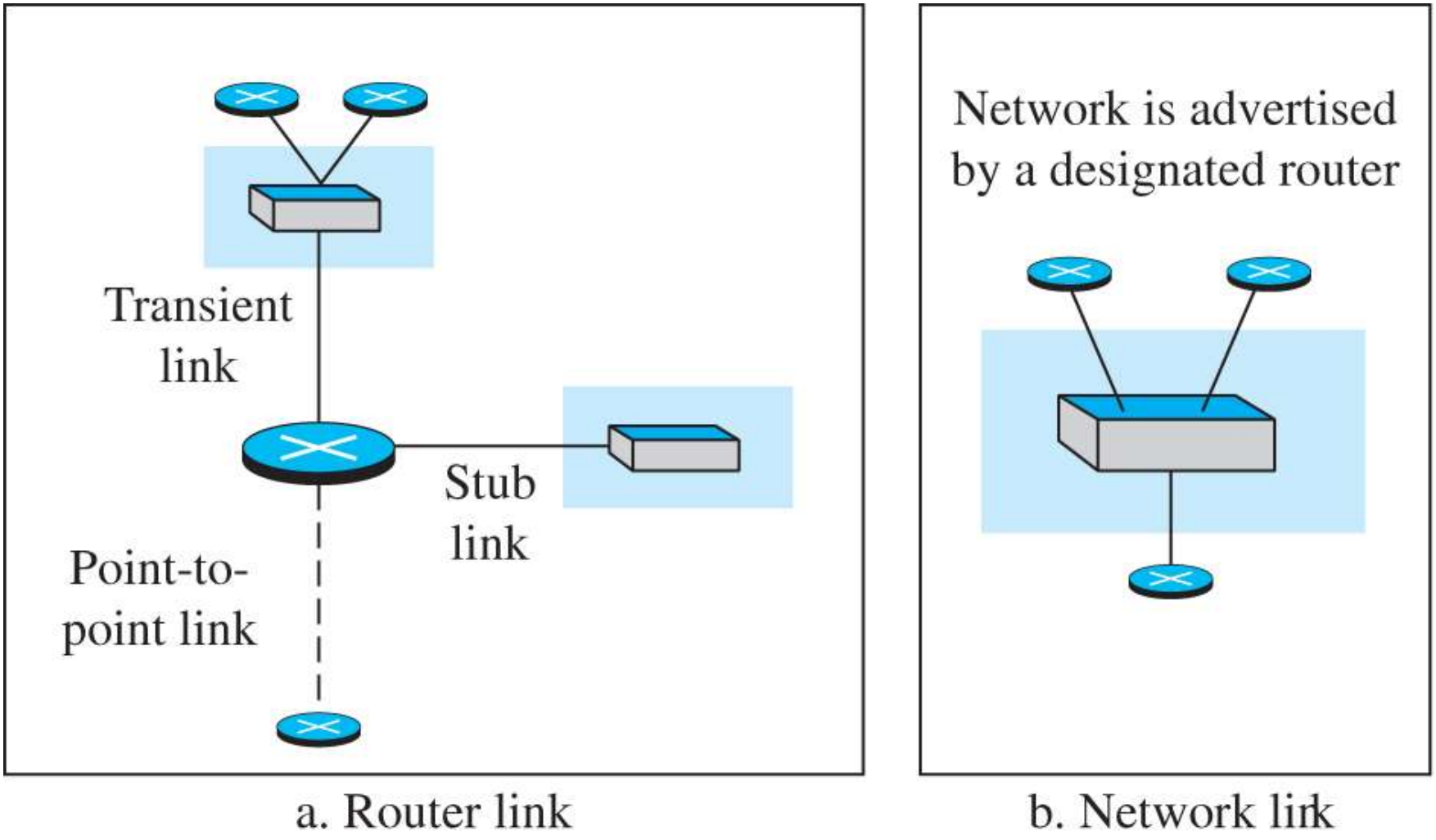


Access the text alternative for slide images.

Link-State Advertisement

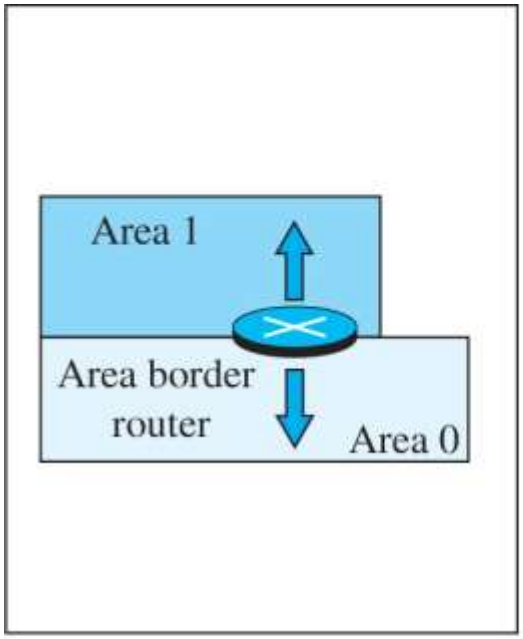
OSPF is based on the link-state routing algorithm, which requires that a router advertise the state of each link to all neighbors for the formation of the LSDB. When we discussed the link-state algorithm, we used the graph theory and assumed that each router is a node and each network between two routers is an edge. The situation is different in the real world, in which we need to advertise the existence of different entities as nodes, the different types of links that connect each node to its neighbors, and the different types of cost associated with each link.

Figure 8.22 Five different LSPs (Part I)

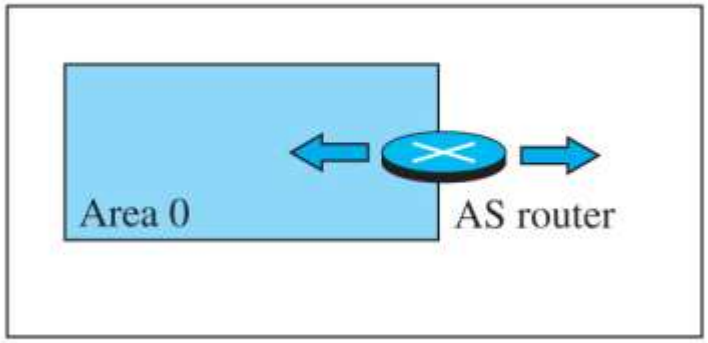


Access the text alternative for slide images.

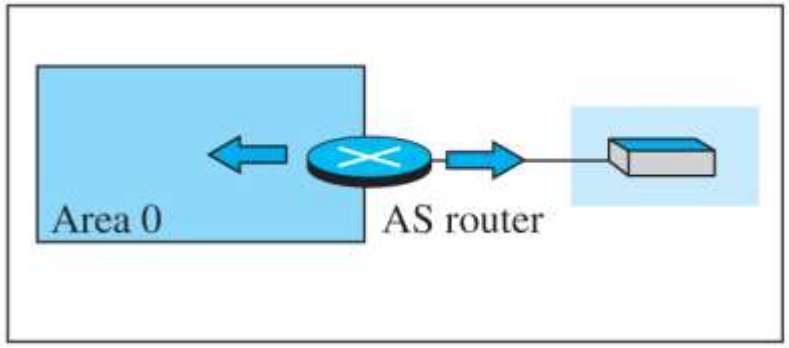
Figure 8.22 Five different LSPs (Part II)



