Computer Networks (Course Code - BTIT301N)

Dr. Rajat Saxena

Shri Vaishnav Vidhyapeeth Viswavidyalaya, Indore

rajatsaxena@svvv.edu.in

November 24, 2022

Overview

1. Introduction to Network Layer
   - Need
   - Network Layer Services
   - Design Issues
   - Routing Algorithm
   - Types of Routing Algorithm
   - IPv4
   - IPv6
   - Classful Addressing
   - Classless Addressing
   - Subnetting
   - Supernetting

Part 1

Need Of Network Layer

1. The Network Layer is the third layer of the TCP/IP Protocol Suite.
2. The network layer in the TCP/IP protocol suite is responsible for the host-to-host delivery of datagrams.
3. It handles the service requests from the transport layer and further forwards the service request to the data link layer.
4. The network layer translates the logical addresses into physical addresses
5. It determines the route from the source to the destination and also manages the traffic problems such as switching, routing and controls the congestion of data packets.
6. The main role of the network layer is to move the packets from sending host to the receiving host.

Main Functions of Network Layer

1. **Routing:** When a packet reaches the router's input link, the router will move the packets to the router's output link. For example, a packet from S1 to R1 must be forwarded to the next router on the path to S2.
2. **Logical Addressing:** The data link layer implements the physical addressing and network layer implements the logical addressing. Logical addressing is also used to distinguish between source and destination system. The network layer adds a header to the packet which includes the logical addresses of both the sender and the receiver.
3. **Internetworking:** This is the main role of the network layer that it provides the logical connection between different types of networks.
4. **Fragmentation:** The fragmentation is a process of breaking the packets into the smallest individual data units that travel through different networks.

Network Layer : Services Provided

1. Packetizing

1. The first duty of the network layer is definitely packetizing: encapsulating the payload (data received from upper layer) in a network-layer packet at the source and decapsulating the payload from the network-layer packet at the destination. In other words, one duty of the network layer is to carry a payload from the source to the destination without changing it or using it. The network layer is doing the service of a carrier such as the postal office, which is responsible for delivery of packages from a sender to a receiver without changing or using the contents.
2. The source host receives the payload from an upper-layer protocol, adds a header that contains the source and destination addresses and some other information that is required by the network-layer protocol (as discussed later) and delivers the packet to the data-link layer. The source is not allowed to change the content of the payload unless it is too large for delivery and needs to be fragmented.

Network Layer : Services Provided

1. Packetizing

1. The destination host receives the network-layer packet from its data-link layer, decapsulates the packet, and delivers the payload to the corresponding upper-layer protocol. If the packet is fragmented at the source or at routers along the path, the network layer is responsible for waiting until all fragments arrive, reassembling them, and delivering them to the upper-layer protocol.
2. The routers in the path are not allowed to decapsulate the packets they received unless the packets need to be fragmented. The routers are not allowed to change source and destination addresses either. They just inspect the addresses for the purpose of forwarding the packet to the next network on the path. However, if a packet is fragmented, the header needs to be copied to all fragments and some changes are needed.

Network Layer : Services Provided

Other duties of the network layer are routing and forwarding, which are directly related to each other.

2. Routing

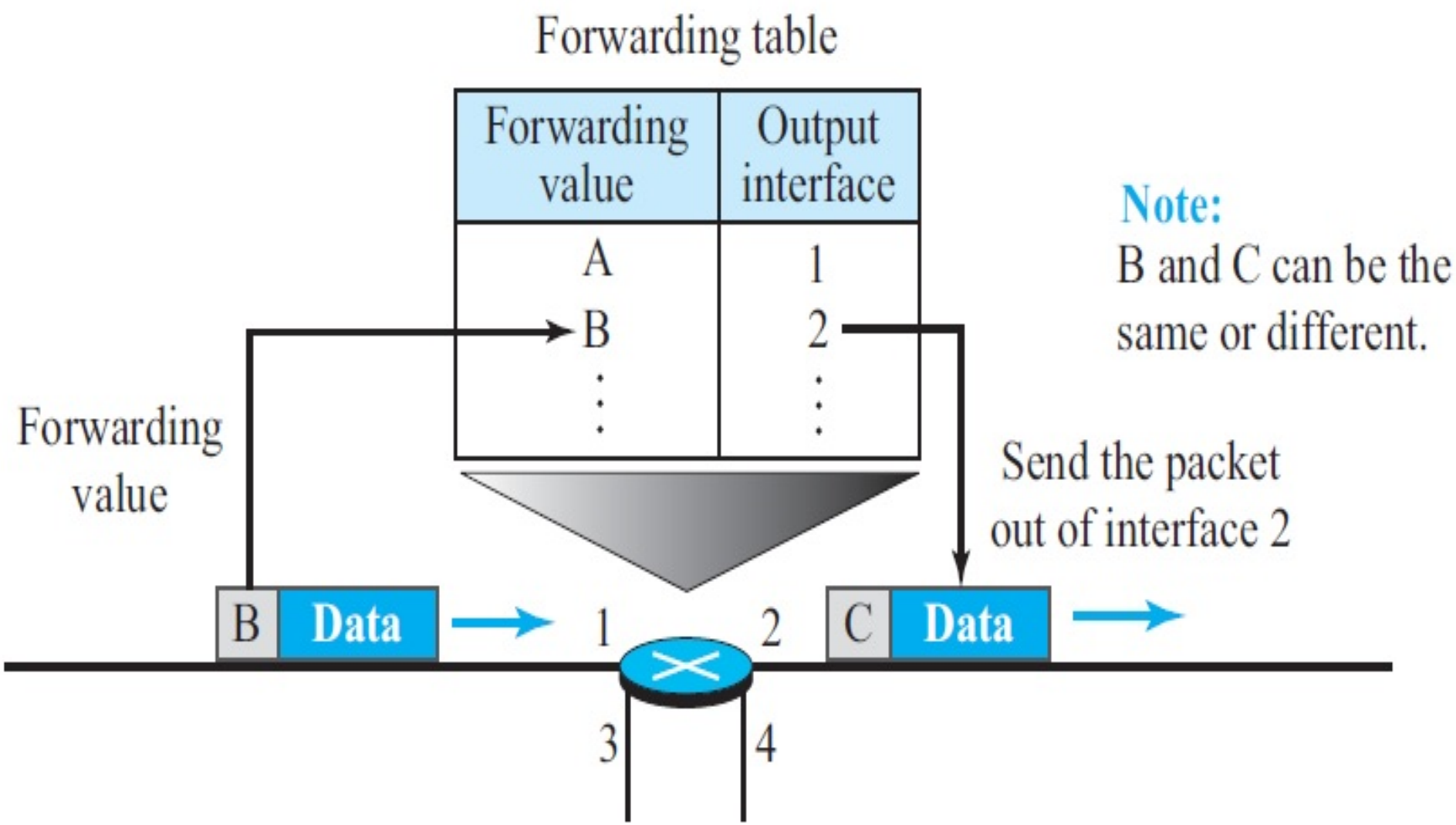
1. The network layer is responsible for routing the packet from its source to the destination. A physical network is a combination of networks (LANs and WANs) and routers that connect them. This means that there is more than one route from the source to the destination.
2. The network layer is responsible for finding the best one among these possible routes. The network layer needs to have some specific strategies for defining the best route. In the Internet today, this is done by running some routing protocols to help the routers coordinate their knowledge about the neighborhood and to come up with consistent tables to be used when a packet arrives.

Network Layer : Services Provided

3. Forwarding

1. If routing is applying strategies and running some routing protocols to create the decision-making tables for each router, forwarding can be defined as the action applied by each router when a packet arrives at one of its interfaces.
2. The decision-making table a router normally uses for applying this action is sometimes called the forwarding table and sometimes the routing table.
3. When a router receives a packet from one of its attached networks, it needs to forward the packet to another attached network (in unicast routing) or to some attached networks (in multicast routing).
4. To make this decision, the router uses a piece of information in the packet header, which can be the destination address or a label, to find the corresponding output interface number in the forwarding table. Figure shows the idea of the forwarding process in a router.

3. Forwarding Process



Network Layer : Services Provided

4. Error Control

1. Although error control also can be implemented in the network layer, the designers of the network layer in the Internet ignored this issue for the data being carried by the network layer. One reason for this decision is the fact that the packet in the network layer may be fragmented at each router, which makes error checking at this layer inefficient.
2. The designers of the network layer, however, have added a checksum field to the datagram to control any corruption in the header, but not in the whole datagram. This checksum may prevent any changes or corruptions in the header of the datagram.
3. We need to mention that although the network layer in the Internet does not directly provide error control, the Internet uses an auxiliary protocol, ICMP, that provides some kind of error control if the datagram is discarded or has some unknown information in the header.

Network Layer : Services Provided

5. Flow Control

1. Flow control regulates the amount of data a source can send without overwhelming the receiver. If the upper layer at the source computer produces data faster than the upper layer at the destination computer can consume it, the receiver will be overwhelmed with data.
2. To control the flow of data, the receiver needs to send some feedback to the sender to inform the latter that it is overwhelmed with data.
3. The network layer in the Internet, however, does not directly provide any flow control. The datagrams are sent by the sender when they are ready, without any attention to the readiness of the receiver.

Network Layer : Services Provided

5. Flow Control

A few reasons for the lack of flow control in the design of the network layer can be mentioned.

1. First, since there is no error control in this layer, the job of the network layer at the receiver is so simple that it may rarely be overwhelmed.
2. Second, the upper layers that use the service of the network layer can implement buffers to receive data from the network layer as they are ready and do not have to consume the data as fast as it is received.
3. Third, flow control is provided for most of the upper-layer protocols that use the services of the network layer, so another level of flow control makes the network layer more complicated and the whole system less efficient.

Network Layer : Services Provided

6. Congestion Control

1. Congestion in the network layer is a situation in which too many datagrams are present in an area of the Internet.
2. Congestion may occur if the number of datagrams sent by source computers is beyond the capacity of the network or routers. In this situation, some routers may drop some of the datagrams.
3. However, as more datagrams are dropped, the situation may become worse because, due to the error control mechanism at the upper layers, the sender may send duplicates of the lost packets.
4. If the congestion continues, sometimes a situation may reach a point where the system collapses and no datagrams are delivered.

Network Layer : Services Provided

7. Quality of Service

1. As the Internet has allowed new applications such as multimedia communication (in particular real-time communication of audio and video), the quality of service (QoS) of the communication has become more and more important.
2. The Internet has thrived by providing better quality of service to support these applications
3. However, to keep the network layer untouched, these provisions are mostly implemented in the upper layer.

Network Layer : Services Provided

8. Security

1. Another issue related to communication at the network layer is security.
2. Security was not a concern when the Internet was originally designed because it was used by a small number of users at universities for research activities; other people had no access to the Internet. The network layer was designed with no security provision.
3. Today, however, security is a big concern. To provide security for a connectionless network layer, we need to have another virtual level, IPSec that changes the connectionless service to a connection-oriented service.