c. Summary link to network



d. Summary link to AS

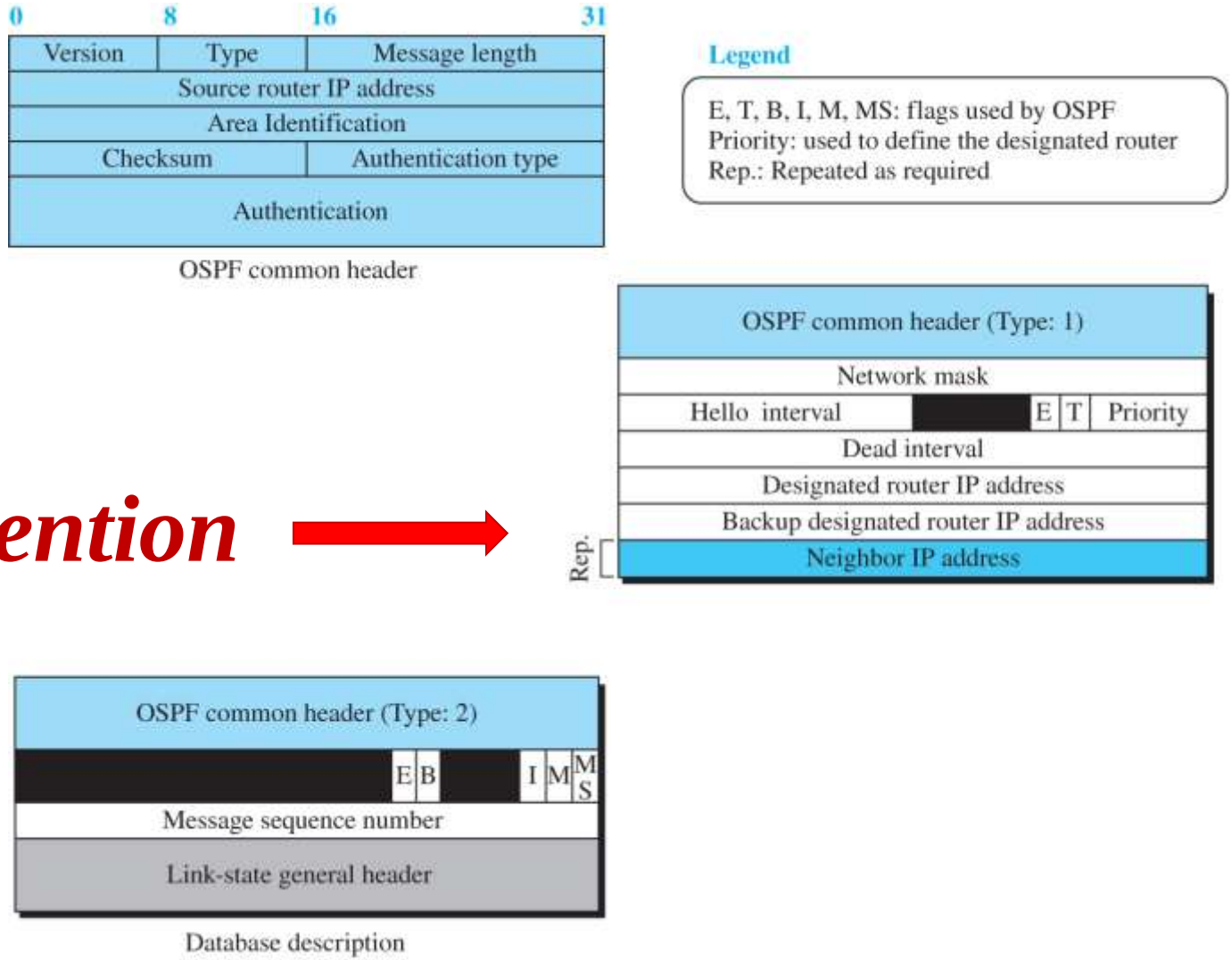


e. External link

OSPF Implementation

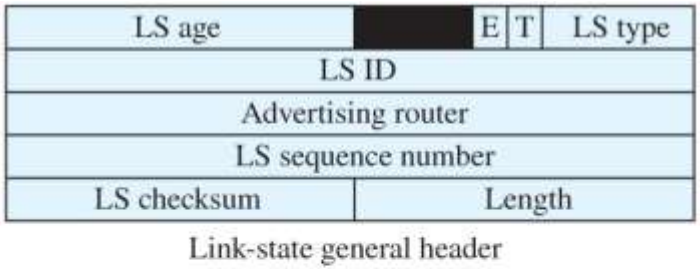
OSPF is implemented as a program in the network layer that uses the service of the IP for propagation. An IP datagram that carries a message from OSPF sets the value of the protocol field to 89. This means that although OSPF is a routing protocol to help IP to route its datagrams inside an AS, the OSPF messages are encapsulated inside datagrams. OSPF has gone through two versions: version 1 and version 2. Most implementations use version 2.

Figure 8.23 OSPF message formats (Part I)



Access the text alternative for slide images.

Figure 8.23 OSPF message formats (Part II)

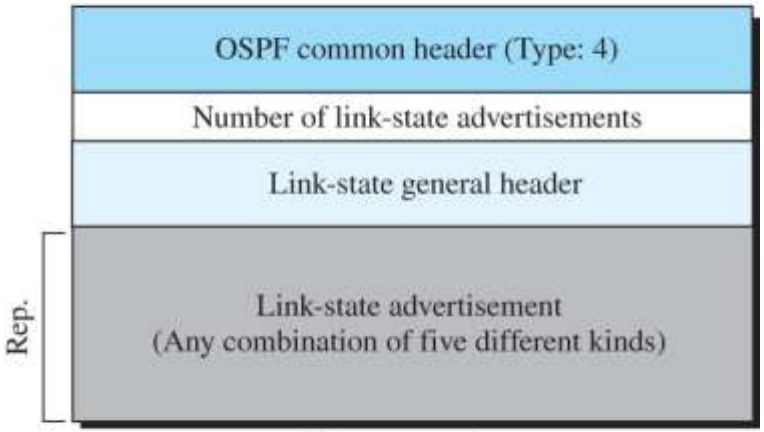


Link-state general header

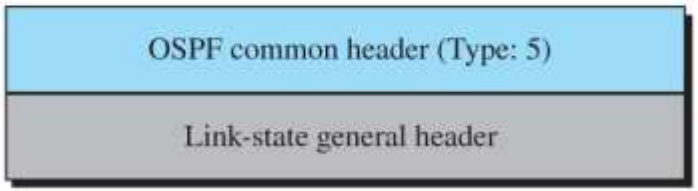
Legend

E, T, B, I, M, MS: flags used by OSPF
Priority: used to define the designated router
Rep.: Repeated as required

Attention →



Link-state update



Link-state acknowledgment

Access the text alternative for slide images.

Performance 1

Update Messages

The link-state messages in OSPF have a somewhat complex format. They also are flooded to the whole area. If the area is large, these messages may create heavy traffic and use a lot of bandwidth.

Convergence of Forwarding Tables

When the flooding of LSPs is completed, each router can create its own shortest-path tree and forwarding table; convergence is fairly quick. However, each router needs to run the Dijkstra's algorithm, which may take some time.

Robustness

The OSPF protocol is more robust than RIP because, after receiving the completed LSDB, each router is independent and does not depend on other.

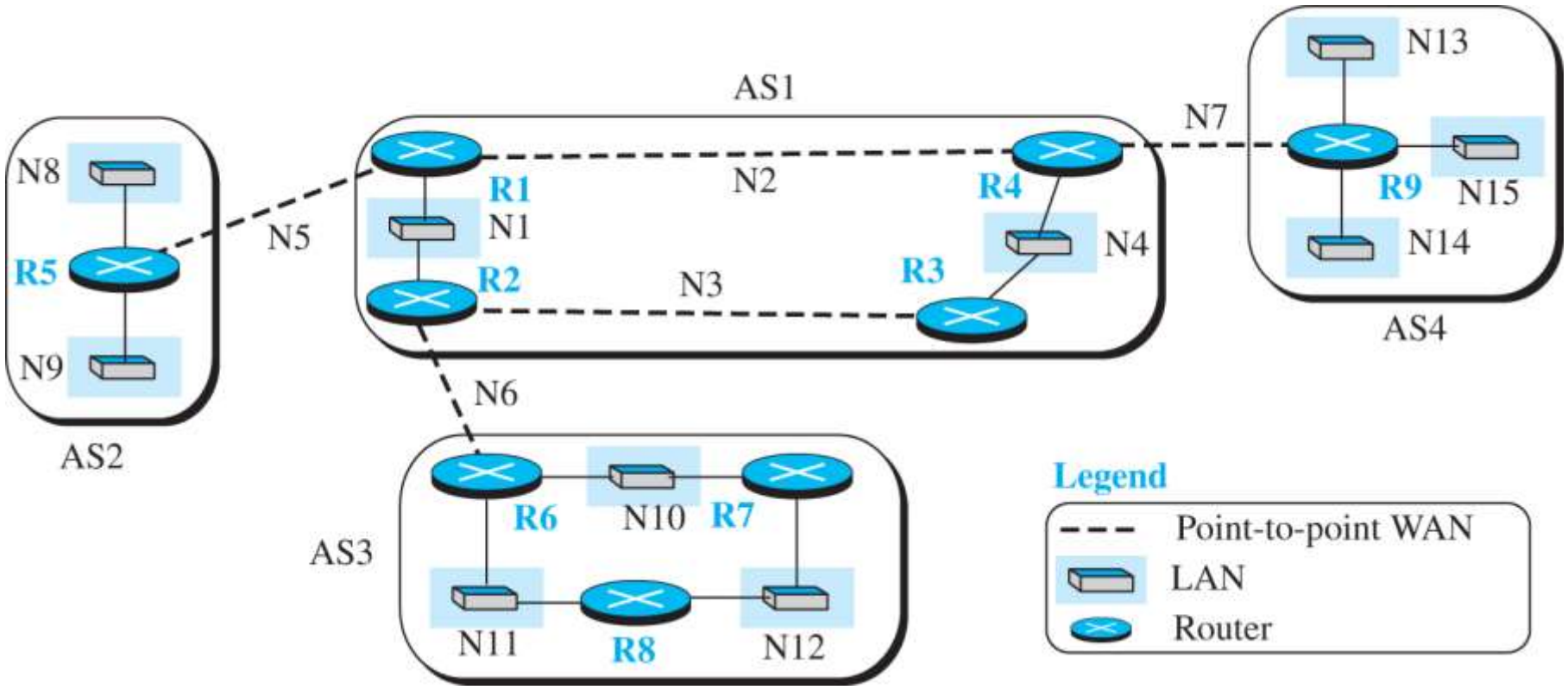
8.3.4 Border Gateway Protocol

The Border Gateway Protocol version 4 (BGP4) is the only interdomain routing protocol used in the Internet today. BGP4 is based on the path-vector algorithm we described before, but it is tailored to provide information about the reachability of networks in the Internet.

Introduction

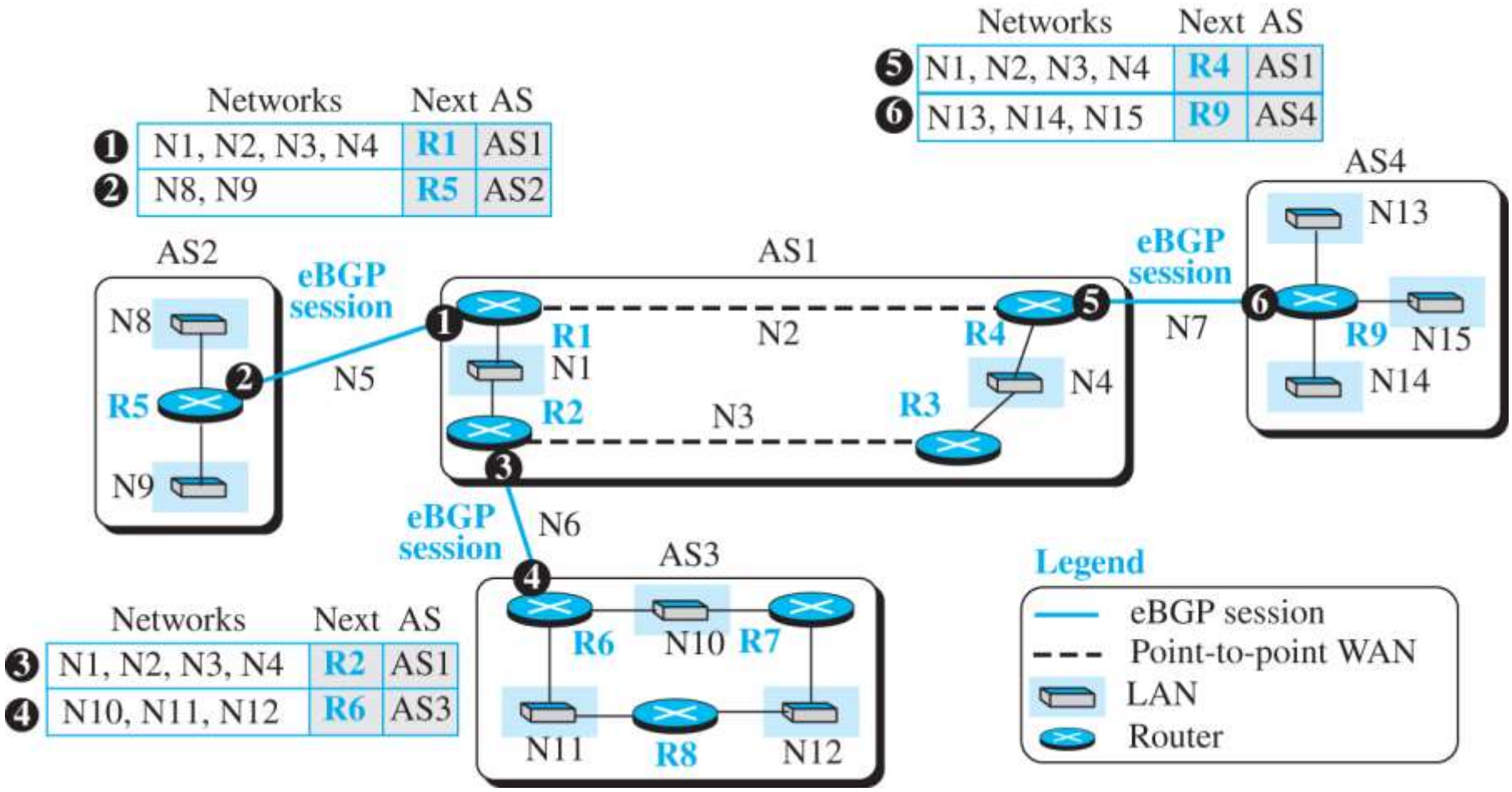
BGP, and in particular BGP4, is a complex protocol. In this section, we introduce the basics of BGP and its relationship with intradomain routing protocols (RIP or OSPF). Figure 8.24 shows an example of an internet with four autonomous systems. AS2, AS3, and AS4 are stub autonomous systems; AS1 is a transient one. In our example, data exchange between AS2, AS3, and AS4 should pass through AS1.

Figure 8.24 A sample internet with four ASs



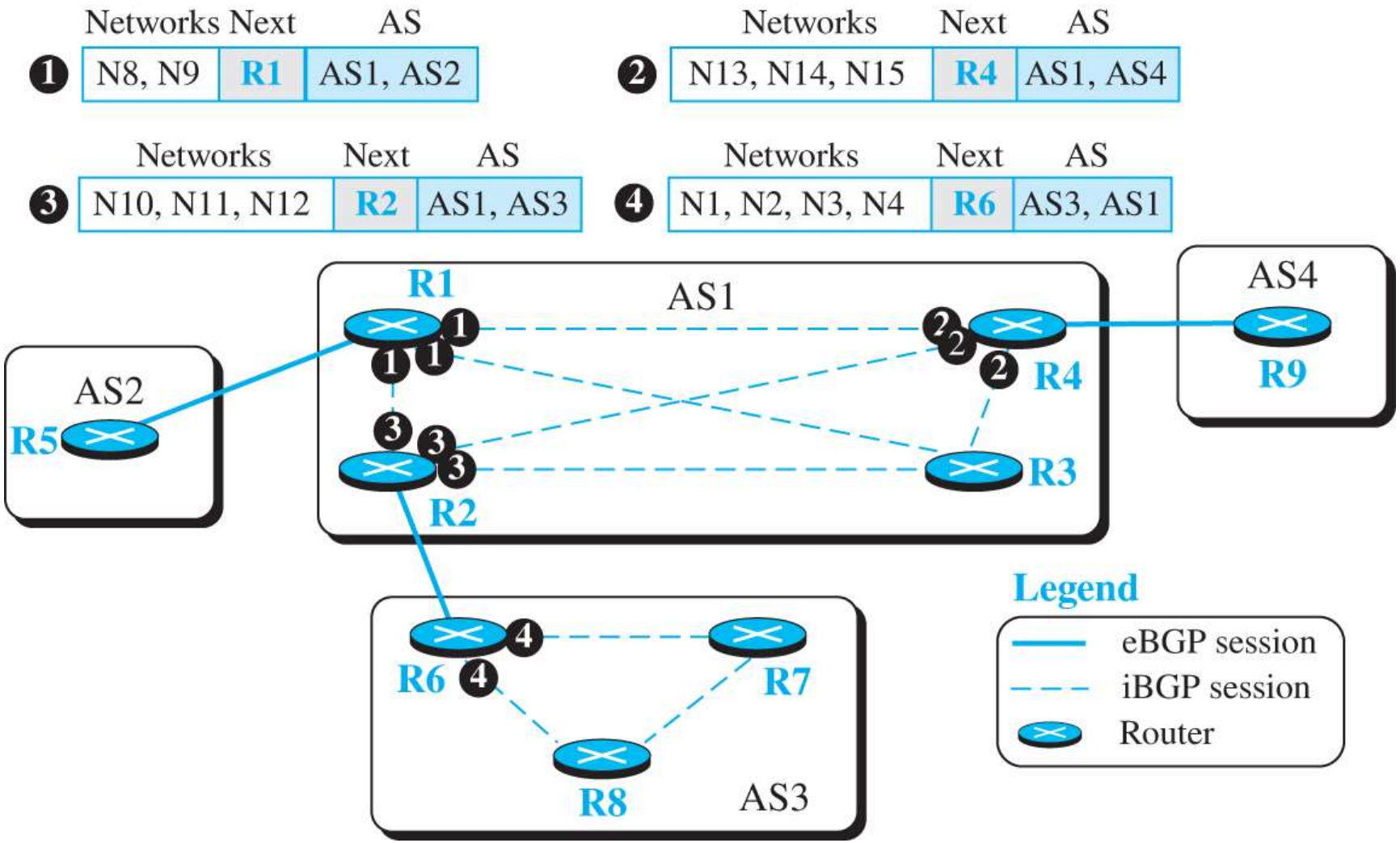
Access the text alternative for slide images.

Figure 8.25 eBGP operation



Access the text alternative for slide images.

Figure 8.26 Combination of eBGP and iBGP sessions in our internet



Access the text alternative for slide images.

Figure 8.27 Finalized BGP path tables (Part I)

Networks	Next	Path
N8, N9	R5	AS1, AS2
N10, N11, N12	R2	AS1, AS3
N13, N14, N15	R4	AS1, AS4

Path table for R1

Networks	Next	Path
N8, N9	R1	AS1, AS2
N10, N11, N12	R6	AS1, AS3
N13, N14, N15	R1	AS1, AS4

Path table for R2

Networks	Next	Path
N8, N9	R2	AS1, AS2
N10, N11, N12	R2	AS1, AS3
N13, N14, N15	R4	AS1, AS4

Path table for R3

Access the text alternative for slide images.