Design Issue: Packet Switching

- We infer from routing and forwarding that a kind of switching occurs at the network layer.
- A router, in fact, is a switch that creates a connection between an input port and an output port (or a set of output ports), just as an electrical switch connects the input to the output to let electricity flow.
- Although in data communication switching techniques are divided into two broad categories, circuit switching and packet switching, only packet switching is used at the network layer because the unit of data at this layer is a packet. Circuit switching is mostly used at the physical layer; the electrical switch mentioned earlier is a kind of circuit switch.

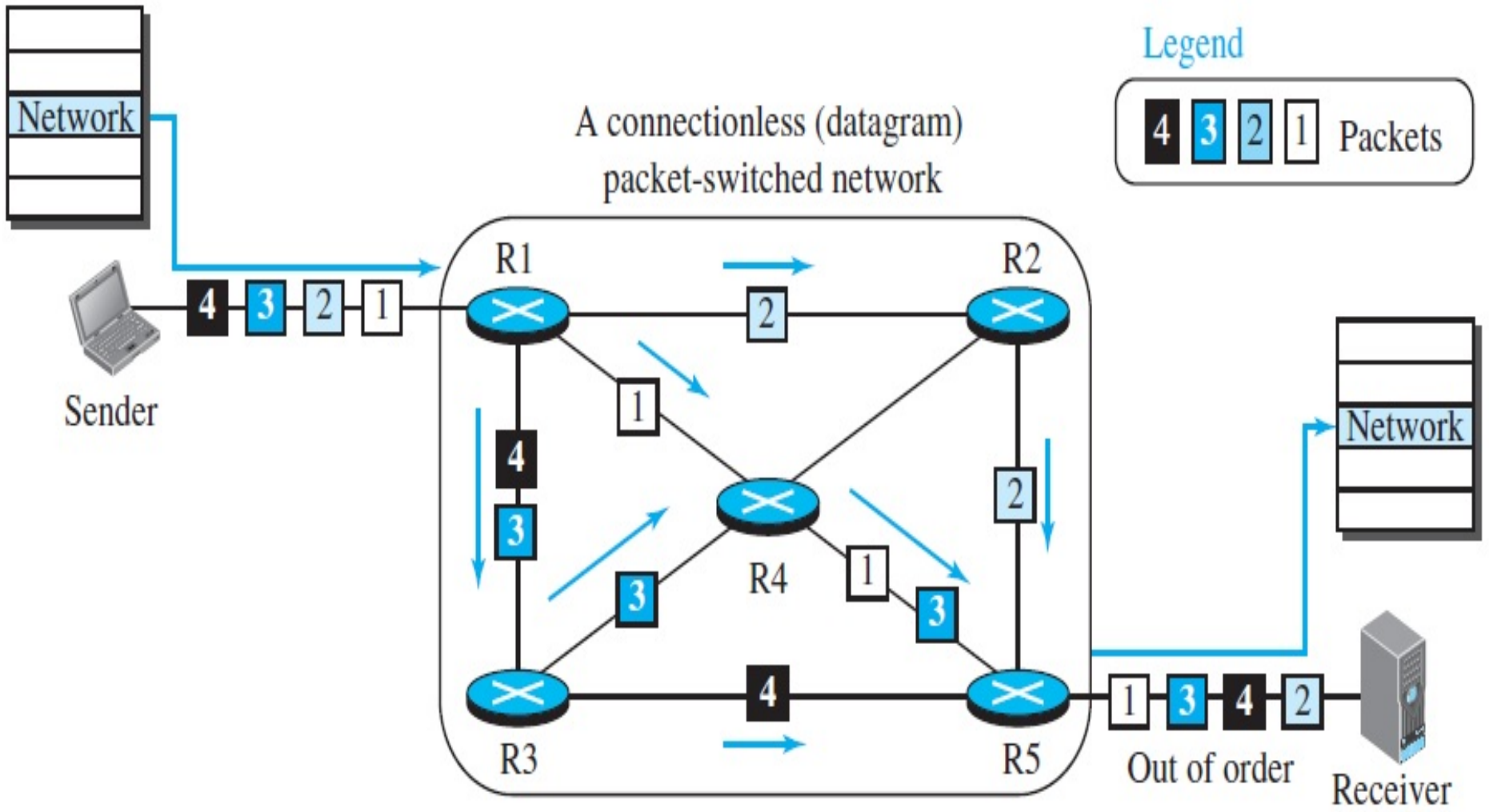
Design Issue : Packet Switching

- At the network layer, a message from the upper layer is divided into manageable packets and each packet is sent through the network.
- The source of the message sends the packets one by one; the destination of the message receives the packets one by one.
- The destination waits for all packets belonging to the same message to arrive before delivering the message to the upper layer.
- The connecting devices in a packet-switched network still need to decide how to route the packets to the final destination.
- Today, a packet-switched network can use two different approaches to route the packets: the datagram approach and the virtual circuit approach.

Datagram Approach: Connectionless Service

- When the Internet started, to make it simple, the network layer was designed to provide a connectionless service in which the network-layer protocol treats each packet independently, with each packet having no relationship to any other packet.
- The idea was that the network layer is only responsible for delivery of packets from the source to the destination. In this approach, the packets in a message may or may not travel the same path to their destination. Figure 18.3 shows the idea.
- In the datagram approach, the forwarding decision is based on the destination address of the packet.

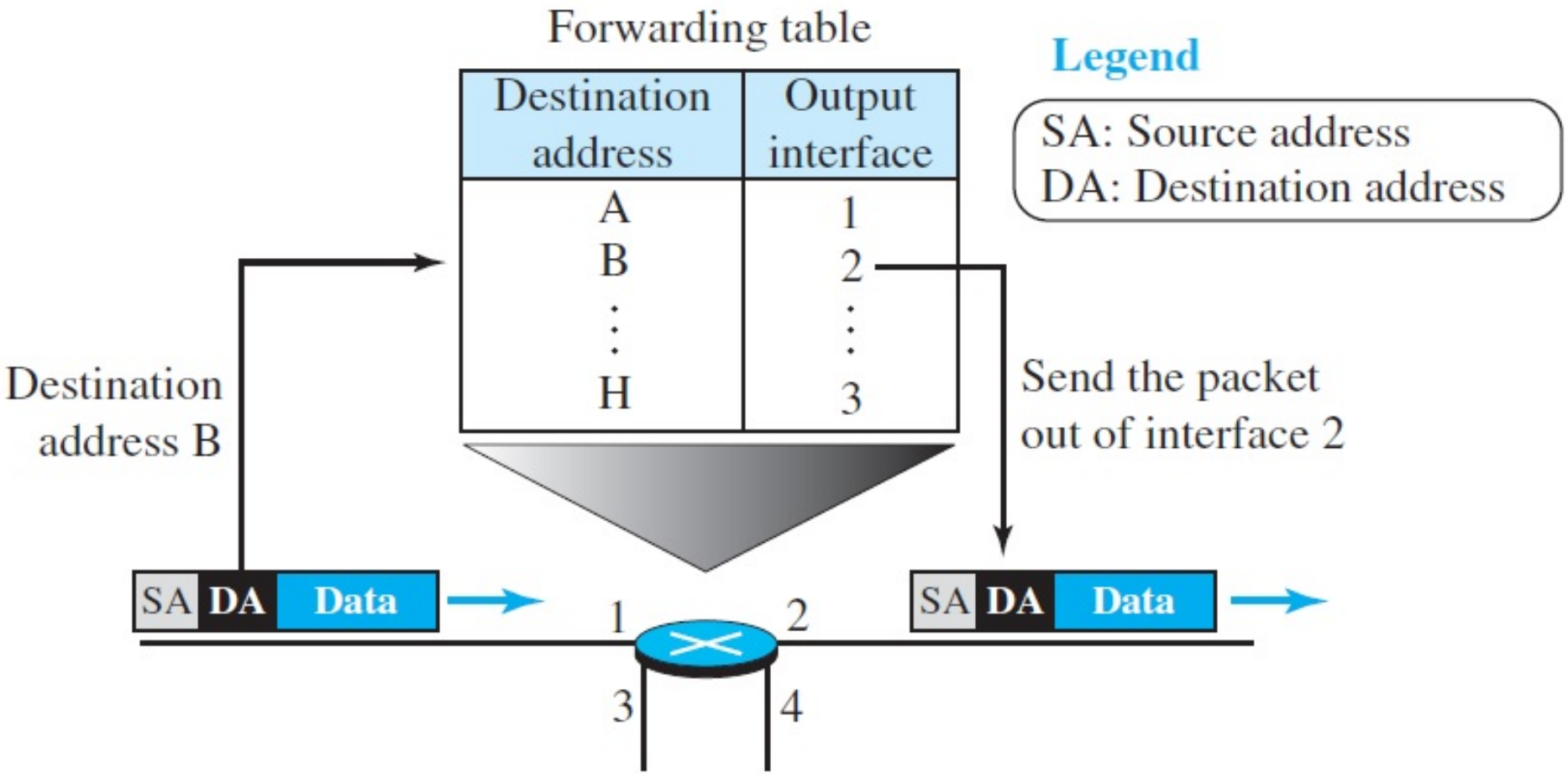
A connectionless packet-switched network



Datagram Approach: Connectionless Service

- When the network layer provides a connectionless service, each packet traveling in the Internet is an independent entity; there is no relationship between packets belonging to the same message. The switches in this type of network are called routers. A packet belonging to a message may be followed by a packet belonging to the same message or to a different message. A packet may be followed by a packet coming from the same or from a different source.
- Each packet is routed based on the information contained in its header: source and destination addresses. The destination address defines where it should go; the source address defines where it comes from. The router in this case routes the packet based only on the destination address. The source address may be used to send an error message to the source if the packet is discarded.
- Figure shows the forwarding process in a router in this case. We have used symbolic addresses such as A and B.