Figure 8.27 Finalized BGP path tables (Part II)

Networks	Next	Path
N8, N9	R1	AS1, AS2
N10, N11, N12	R1	AS1, AS3
N13, N14, N15	R9	AS1, AS4

Path table for R4

Networks	Next	Path
N1, N2, N3, N4	R1	AS2, AS1
N10, N11, N12	R1	AS2, AS1, AS3
N13, N14, N15	R1	AS2, AS1, AS4

Path table for R5

Networks	Next	Path
N1, N2, N3, N4	R2	AS3, AS1
N8, N9	R2	AS3, AS1, AS2
N13, N14, N15	R2	AS3, AS1, AS4

Path table for R6

Figure 8.27 Finalized BGP path tables (Part III)

Networks	Next	Path
N1, N2, N3, N4	R6	AS3, AS1
N8, N9	R6	AS3, AS1, AS2
N13, N14, N15	R6	AS3, AS1, AS4

Path table for R7

Networks	Next	Path
N1, N2, N3, N4	R6	AS3, AS1
N8, N9	R6	AS3, AS1, AS2
N13, N14, N15	R6	AS3, AS1, AS4

Path table for R8

Networks	Next	Path
N1, N2, N3, N4	R4	AS4, AS1
N8, N9	R4	AS4, AS1, AS2
N10, N11, N12	R4	AS4, AS1, AS3

Path table for R9

Figure 8.28 Forwarding tables after injection from BGP (Part I)

Des.	Next	Cost
N1	—	1
N4	R4	2
N8	R5	1
N9	R5	1
N10	R2	2
N11	R2	2
N12	R2	2
N13	R4	2
N14	R4	2
N15	R4	2

Table for R1

Des.	Next	Cost
N1	—	1
N4	R3	2
N8	R1	2
N9	R1	2
N10	R6	1
N11	R6	1
N12	R6	1
N13	R3	3
N14	R3	3
N15	R3	3

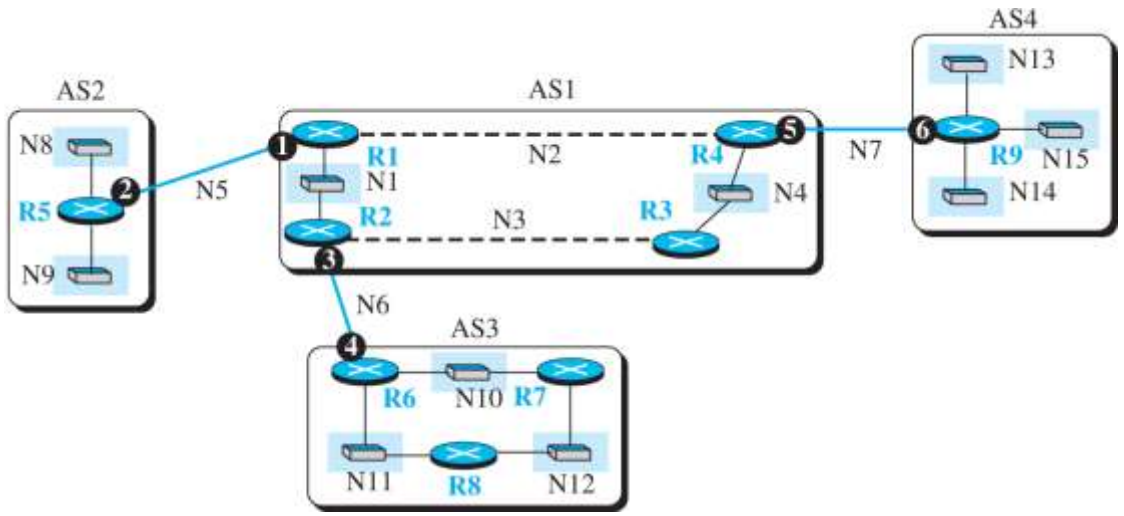
Table for R2

Des.	Next	Cost
N1	R2	2
N4	—	1
N8	R2	3
N9	R2	3
N10	R2	2
N11	R2	2
N12	R2	2
N13	R4	2
N14	R4	2
N15	R4	2

Table for R3

Des.	Next	Cost
N1	R1	2
N4	—	1
N8	R1	2
N9	R1	2
N10	R3	3
N11	R3	3
N12	R3	3
N13	R9	1
N14	R9	1
N15	R9	1

Table for R4



Access the text alternative for slide images.

Figure 8.28 Forwarding tables after injection from BGP (Part II)

Des.	Next	Cost
N8	—	1
N9	—	1
0	R1	1

Table for R5

Des.	Next	Cost
N10	—	1
N11	—	1
N12	R7	2
0	R2	1

Table for R6

Des.	Next	Cost
N10	—	1
N11	R6	2
N12	—	1
0	R6	2

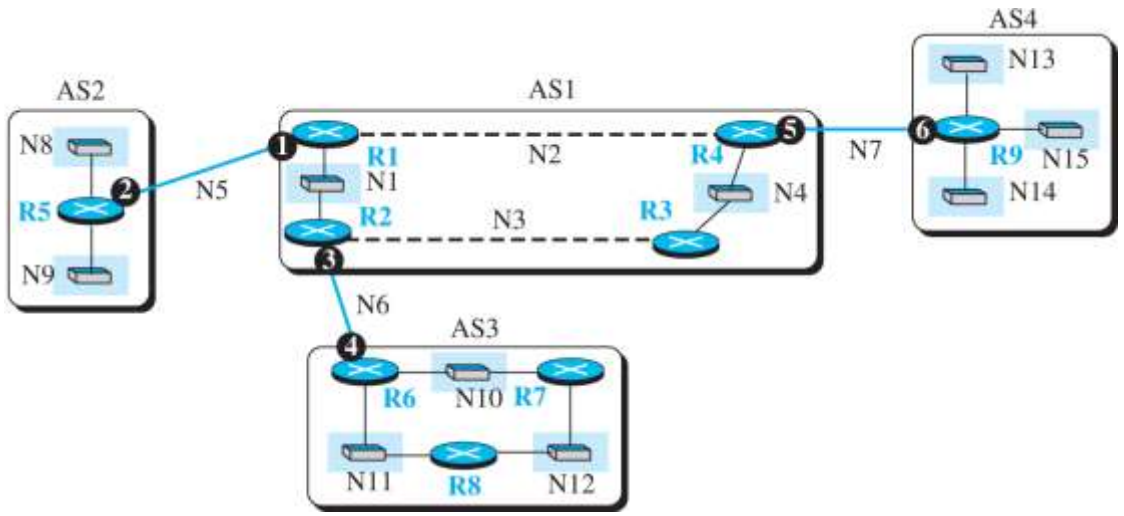
Table for R7

Des.	Next	Cost
N10	R6	2
N11	—	1
N12	—	1
0	R6	2

Table for R8

Des.	Next	Cost
N13	—	1
N14	—	1
N15	—	1
0	R4	1

Table for R9

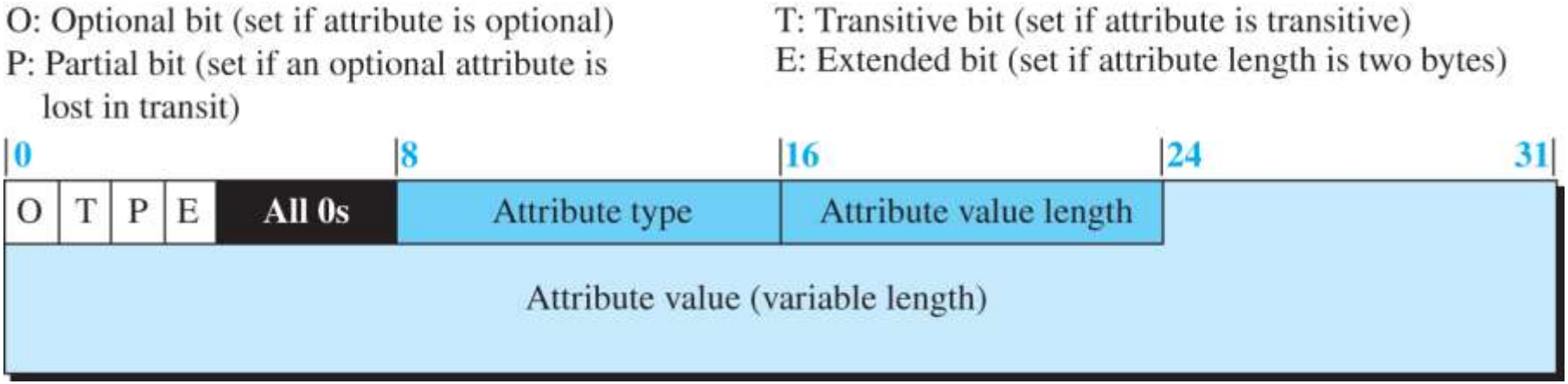


Access the text alternative for slide images.

Path Attributes

In both intradomain routing protocols (RIP or OSPF), a destination is normally associated with two pieces of information: next hop and cost. The first one shows the address of the next router to deliver the packet; the second defines the cost to the final destination. Interdomain routing is more involved and naturally needs more information about how to reach the final destination. In BGP these pieces are called path attributes.

Figure 8.29 Format of path attribute

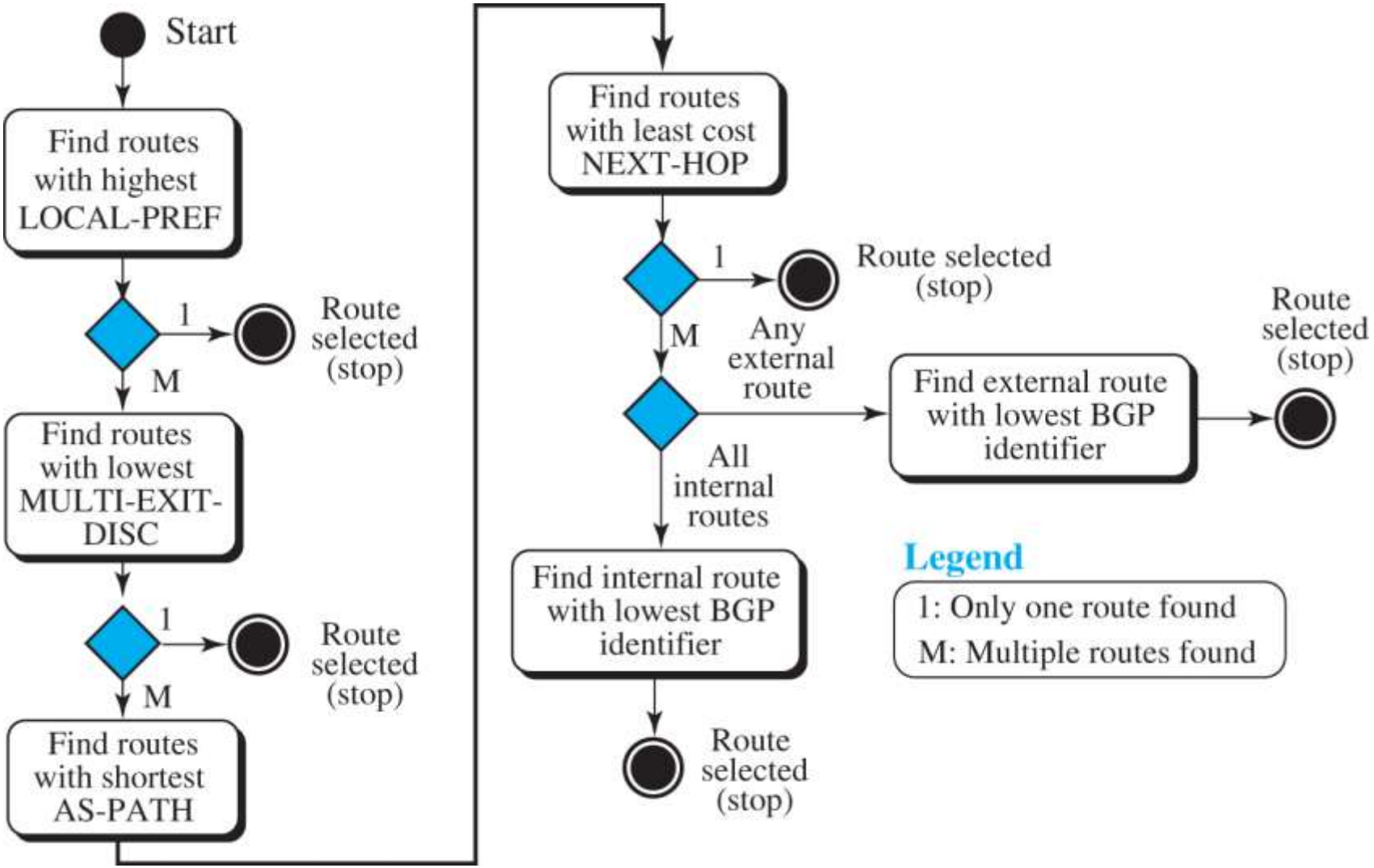


Access the text alternative for slide images.

Route Selection

So far in this section, we have been silent about how a route is selected by a BGP router mostly because our simple example has one route to a destination. In the case where multiple routes are received to a destination, BGP needs to select one among them. The route selection process in BGP is not as easy as the ones in the intradomain routing protocol that is based on the shortest-path tree. A route in BGP has some attributes attached to it and it may come from an eBGP session or an iBGP session. Figure 8.30 shows the flow diagram as used by common implementations

Figure 20.30 Flow diagram for route selection

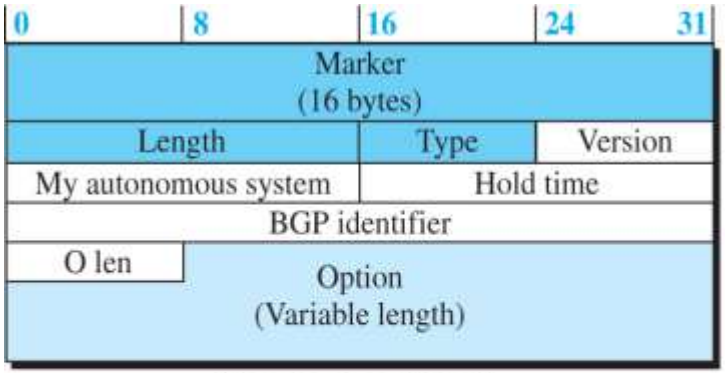


Access the text alternative for slide images.

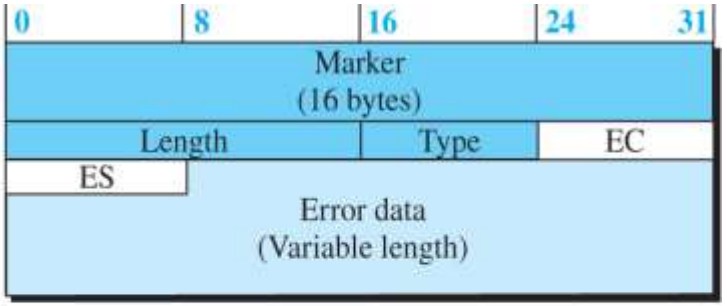
Messages

BGP uses four types of messages for communication between the BGP speakers across the ASs and inside an AS: open, update, keepalive, and notification (see Figure 8.31). All BGP packets share the same common header.

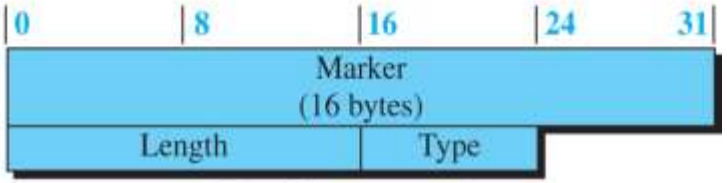
Figure 8.31 BGP messages



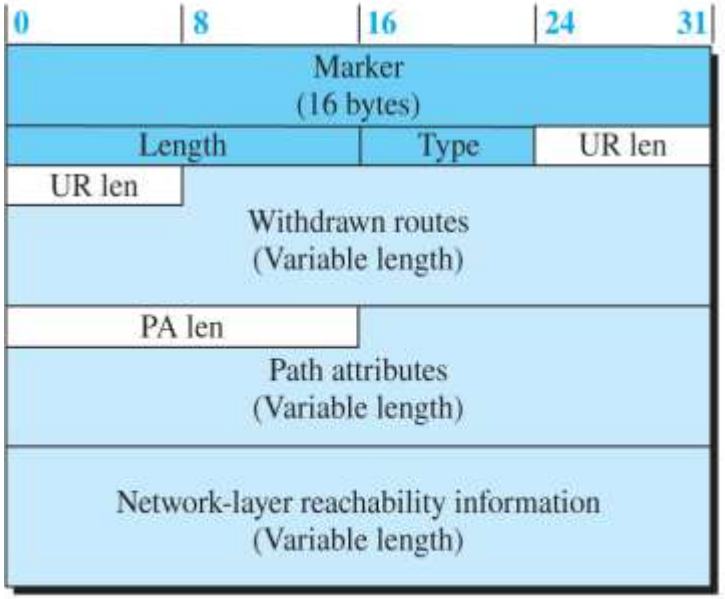
Open message (type 1)



Notification message (type 3)



Keepalive message (type 4)



Update message (type 2)

Fields in common header

Marker: Reserved for authentication
Length: Length of total message in bytes
Type: Type of message (1 to 4)

Abbreviations

O len: Option length
EC: Error code
ES: Error subcode
UR len: Unfeasible route length
PA len: Path attribute length

Access the text alternative for slide images.

Performance 2

BGP performance can be compared with RIP. BGP speakers exchange a lot of messages to create forwarding tables, but BGP is free from loops and count-to-infinity. The same weakness we mention for RIP about propagation of failure and corruptness also exists in BGP.