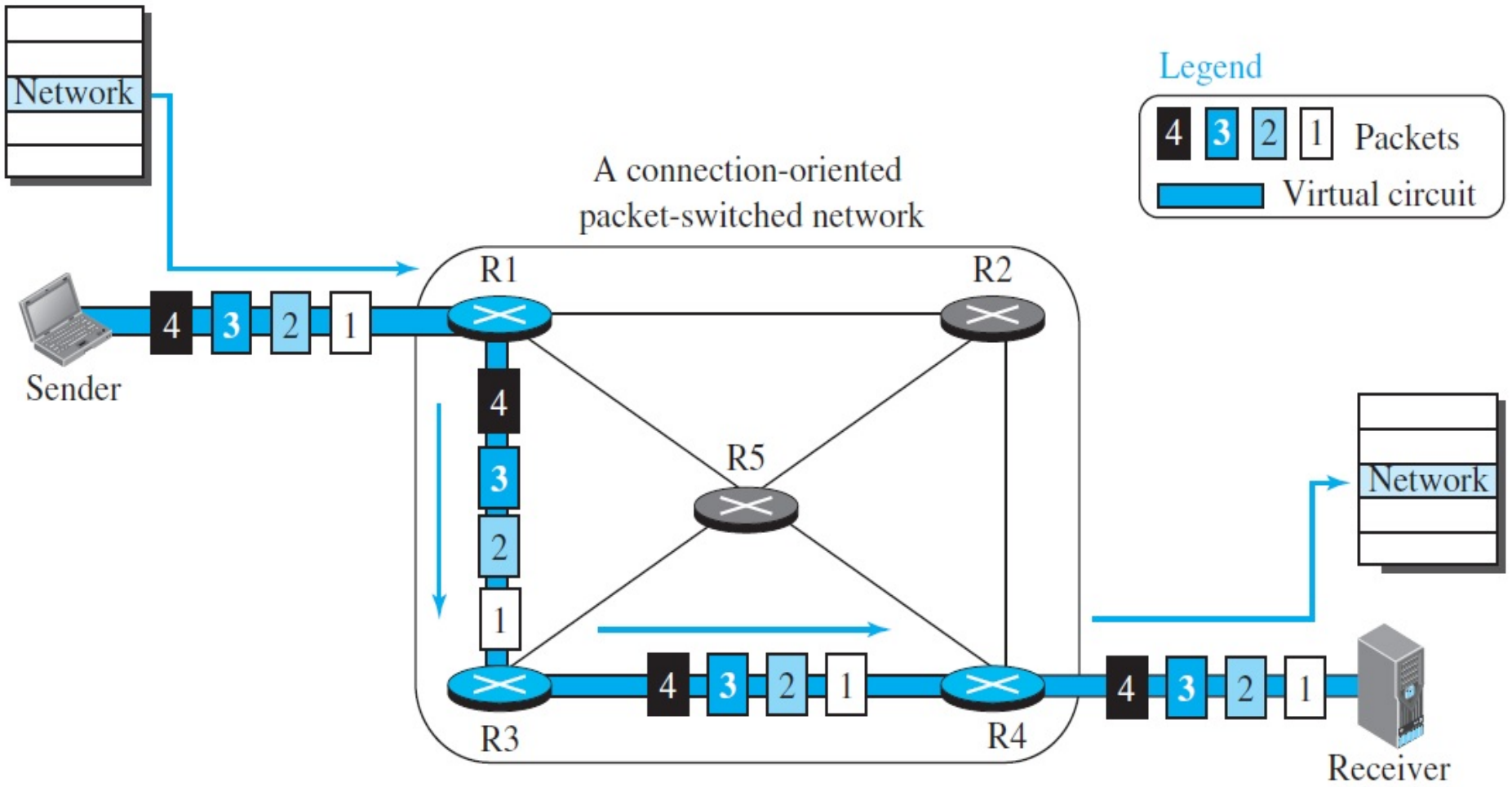
Forwarding process in a router when used in a connectionless network



Virtual-Circuit Approach: Connection-Oriented Service

- In a connection-oriented service (also called virtual-circuit approach), there is a relationship between all packets belonging to a message. Before all datagrams in a message can be sent, a virtual connection should be set up to define the path for the datagrams.
- After connection setup, the datagrams can all follow the same path. In this type of service, not only must the packet contain the source and destination addresses, it must also contain a flow label, a virtual circuit identifier that defines the virtual path the packet should follow.
- Shortly, we will show how this flow label is determined, but for the moment, we assume that the packet carries this label. Although it looks as though the use of the label may make the source and destination addresses unnecessary during the data transfer phase, parts of the Internet at the network layer still keep these addresses.
- One reason is that part of the packet path may still be using the connectionless service. Another reason is that the protocol at the network layer is designed with these addresses, and it may take a while before they can be changed.

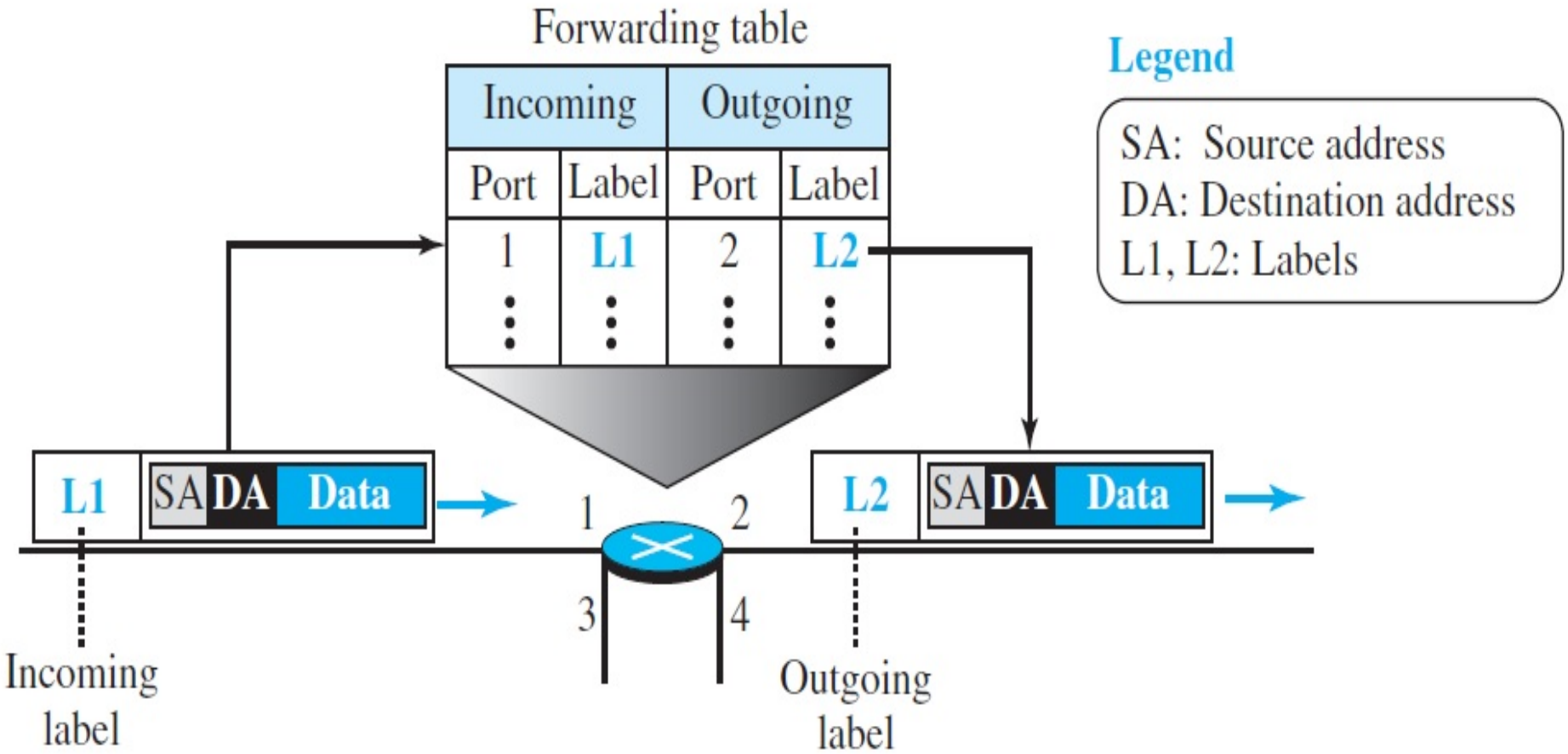
A virtual-circuit packet-switched network



A virtual-circuit packet-switched network

- Each packet is forwarded based on the label in the packet. To follow the idea of connection-oriented design to be used in the Internet, we assume that the packet has a label when it reaches the router.
- Figure shows the idea. In this case, the forwarding decision is based on the value of the label, or virtual circuit identifier, as it is sometimes called.
- To create a connection-oriented service, a three-phase process is used: setup, data transfer, and teardown.
- In the setup phase, the source and destination addresses of the sender and receiver are used to make table entries for the connection-oriented service.
- In the teardown phase, the source and destination inform the router to delete the corresponding entries.
- Data transfer occurs between these two phases.
- In the virtual-circuit approach, the forwarding decision is based on the label of the packet.

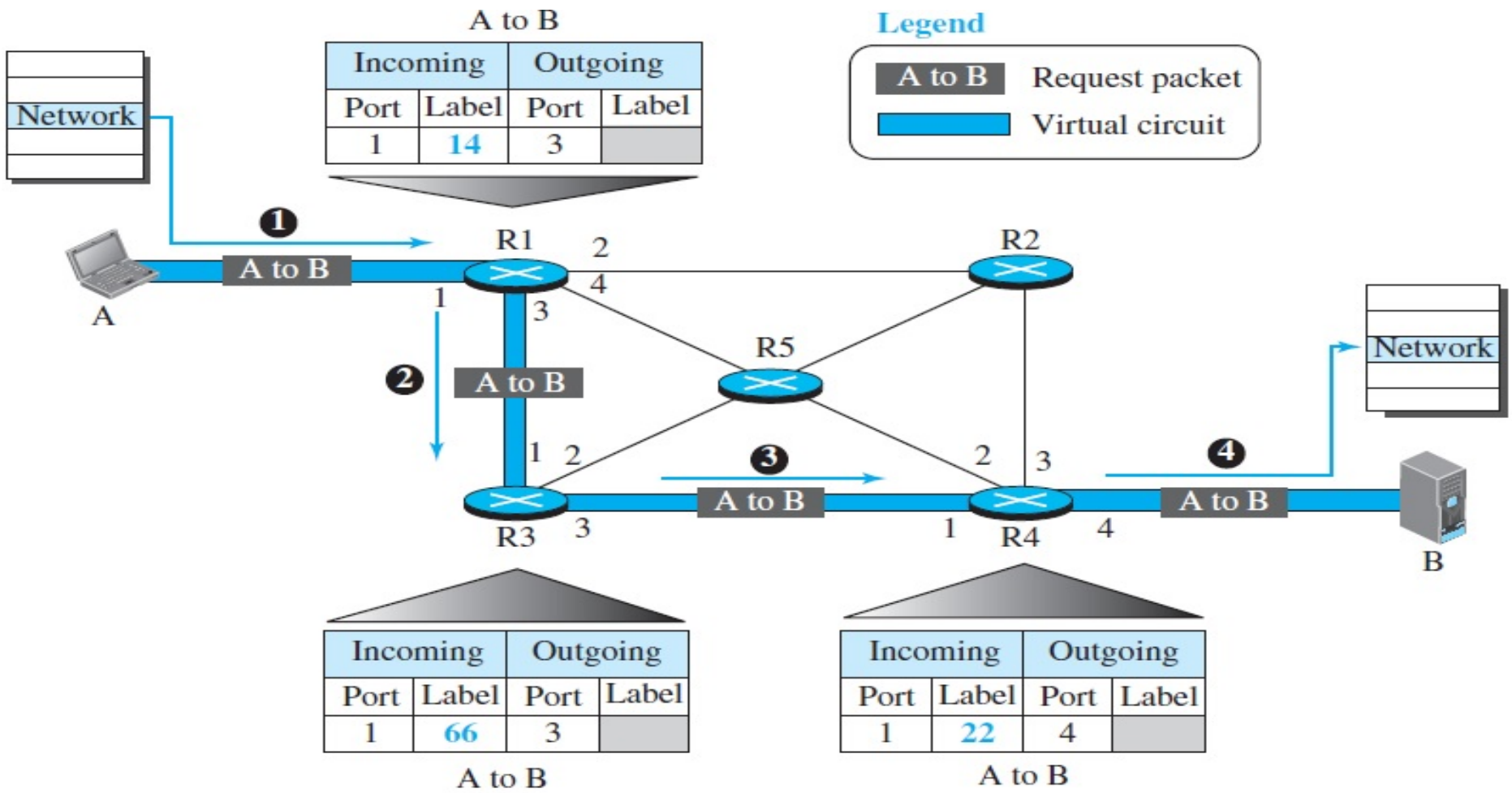
Forwarding process in a router when used in a virtual-circuit network



connection-oriented service: Setup Phase

- In the setup phase, a router creates an entry for a virtual circuit. For example, suppose source A needs to create a virtual circuit to destination B. Two auxiliary packets need to be exchanged between the sender and the receiver: the request packet and the acknowledgment packet.
- **Request packet:** A request packet is sent from the source to the destination. This auxiliary packet carries the source and destination addresses.
- Figure shows the process.