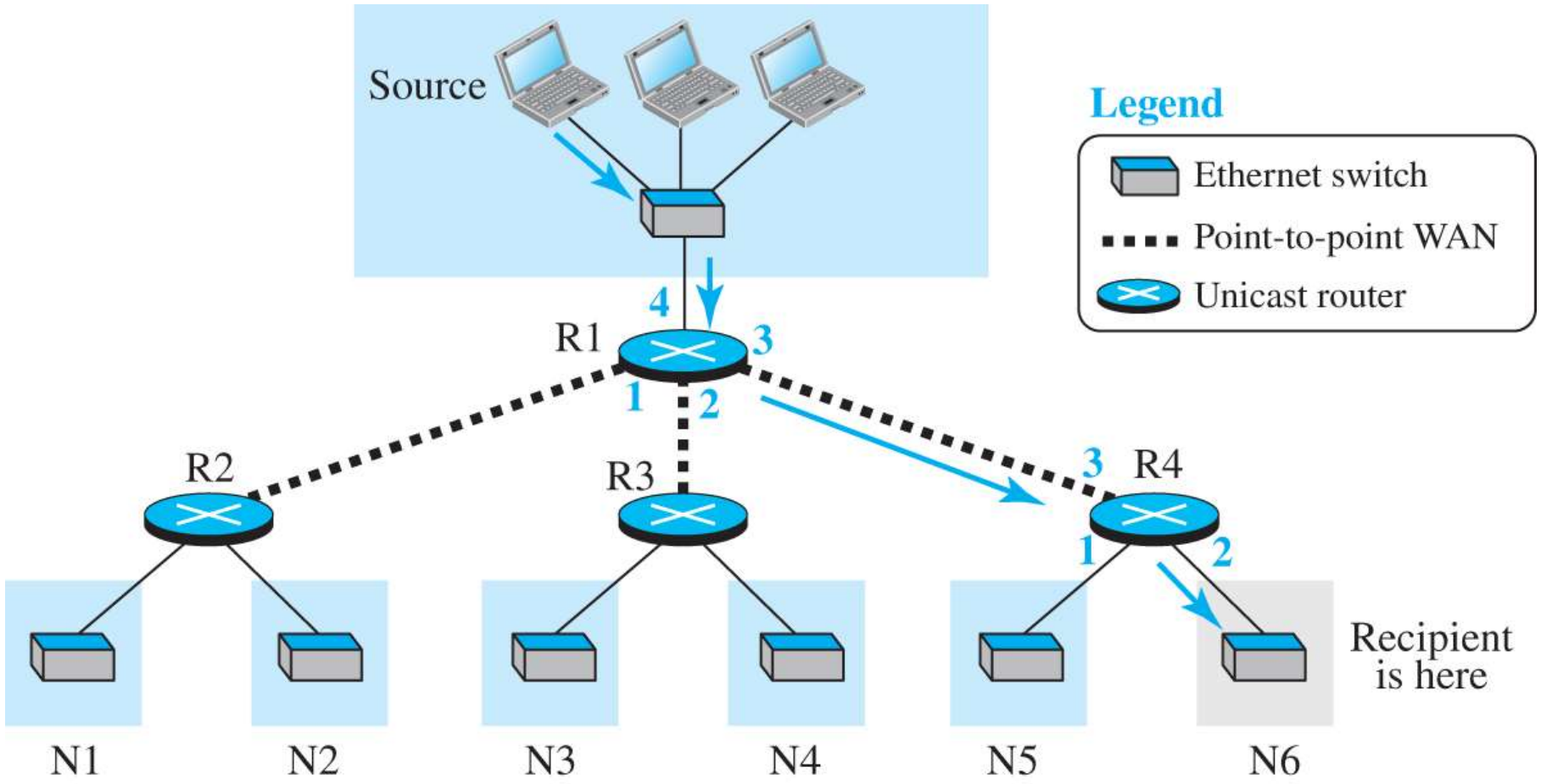
8-4 MULTICAST ROUTING

Communication in the Internet today is not only unicasting; multicasting communication is growing fast. In this section, we first discuss the general ideas behind unicasting, multicasting, and broadcasting. We then talk about some basic issues in multicast routing. Finally, we discuss multicasting routing protocols in the Internet.

8.4.1 Unicasting

In unicasting, there is one source and one destination network. The relationship between the source and the destination network is one to one. Each router in the path of the datagram tries to forward the packet to one and only one of its interfaces.

Figure 8.32 Unicasting



Access the text alternative for slide images.

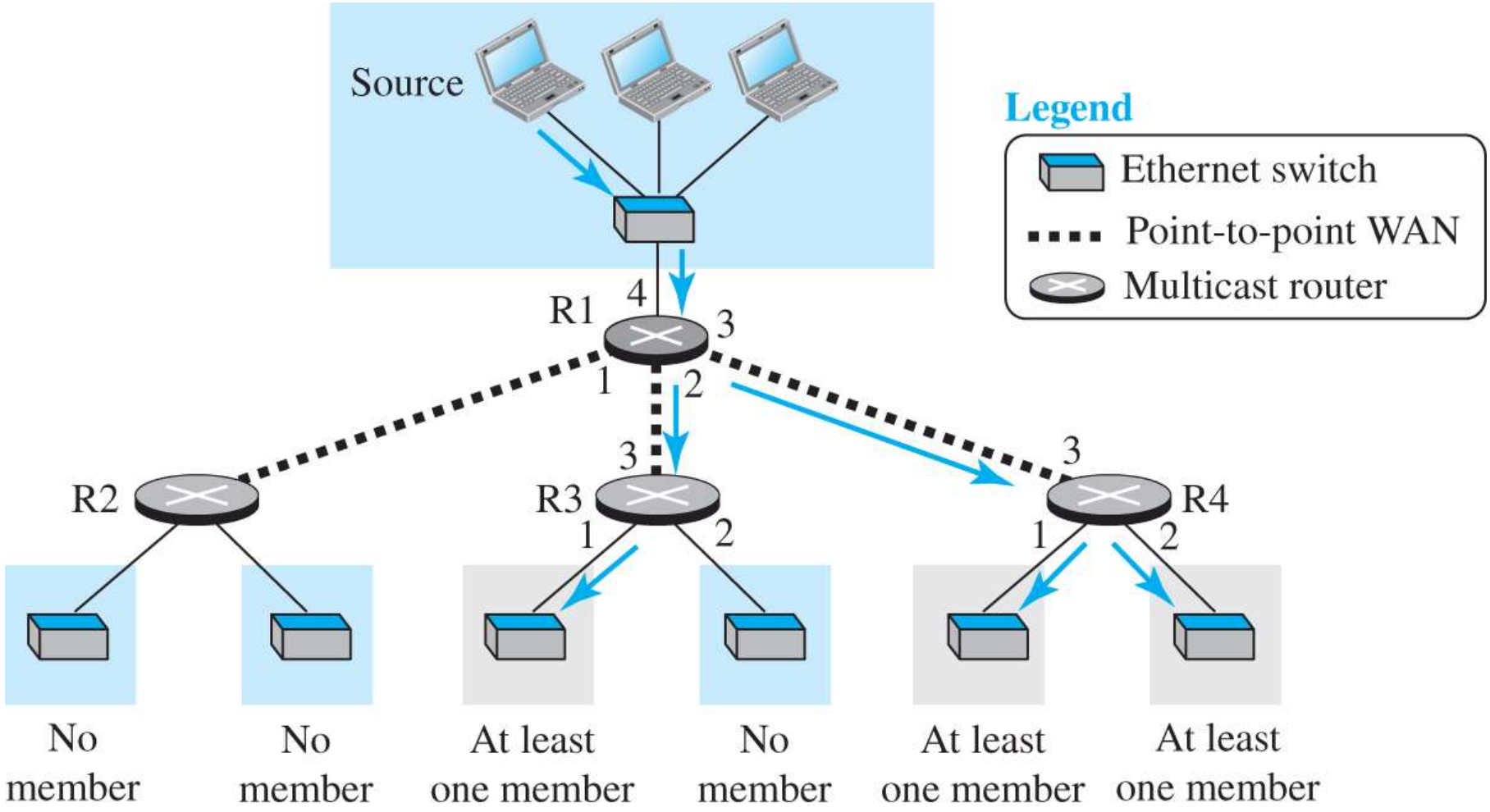
8.4.2 Multicasting

In multicasting, there is one source and a group of destinations. The relationship is one to many. In this type of communication, the source address is a unicast address, but the destination address is a group address, a group of one or more destination networks in which there is at least one member of the group that is interested in receiving the multicast datagram. The group address defines the members of the group.

Multicast Applications

Multicasting has many applications today, such as access to distributed databases, information dissemination, teleconferencing, and distance learning.

Figure 8.33 Multicasting



Access the text alternative for slide images.

Multicast Addresses

We discuss multicast addresses for IPv4 and IPv6 in Chapter 7.

8.4.3 Multicast Distance Vector

The Distance Vector Multicast Routing Protocol (DVMRP) is the extension of the Routing Information Protocol (RIP) which is used in unicast routing.

It uses the source-based tree approach to multicasting.

It is worth mentioning that each router in this protocol that receives a multicast packet to be forwarded implicitly creates a source-based multicast tree in

three steps:

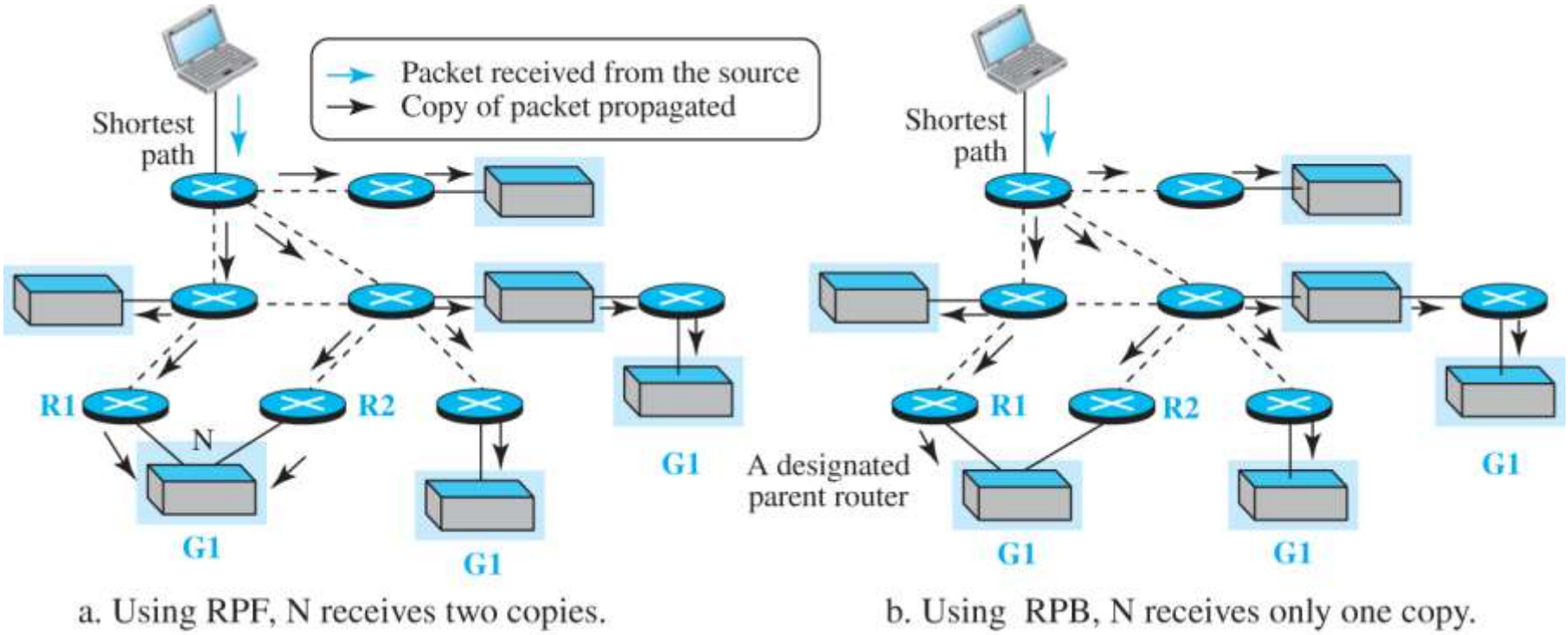
Reverse Cast Forwarding (RPF)

The first algorithm, reverse path forwarding (RPF), forces the router to forward a multicast packet from one specific interface: the one which has come through the shortest path from the source to the router. How can a router know which interface is in this path if the router does not have a shortest-path tree rooted at the source? The router uses the first property of the shortest-path tree we discussed in unicast routing, which says that the shortest path from A to B is also the shortest path from B to A.

Reverse Path Broadcasting (RPB)

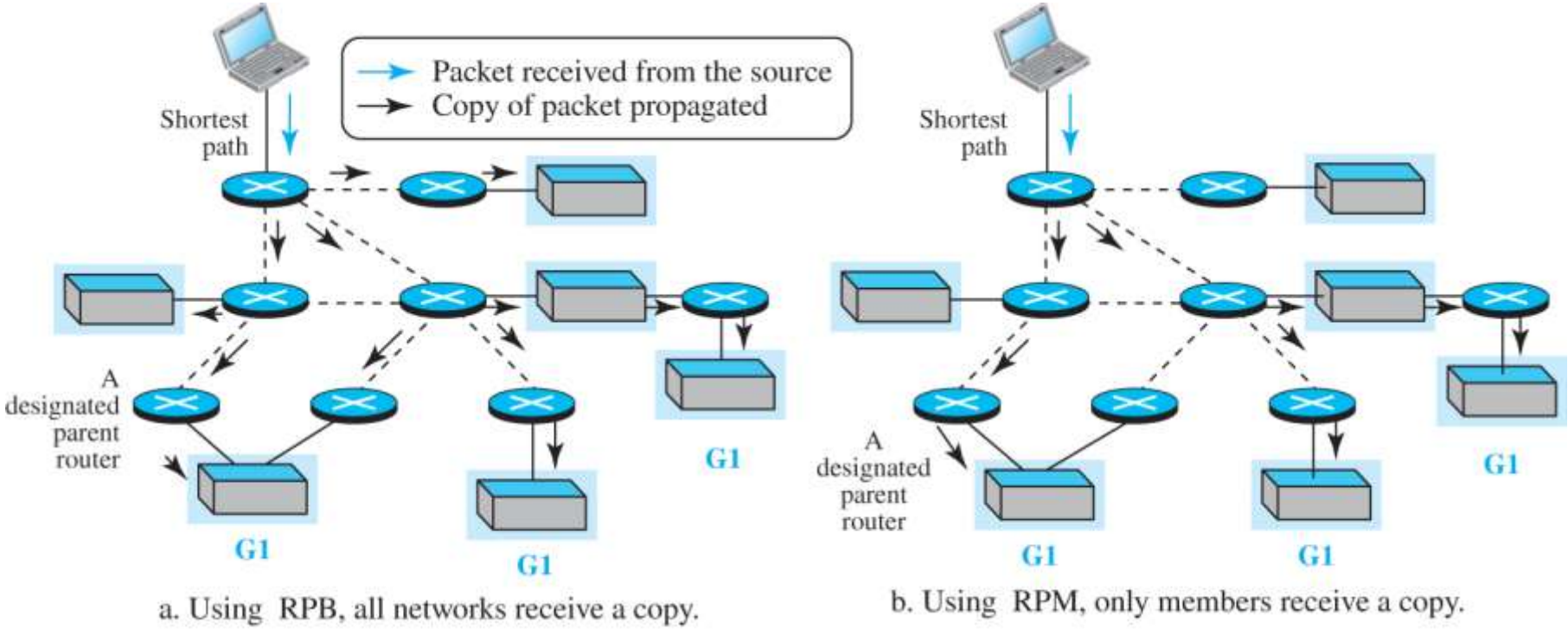
The RPF algorithm helps a router to forward only one copy received from a source and drop the rest. However, when we think about broadcasting in the second step, we need to remember that destinations are all the networks (LANs) in the internet. To be efficient, we need to prevent each network from receiving more than one copy of the packet. When a router that is not the parent of the attached network receives a multicast packet, it simply drops the packet. There are several ways that the parent of the network related to a network can be selected; one way is to select the router that has the shortest path to the source.

Figure 8.34 Multicasting versus multiple unicasting



Access the text alternative for slide images.

Figure 8.35 RPF versus RPB



a. Using RPB, all networks receive a copy.

b. Using RPM, only members receive a copy.

Access the text alternative for slide images.

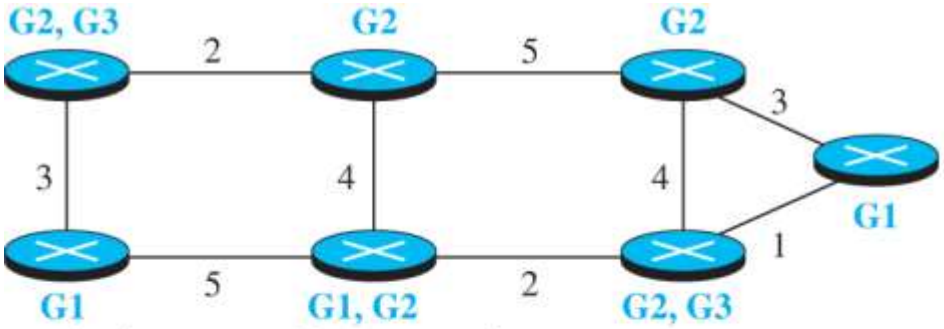
Reverse Cast Multicasting (RPM)

RPB does not multicast the packet, it broadcasts it. This is not efficient. To increase efficiency, the multicast packet must reach only those networks that have active members for that particular group. This is called reverse path multicasting (RPM). To change the broadcast shortest-path tree to a multicast shortest-path tree, each router needs to prune (make inactive) the interfaces that do not reach a network with active members corresponding to a particular source-group combination.

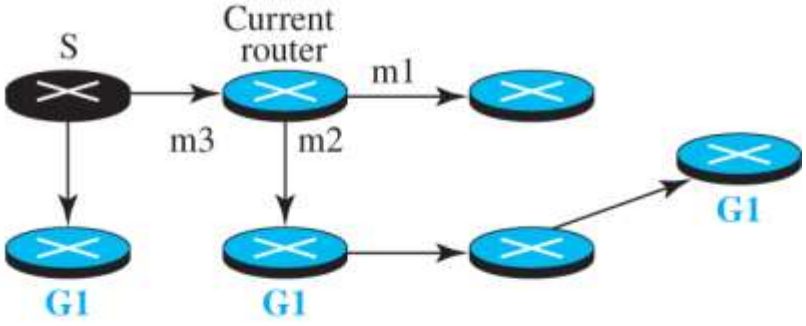
8.4.4 Multicast Link State (MOSPF)

Multicast Open Shortest Path First (MOSPF) is the extension of the Open Shortest Path First (OSPF) protocol, which is used in unicast routing. It also uses the source-based tree approach to multicasting. If the internet is running a unicast link-state routing algorithm, the idea can be extended to provide a multicast link-state routing algorithm. To extend unicasting to multicasting, each router needs to have another database, as with the case of unicast distance-vector routing, to show which interface has an active member in a particular group.

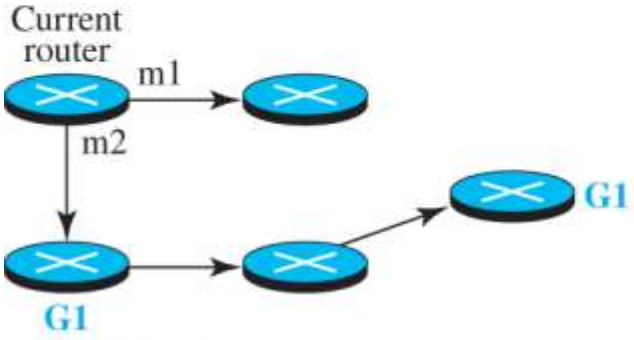
Figure 8.36 Example of tree formation in MOSPF



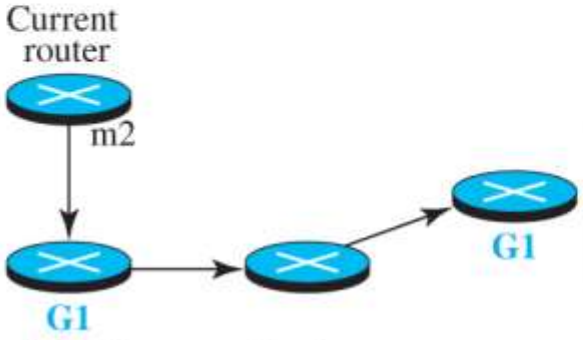
a. An internet with some active groups



b. S-G1 shortest-path tree



c. S-G1 subtree seen by current router



d. S-G1 pruned subtree

Forwarding table for current router

Group-Source	Interface
S, G1	m2
...	...