Sending request packet in a virtual-circuit network



Setup Phase : Request Packet

1. Source A sends a request packet to router R1.
2. Router R1 receives the request packet. It knows that a packet going from A to B goes out through port 3. For the moment, assume that it knows the output port. The router creates an entry in its table for this virtual circuit, but it is only able to fill three of the four columns. The router assigns the incoming port (1) and chooses an available incoming label (14) and the outgoing port (3). It does not yet know the outgoing label, which will be found during the acknowledgment step. The router then forwards the packet through port 3 to router R3.
3. Router R3 receives the setup request packet. The same events happen here as at router R1; three columns of the table are completed: in this case, incoming port (1), incoming label (66), and outgoing port (3).

Setup Phase : Request Packet

1. Router R4 receives the setup request packet. Again, three columns are completed: incoming port (1), incoming label (22), and outgoing port (4).
2. Destination B receives the setup packet, and if it is ready to receive packets from A, it assigns a label to the incoming packets that come from A, in this case 77, as shown in Figure. This label lets the destination know that the packets come from A, and not from other sources.

Setup Phase : Acknowledgment Packet

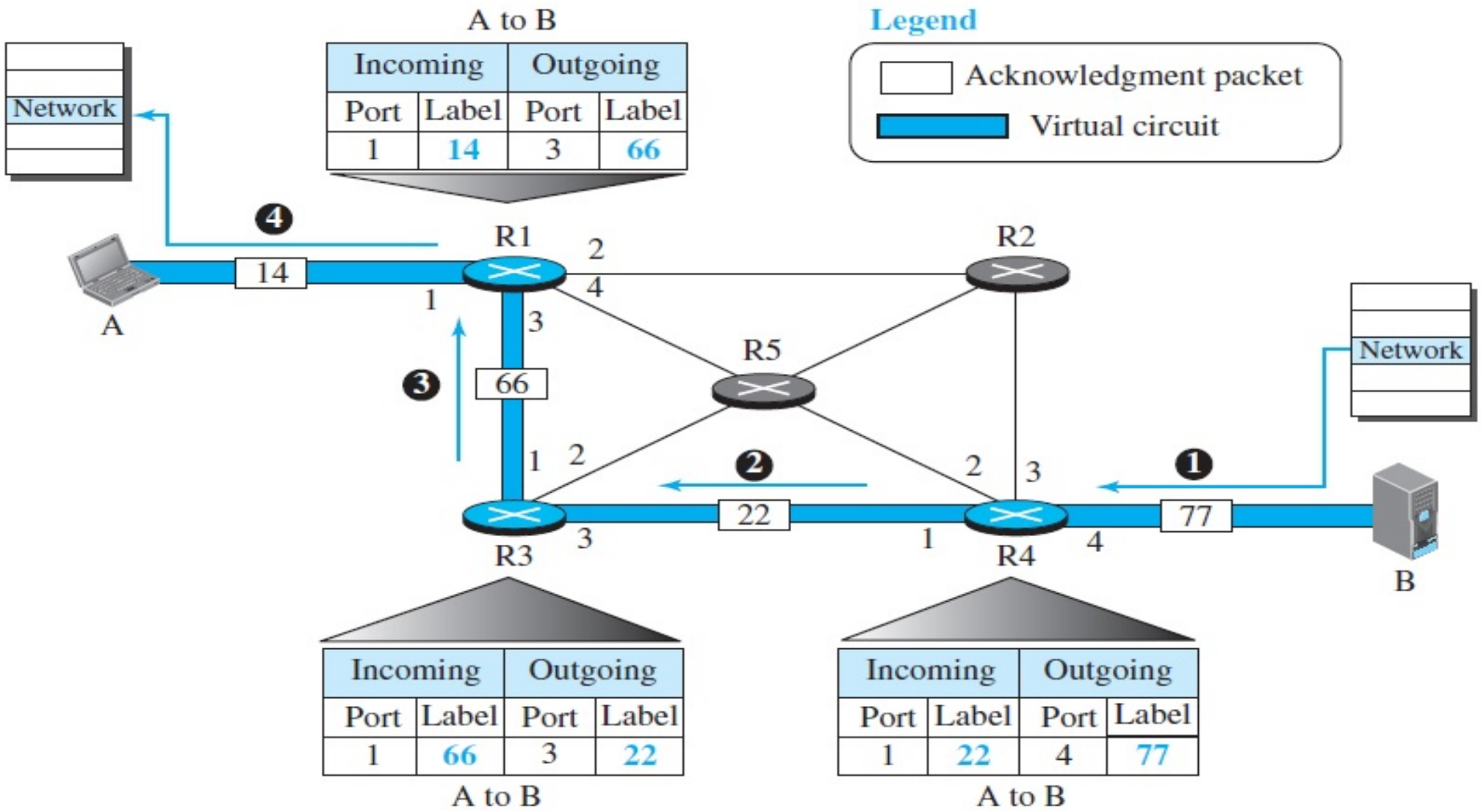
A special packet, called the acknowledgment packet, completes the entries in the switching tables. Figure shows the process.

- The destination sends an acknowledgment to router R4. The acknowledgment carries the global source and destination addresses so the router knows which entry in the table is to be completed. The packet also carries label 77, chosen by the destination as the incoming label for packets from A. Router R4 uses this label to complete the outgoing label column for this entry. Note that 77 is the incoming label for destination B, but the outgoing label for router R4.
- Router R4 sends an acknowledgment to router R3 that contains its incoming label in the table, chosen in the setup phase. Router R3 uses this as the outgoing label in the table.

Setup Phase : Acknowledgment Packet

- Router R3 sends an acknowledgment to router R1 that contains its incoming label in the table, chosen in the setup phase. Router R1 uses this as the outgoing label in the table.
- Finally router R1 sends an acknowledgment to source A that contains its incoming label in the table, chosen in the setup phase.
- The source uses this as the outgoing label for the data packets to be sent to destination B.

Sending acknowledgments in a virtual-circuit network



Network Layer Performance

The performance of a network can be measured in terms of delay, throughput, and packet loss. Congestion control is an issue that can improve the performance.

Delay

All of us expect instantaneous response from a network, but a packet, from its source to its destination, encounters delays. The delays in a network can be divided into four types: transmission delay, propagation delay, processing delay, and queuing delay.

Network Layer Performance: Delay

Transmission Delay

A source host or a router cannot send a packet instantaneously. A sender needs to put the bits in a packet on the line one by one. If the first bit of the packet is put on the line at time t_1 and the last bit is put on the line at time t_2, transmission delay of the packet is $(t_2 - t_1)$. Definitely, the transmission delay is longer for a longer packet and shorter if the sender can transmit faster. In other words, the transmission delay is

$$Delay_{tr} = (Packet\ length)/(Transmission\ rate). \quad (1)$$

For example, in a Fast Ethernet LAN with the transmission rate of 100 million bits per second and a packet of 10,000 bits, it takes (10,000)/(100,000,000) or 100 microseconds for all bits of the packet to be put on the line.

Network Layer Performance: Delay

Propagation Delay

Propagation delay is the time it takes for a bit to travel from point A to point B in the transmission media. The propagation delay for a packet-switched network depends on the propagation delay of each network (LAN or WAN). The propagation delay depends on the propagation speed of the media, which is 3×10^8 meters/second in a vacuum and normally much less in a wired medium; it also depends on the distance of the link. In other words, propagation delay is

$$Delay_{pg} = (Distance)/(Propagation\ Speed). \tag{2}$$

For example, if the distance of a cable link in a point-to-point WAN is 2000 meters and the propagation speed of the bits in the cable is 2×10^8 meters/second, then the propagation delay is 10 microseconds.

Network Layer Performance: Delay

Processing Delay

The processing delay is the time required for a router or a destination host to receive a packet from its input port, remove the header, perform an error detection procedure, and deliver the packet to the output port (in the case of a router) or deliver the packet to the upper-layer protocol (in the case of the destination host). The processing delay may be different for each packet, but normally is calculated as an average.

$$Delay_{pr} = Time\ required\ to\ process\ a\ packet\ in\ a\ router\ or\ a\ destination\ host. \quad (3)$$

Network Layer Performance: Delay

Queuing Delay

Queuing delay can normally happen in a router. A router has an input queue connected to each of its input ports to store packets waiting to be processed; the router also has an output queue connected to each of its output ports to store packets waiting to be transmitted. The queuing delay for a packet in a router is measured as the time a packet waits in the input queue and output queue of a router. We can compare the situation with a busy airport. Some planes may need to wait to get the landing band (input delay); some planes may need to wait to get the departure band (output delay).

$$Delay_{qu} = \textit{The time a packet waits in input and output queues in a router}. \quad (4)$$

Network Layer Performance: Delay

Total Delay

Assuming equal delays for the sender, routers, and receiver, the total delay (source-to destination delay) a packet encounters can be calculated if we know the number of routers, n, in the whole path.

$$Total\ delay = (n+1)\ (Delay_{tr} + Delay_{pg} + Delay_{pr}) + (n)(Delay_{qu}) \quad (5)$$

Note that if we have n routers, we have (n + 1) links. Therefore, we have (n + 1) transmission delays related to n routers and the source, (n + 1) propagation delays related to (n + 1) links, (n + 1) processing delays related to n routers and the destination, and only n queuing delays related to n routers.

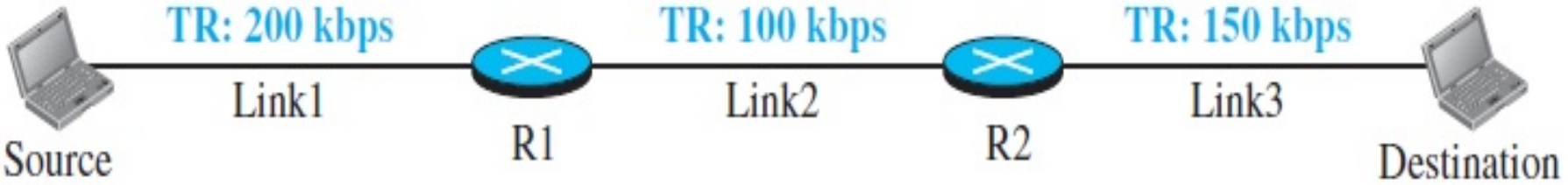
Network Layer Performance: Throughput

- Throughput at any point in a network is defined as the number of bits passing through the point in a second, which is actually the transmission rate of data at that point. In a path from source to destination, a packet may pass through several links (networks), each with a different transmission rate.
- Assume that we have three links, each with a different transmission rate, as shown in Figure. In this figure, the data can flow at the rate of 200 kbps in Link1. However, when the data arrives at router R1, it cannot pass at this rate. Data needs to be queued at the router and sent at 100 kbps. When data arrives at router R2, it could be sent at the rate of 150 kbps, but there is not enough data to be sent. In other words, the average rate of the data flow in Link3 is also 100 kbps. We can conclude that the average data rate for this path is 100 kbps, the minimum of the three different data rates. The figure also shows that we can simulate the behavior of each link with pipes of different sizes; the average throughput is determined by the bottleneck, the pipe with the smallest diameter.

Network Layer Performance: Throughput

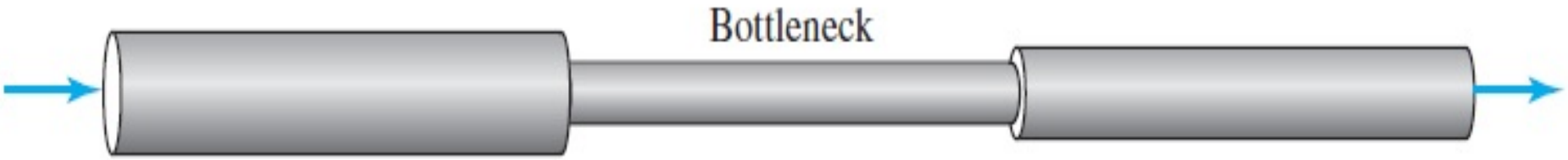
In general, in a path with n links in series, we have

$$Throughput = minimum\{TR_1, TR_2, ...TR_n\}. \quad (6)$$



a. A path through three links

TR: Transmission rate

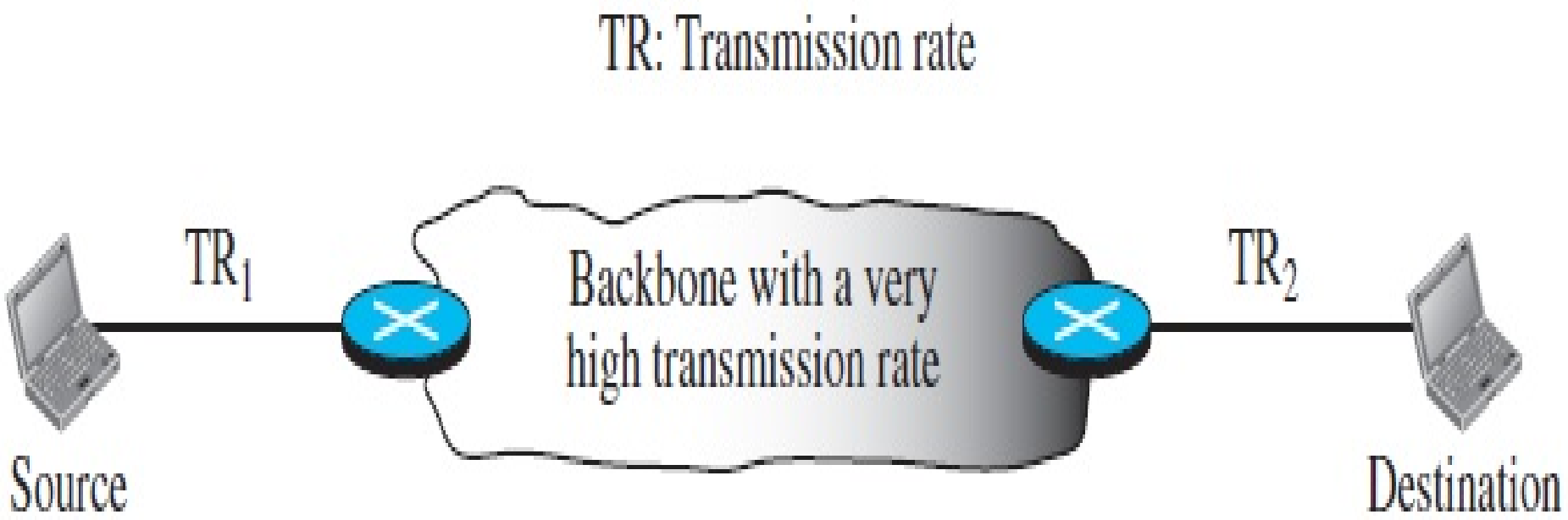


b. Simulation using pipes

Case Study 1 : Throughput

- The Internet backbone has a very high transmission rate, in the range of gigabits per second. This means that the throughput is normally defined as the minimum transmission rate of the two access links that connect the source and destination to the backbone.
- Figure shows this situation, in which the throughput is the minimum of TR_1 and TR_2. For example, if a server connects to the Internet via a Fast Ethernet LAN with the data rate of 100 Mbps, but a user who wants to download a file connects to the Internet via a dial-up telephone line with the data rate of 40 kbps, the throughput is 40 kbps. The bottleneck is definitely the dial-up line.

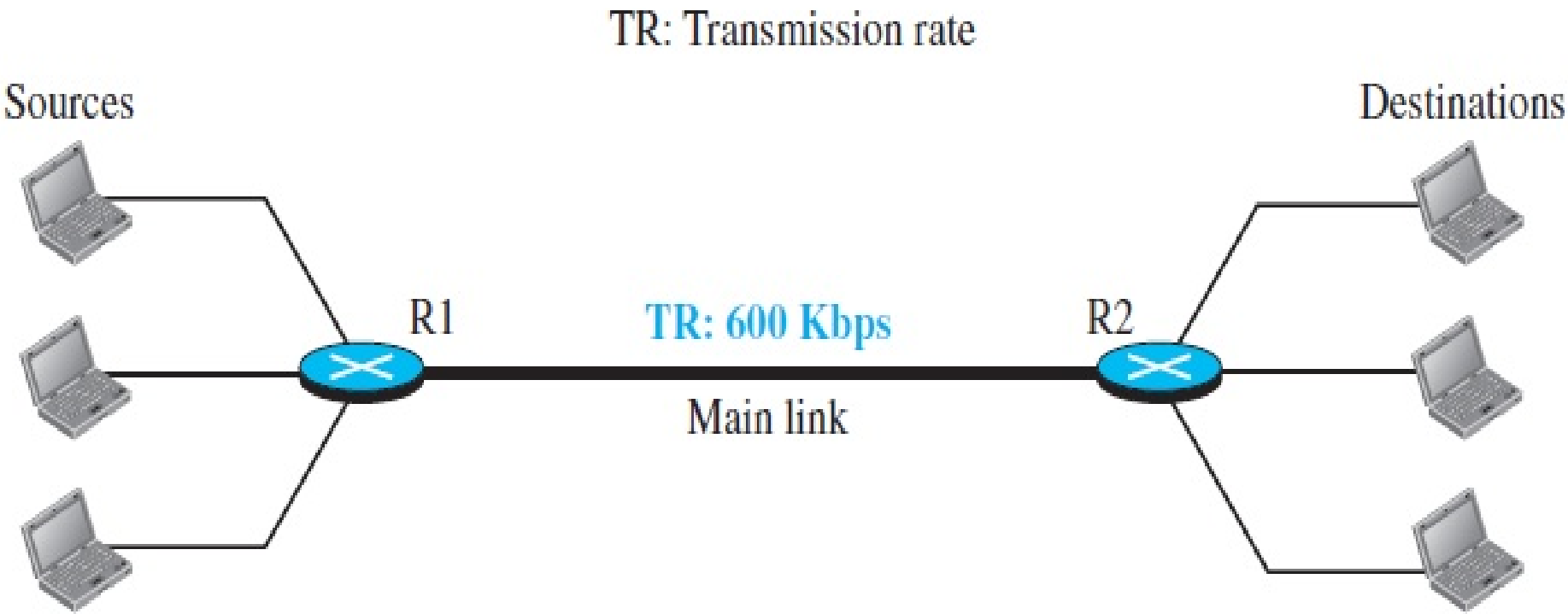
Case Study 1 : Throughput



Case Study 2 : Throughput

- The link between two routers is not always dedicated to one flow. A router may collect the flow from several sources or distribute the flow between several sources. In this case the transmission rate of the link between the two routers is actually shared between the flows and this should be considered when we calculate the throughput.
- For example, in Figure the transmission rate of the main link in the calculation of the throughput is only 200 kbps because the link is shared between three paths.

Case Study 2 : Throughput



Network Layer Performance: Packet Loss

- Another issue that severely affects the performance of communication is the number of packets lost during transmission.
- When a router receives a packet while processing another packet, the received packet needs to be stored in the input buffer waiting for its turn. A router, however, has an input buffer with a limited size. A time may come when the buffer is full and the next packet needs to be dropped.
- The effect of packet loss on the Internet network layer is that the packet needs to be resent, which in turn may create overflow and cause more packet loss.

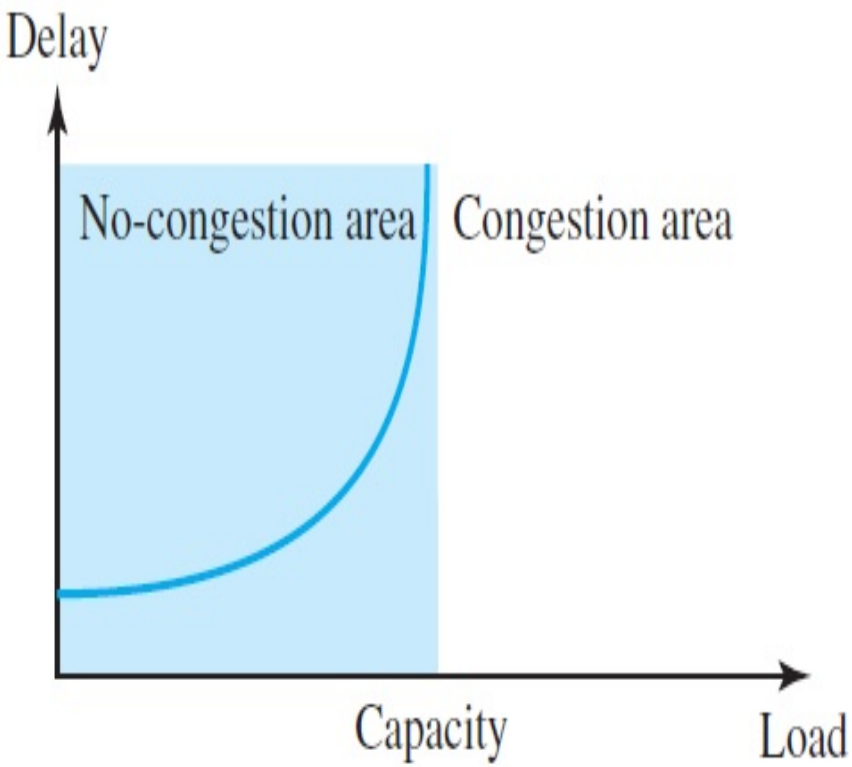
Congestion Control

- Congestion control is a mechanism for improving performance. Congestion at the network layer is related to two issues, throughput and delay. Figure shows these two performance measures as functions of load.
- When the load is much less than the capacity of the network, the delay is at a minimum. This minimum delay is composed of propagation delay and processing delay, both of which are negligible. However, when the load reaches the network capacity, the delay increases sharply because we now need to add the queuing delay to the total delay. Note that the delay becomes infinite when the load is greater than the capacity.