Access the text alternative for slide images.

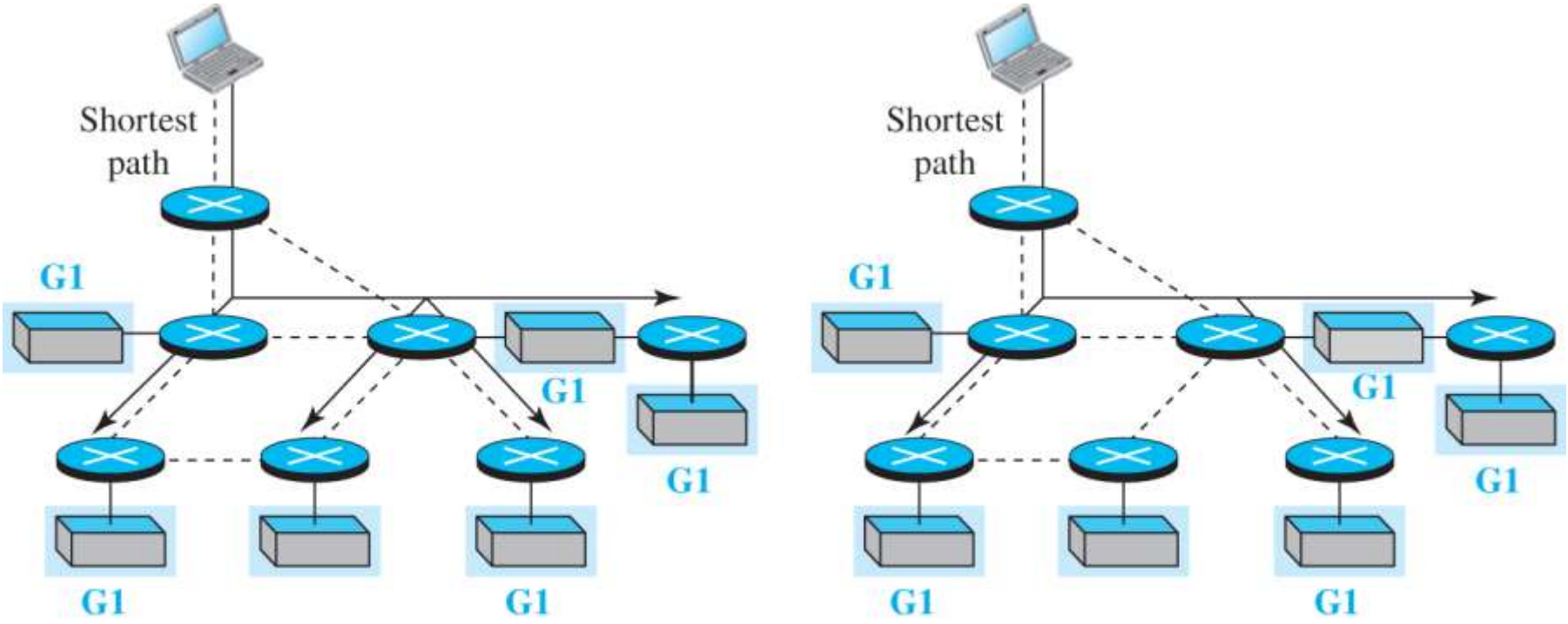
8.4.5 Protocol Independent Multicast

Protocol Independent Multicast (PIM) is the name given to a common protocol that needs a unicast routing protocol for its operation, but the unicast protocol can be either a distance-vector protocol or a link-state protocol. In other words, PIM needs to use the forwarding table of a unicast routing protocol to find the next router in a path to the destination, but it does not matter how the forwarding table is created. PIM has another interesting feature: it can work in two different modes: dense and sparse.

Protocol Independent Multicast-Dense Mode)

When the number of routers with attached members is large relative to the number of routers in the internet, PIM works in the dense mode and is called PIM-DM. In this mode, the protocol uses a source-based tree approach and is similar to DVMRP, but simpler. PIM-DM uses only two strategies described in DVMRP: RPF and RPM. But unlike DVMRP, forwarding of a packet is not suspended awaiting pruning of the first subtree. Let us explain the two steps used in PIM-DM to clear the matter.

Figure 8.37 Idea behind PIM-DM



a. First packet is broadcast.

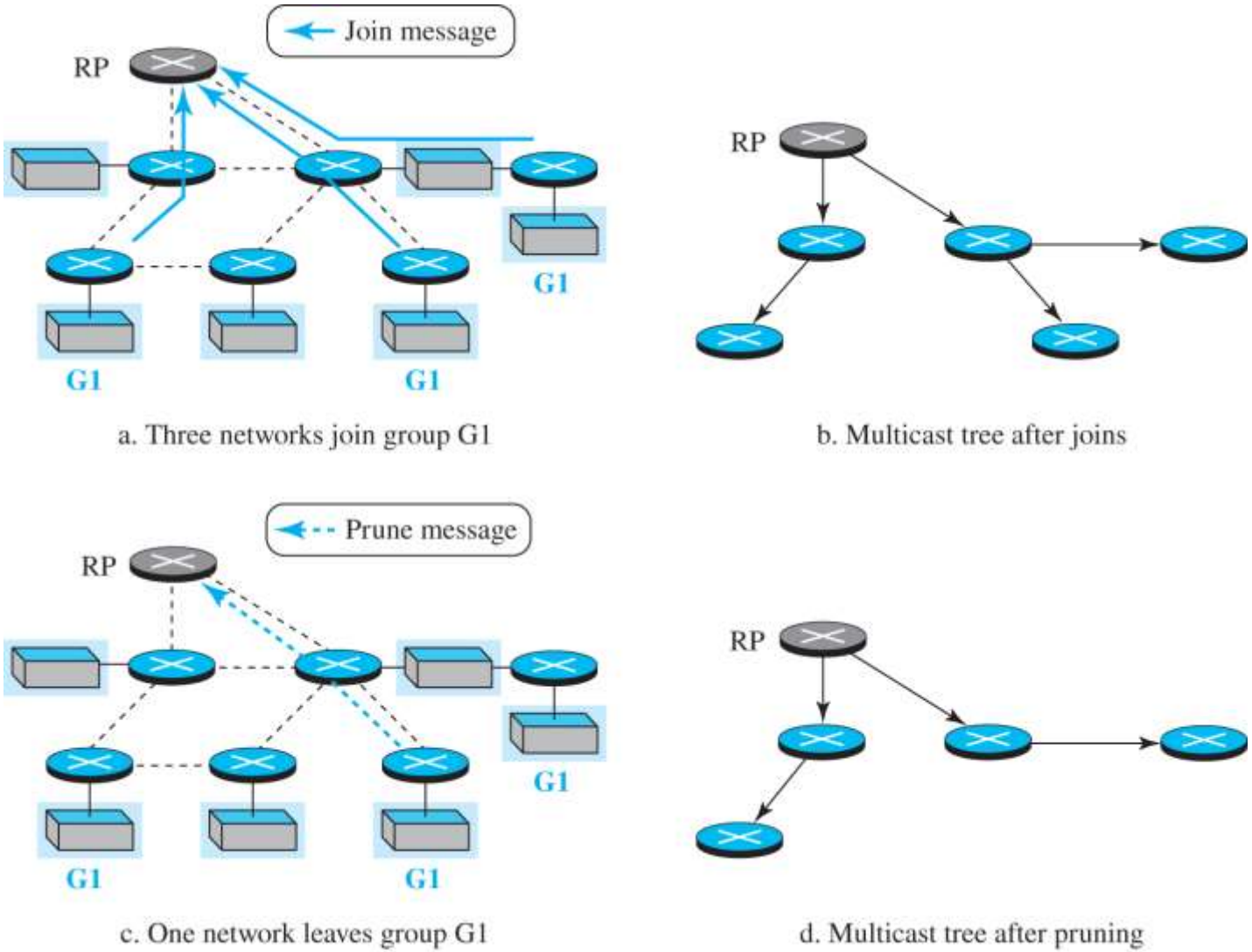
b. Second packet is multicast.

Access the text alternative for slide images.

Protocol Independent Multicast-Sparse Mode

When the number of routers with attached members is small relative to the number of routers in the internet, PIM works in the sparse mode and is called PIM-SM. In this environment, the use of a protocol that broadcasts the packets until the tree is pruned is not justified; PIM-SM uses a group-shared tree approach to multicasting. The core router in PIM-SM is called the rendezvous point (RP). Multicast communication is achieved in two steps. Any router that has a multicast packet to send to a group of destinations first encapsulates the multicast packet in a unicast packet (tunneling) and sends it to the RP. The RP then decapsulates the unicast packet and sends the multicast packet to its destination.

Figure 8.38 Idea behind PIM-SM



Access the text alternative for slide images.

Because learning changes everything.®

www.mheducation.com