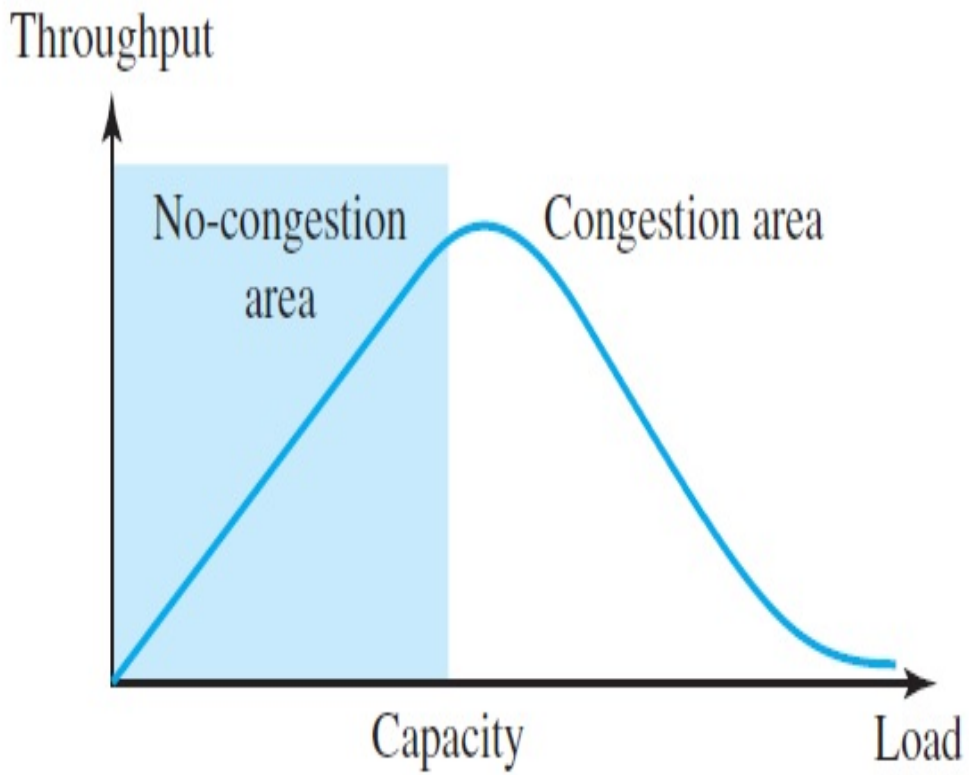
Congestion Control

- When the load is below the capacity of the network, the throughput increases proportionally with the load. We expect the throughput to remain constant after the load reaches the capacity, but instead the throughput declines sharply. The reason is the discarding of packets by the routers. When the load exceeds the capacity, the queues become full and the routers have to discard some packets. Discarding packets does not reduce the number of packets in the network because the sources retransmit the packets, using time-out mechanisms, when the packets do not reach the destinations.
- Congestion control refers to techniques and mechanisms that can either prevent congestion before it happens or remove congestion after it has happened. In general, we can divide congestion control mechanisms into two broad categories: open-loop congestion control (prevention) and closed-loop congestion control (removal).

Congestion Control



a. Delay as a function of load



b. Throughput as a function of load

Open-Loop Congestion Control

In open-loop congestion control, policies are applied to prevent congestion before it happens. In these mechanisms, congestion control is handled by either the source or the destination. We give a brief list of policies that can prevent congestion.

Retransmission Policy

Retransmission is sometimes unavoidable. If the sender feels that a sent packet is lost or corrupted, the packet needs to be retransmitted. Retransmission in general may increase congestion in the network. However, a good retransmission policy can prevent congestion. The retransmission policy and the retransmission timers must be designed to optimize efficiency and at the same time prevent congestion.

Part 2

IPV4 Addresses

- The identifier used in the IP layer of the TCP/IP protocol suite to identify the connection of each device to the Internet is called the Internet address or IP address.
- An IPv4 address is a 32-bit address that uniquely and universally defines the connection of a host or a router to the Internet. The IP address is the address of the connection, not the host or the router, because if the device is moved to another network, the IP address may be changed.
- IPv4 addresses are unique in the sense that each address defines one, and only one, connection to the Internet. If a device has two connections to the Internet, via two networks, it has two IPv4 addresses. IPv4 addresses are universal in the sense that the addressing system must be accepted by any host that wants to be connected to the Internet.

Address Space

- A protocol like IPv4 that defines addresses has an address space. An address space is the total number of addresses used by the protocol.
- If a protocol uses b bits to define an address, the address space is 2^b because each bit can have two different values (0 or 1).
- IPv4 uses 32-bit addresses, which means that the address space is 2^{32} or 4,294,967,296 (more than four billion). If there were no restrictions, more than 4 billion devices could be connected to the Internet.

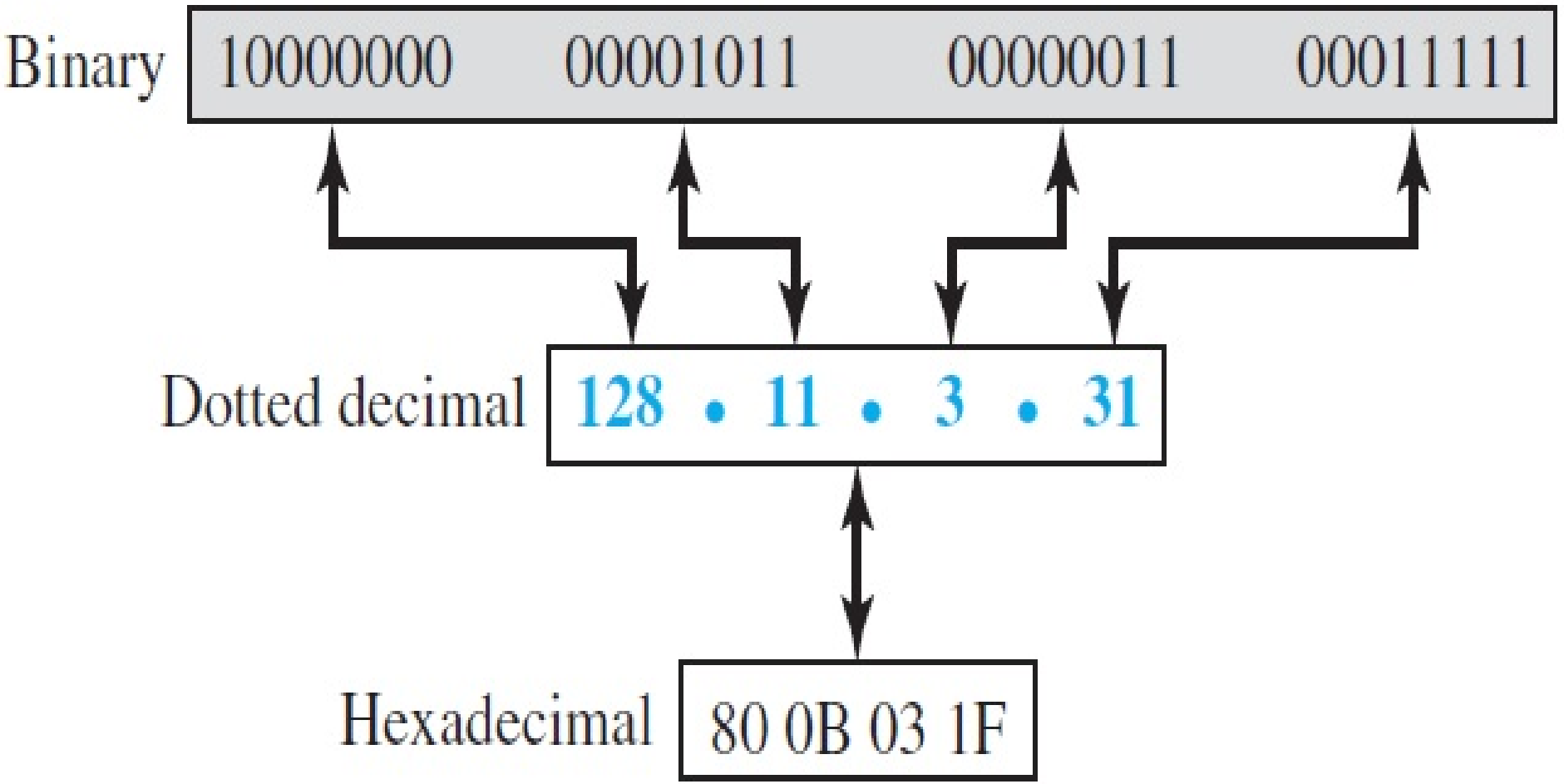
Notation

- There are three common notations to show an IPv4 address: binary notation (base 2), dotted-decimal notation (base 256), and hexadecimal notation (base 16).
- In binary notation, an IPv4 address is displayed as 32 bits. To make the address more readable, one or more spaces are usually inserted between each octet (8 bits). Each octet is often referred to as a byte.

Address Space : Notation

- To make the IPv4 address more compact and easier to read, it is usually written in decimal form with a decimal point (dot) separating the bytes. This format is referred to as dotted-decimal notation.
- Note that because each byte (octet) is only 8 bits, each number in the dotted-decimal notation is between 0 and 255.
- We sometimes see an IPv4 address in hexadecimal notation. Each hexadecimal digit is equivalent to four bits. This means that a 32-bit address has 8 hexadecimal digits. This notation is often used in network programming.
- Figure shows an IP address in the three discussed notations.

Three different notations in IPv4 addressing

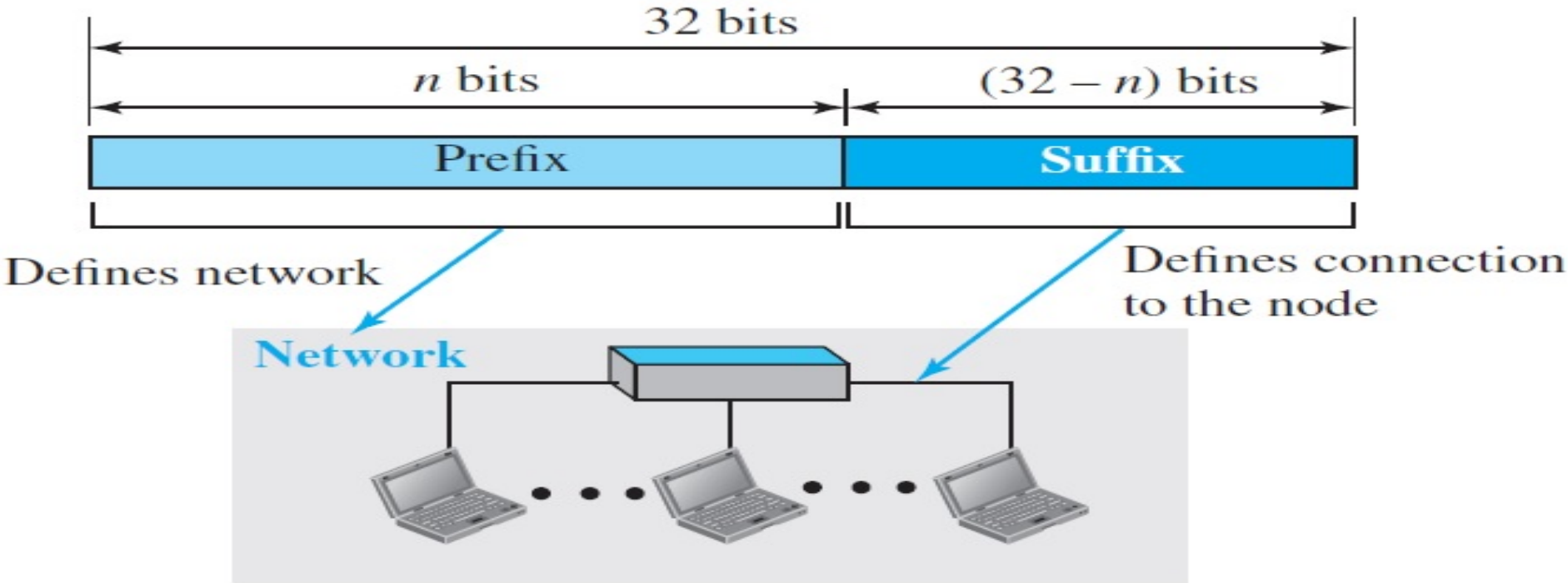


Hierarchy in Addressing

- In any communication network that involves delivery, such as a telephone network or a postal network, the addressing system is hierarchical.
- In a postal network, the postal address (mailing address) includes the country, state, city, street, house number, and the name of the mail recipient.
- Similarly, a telephone number is divided into the country code, area code, local exchange, and the connection.
- A 32-bit IPv4 address is also hierarchical, but divided only into two parts. The first part of the address, called the prefix, defines the network; the second part of the address, called the suffix, defines the node (connection of a device to the Internet).
- Figure shows the prefix and suffix of a 32-bit IPv4 address. The prefix length is n bits and the suffix length is (32 - n) bits.

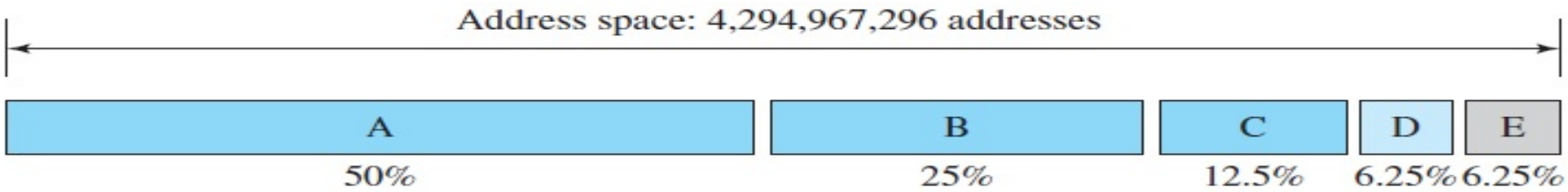
Hierarchy in Addressing

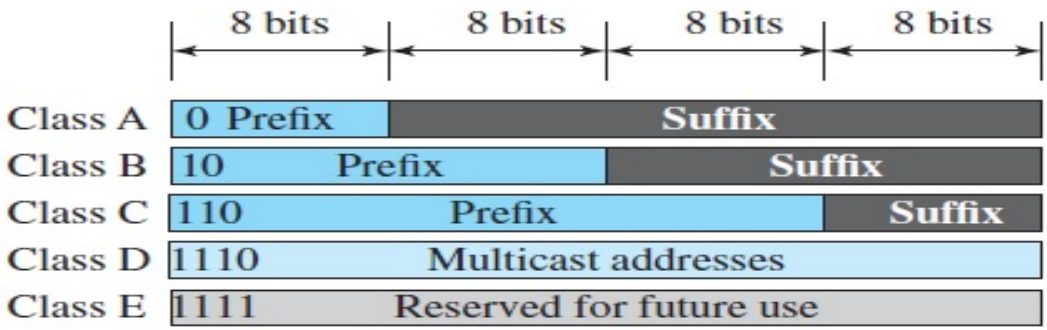
- A prefix can be fixed length or variable length. The network identifier in the IPv4 was first designed as a fixed-length prefix. This scheme, which is now obsolete, is referred to as classful addressing.
- The new scheme, which is referred to as classless addressing, uses a variable-length network prefix.



Classful Addressing

- When the Internet started, an IPv4 address was designed with a fixed-length prefix, but to accommodate both small and large networks, three fixed-length prefixes were designed instead of one (n = 8, n = 16, and n = 24).
- The whole address space was divided into five classes (class A, B, C, D, and E), as shown in Figure. This scheme is referred to as classful addressing.





Class	Prefixes	First byte
A	$n = 8$ bits	0 to 127
B	$n = 16$ bits	128 to 191
C	$n = 24$ bits	192 to 223
D	Not applicable	224 to 239
E	Not applicable	240 to 255

Classful Addressing

- In class A, the network length is 8 bits, but since the first bit, which is 0, defines the class, we can have only seven bits as the network identifier. This means there are only $2^7 = 128$ networks in the world that can have a class A address.
- In class B, the network length is 16 bits, but since the first two bits, which are $(10)_2$, define the class, we can have only 14 bits as the network identifier. This means there are only $2^{14} = 16{,}384$ networks in the world that can have a class B address.
- All addresses that start with $(110)_2$ belong to class C. In class C, the network length is 24 bits, but since three bits define the class, we can have only 21 bits as the network identifier. This means there are $2^{21} = 2{,}097{,}152$ networks in the world that can have a class C address.
- Class D is not divided into prefix and suffix. It is used for multicast addresses. All addresses that start with 1111 in binary belong to class E. As in Class D, Class E is not divided into prefix and suffix and is used as reserve.

Address Depletion

- classful addressing has become obsolete because of address depletion.
- Since the addresses were not distributed properly, the Internet was faced with the problem of the addresses being rapidly used up, resulting in no more addresses available for organizations and individuals that needed to be connected to the Internet.
- To understand the problem, let us think about class A. This class can be assigned to only 128 organizations in the world, but each organization needs to have a single network (seen by the rest of the world) with 16,777,216 nodes (computers in this single network). Since there may be only a few organizations that are this large, most of the addresses in this class were wasted (unused).
- Class B addresses were designed for midsize organizations, but many of the addresses in this class also remained unused. Class C addresses have a completely different flaw in design. The number of addresses that can be used in each network (256) was so small that most companies were not comfortable using a block in this address class. Class E addresses were almost never used, wasting the whole class.

Subnetting and Supernetting

- To alleviate address depletion, two strategies were proposed : subnetting and supernetting.
- In subnetting, a class A or class B block is divided into several subnets. Each subnet has a larger prefix length than the original network. For example, if a network in class A is divided into four subnets, each subnet has a prefix of $n_{sub} = 10$. At the same time, if all of the addresses in a network are not used, subnetting allows the addresses to be divided among several organizations. This idea did not work because most large organizations were not happy about dividing the block and giving some of the unused addresses to smaller organization.
- While subnetting was devised to divide a large block into smaller ones, supernetting was devised to combine several class C blocks into a larger block to be attractive to organizations that need more than the 256 addresses available in a class C block. This idea did not work either because it makes the routing of packets more difficult.

Advantage of Classful Addressing

classful addressing have one advantage: Given an address, we can easily find the class of the address and, since the prefix length for each class is fixed, we can find the prefix length immediately. In other words, the prefix length in classful addressing is inherent in the address; no extra information is needed to extract the prefix and the suffix.

Classless Addressing

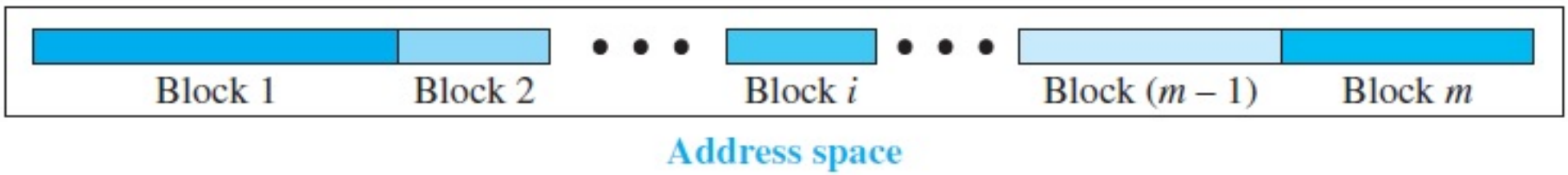
- Subnetting and supernetting in classful addressing did not really solve the address depletion problem.
- With the growth of the Internet, it was clear that a larger address space was needed as a long-term solution. The larger address space, however, requires that the length of IP addresses also be increased, which means the format of the IP packets needs to be changed.
- Although the long-range solution has already been devised and is called IPv6, a short-term solution was also devised to use the same address space but to change the distribution of addresses to provide a fair share to each organization.
- The short-term solution still uses IPv4 addresses, but it is called classless addressing. In other words, the class privilege was removed from the distribution to compensate for the address depletion.

Motivation for Classless Addressing.

- During the 1990s, Internet Service Providers (ISPs) came into prominence.
- An ISP is an organization that provides Internet access for individuals, small businesses, and midsize organizations that do not want to create an Internet site and become involved in providing Internet services (such as electronic mail) for their employees. An ISP can provide these services.
- An ISP is granted a large range of addresses and then subdivides the addresses (in groups of 1, 2, 4, 8, 16, and so on), giving a range of addresses to a household or a small business. The customers are connected via a dial-up modem, DSL, or cable modem to the ISP. However, each customer needs some IPv4 addresses.

Classless Addressing

- In 1996, the Internet authorities announced a new architecture called classless addressing. In classless addressing, variable-length blocks are used that belong to no classes. We can have a block of 1 address, 2 addresses, 4 addresses, 128 addresses, and so on.
- In classless addressing, the whole address space is divided into variable length blocks. The prefix in an address defines the block (network); the suffix defines the node (device).
- Theoretically, we can have a block of $2^0, 2^1, 2^2, ..., 2^{32}$ addresses. One of the restrictions is that the number of addresses in a block needs to be a power of 2. An organization can be granted one block of addresses. Figure shows the division of the whole address space into non-overlapping blocks.

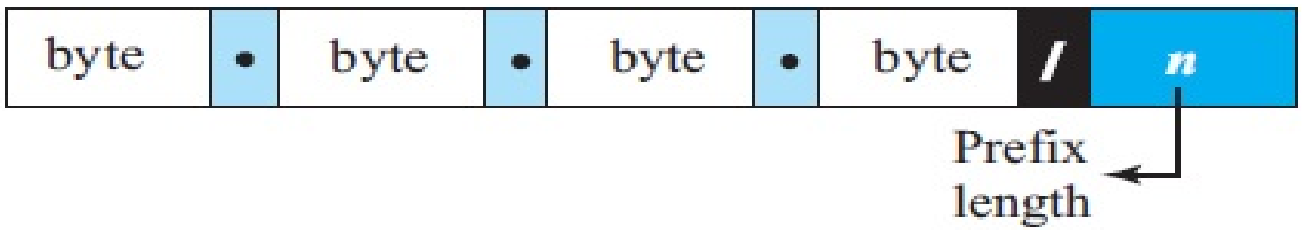


Classless Addressing

- Unlike classful addressing, the prefix length in classless addressing is variable. We can have a prefix length that ranges from 0 to 32. The size of the network is inversely proportional to the length of the prefix. A small prefix means a larger network; a large prefix means a smaller network.
- We need to emphasize that the idea of classless addressing can be easily applied to classful addressing. An address in class A can be thought of as a classless address in which the prefix length is 8. An address in class B can be thought of as a classless address in which the prefix is 16, and so on. In other words, classful addressing is a special case of classless addressing.

Classless Addressing: Prefix Length: Slash Notation

- Since the prefix length is not inherent in the classless addressing, we need to separately give the length of the prefix.
- In this case, the prefix length, n, is added to the address, separated by a slash. The notation is informally referred to as slash notation and formally as classless interdomain routing or CIDR (pronounced cider) strategy.
- In other words, an address in classless addressing does not define the block or network to which the address belongs; we need to give the prefix length also.
- An address in classless addressing can then be represented as shown in Figure.

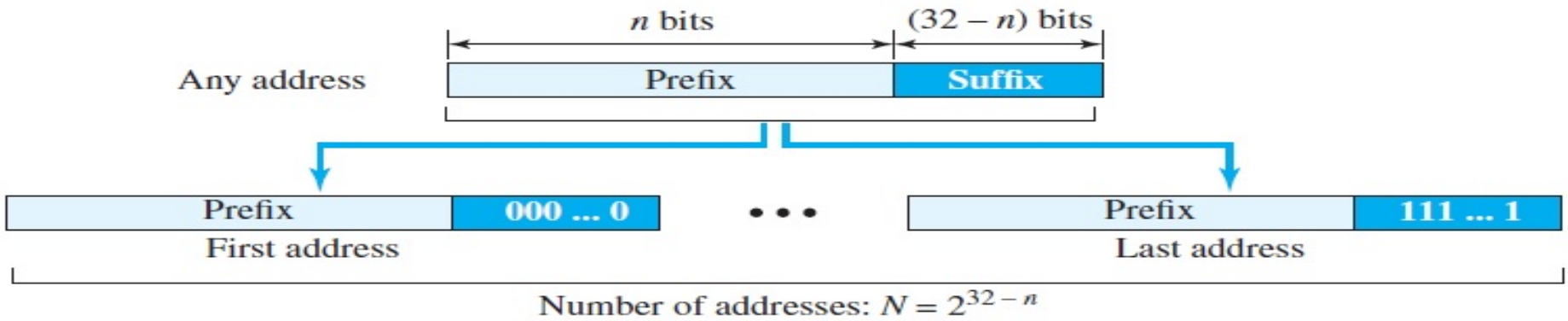


Examples:
12.24.76.8/8
23.14.67.92/12
220.8.24.255/25

Extracting Information from an Address

Given any address in the block, we normally like to know three pieces of information about the block to which the address belongs: the number of addresses, the first address in the block, and the last address. Since the value of prefix length, n, is given, we can easily find these three pieces of information, as shown in Figure.

1. The number of addresses in the block is found as $N = 2^{32-n}$.
2. To find the first address, we keep the n leftmost bits and set the (32 - n) rightmost bits all to 0s.
3. To find the last address, we keep the n leftmost bits and set the (32 - n) rightmost bits all to 1s.



Example: Classless Addressing

- A classless address is given as 167.199.170.82/27. We can find the above three pieces of information as follows.
- The number of addresses in the network is $2^{32-n} = 2^5 = 32$ addresses.
- The first address can be found by keeping the first 27 bits and changing the rest of the bits to 0s.
  Address: 167.199.170.82/27 10100111 11000111 10101010 01010010
  First address: 167.199.170.64/27 10100111 11000111 10101010 010**00000**
- The last address can be found by keeping the first 27 bits and changing the rest of the bits to 1s.
  Address: 167.199.170.82/27 10100111 11000111 10101010 01011111
  Last address: 167.199.170.95/27 10100111 11000111 10101010 010**11111**

Classless Addressing: Address Mask

- Another way to find the first and last addresses in the block is to use the address mask.
- The address mask is a 32-bit number in which the n leftmost bits are set to 1s and the rest of the bits (32 n) are set to 0s.
- A computer can easily find the address mask because it is the complement of (2^{32-n} 1).
- The reason for defining a mask in this way is that it can be used by a computer program to extract the information in a block, using the three bit-wise operations NOT, AND, and OR.
- The number of addresses in the block N = NOT (mask) + 1.
- The first address in the block = (Any address in the block) AND (mask).
- The last address in the block = (Any address in the block) OR [(NOT (mask)].

Example : Classless Addressing

- A classless address is given as 167.199.170.82/27.
- The mask in dotted-decimal notation is 255.255.255.224.
- The AND, OR, and NOT operations can be applied to individual bytes.
- Number of addresses in the block: N = NOT (mask) + 1 = 0.0.0.31 + 1 = 32 addresses.
- First address: First = (address) AND (mask) = 167.199.170.82
- Last address: Last = (address) OR (NOT mask) = 167.199.170.255

Example: Classless Addressing

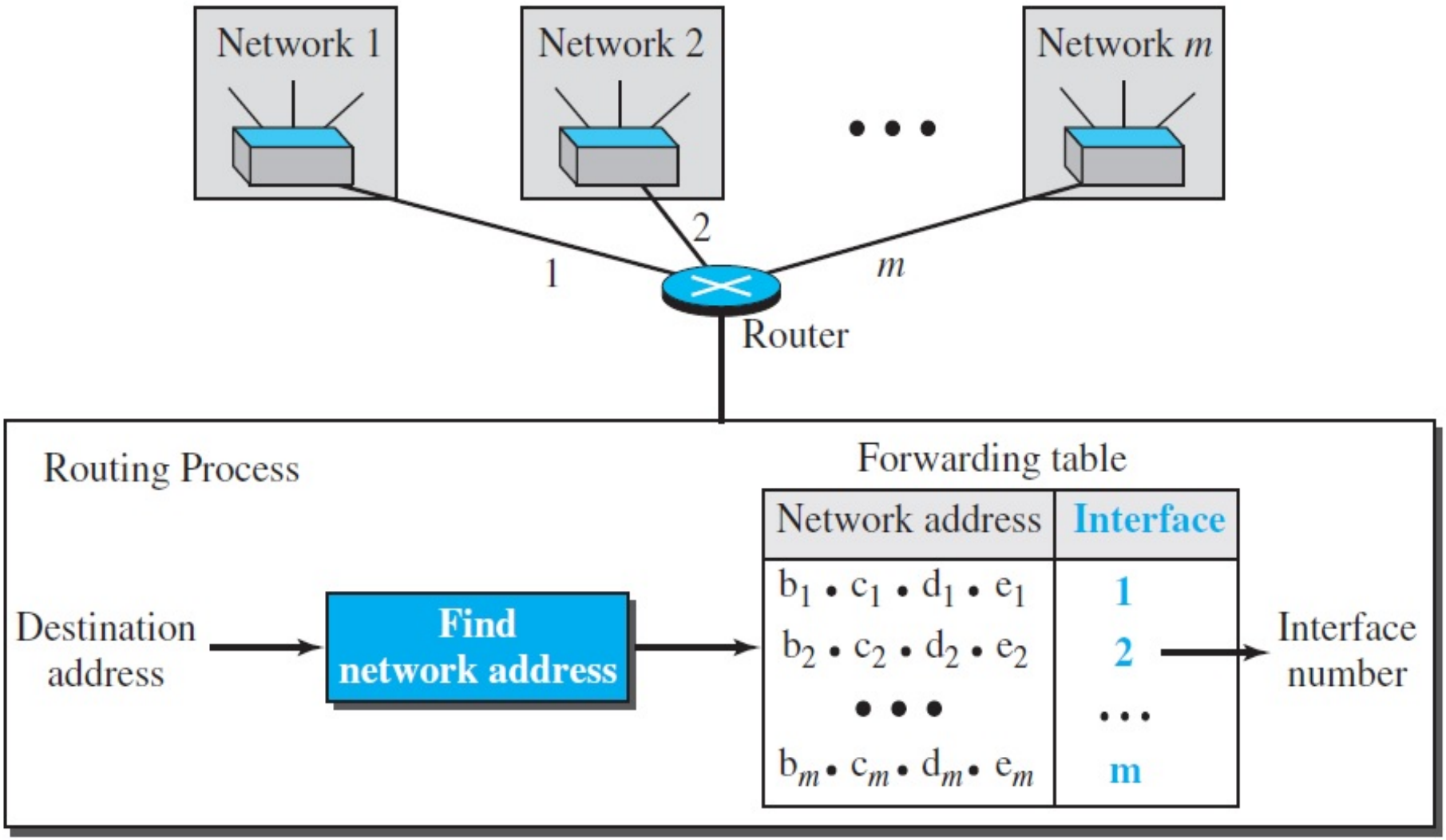
In classless addressing, an address cannot define the block the address belongs to. For example, the address 230.8.24.56 can belong to many blocks. Some of them are shown below with the value of the prefix associated with that block.

Prefix length:16	→	Block:	230.8.0.0	to	230.8.255.255
Prefix length:20	→	Block:	230.8.16.0	to	230.8.31.255
Prefix length:26	→	Block:	230.8.24.0	to	230.8.24.63
Prefix length:27	→	Block:	230.8.24.32	to	230.8.24.63
Prefix length:29	→	Block:	230.8.24.56	to	230.8.24.63
Prefix length:31	→	Block:	230.8.24.56	to	230.8.24.57

Classless Addressing : Network Address

- We can find all information about the block by any given address.
- The first address, the network address, is particularly important because it is used in routing a packet to its destination network.
- For the moment, let us assume that an internet is made of m networks and a router with m interfaces.
- Figure shows the idea. After the network address has been found, the router consults its forwarding table to find the corresponding interface from which the packet should be sent out. The network address is actually the identifier of the network; each network is identified by its network address.

Classless Addressing : Network Address



Classless Addressing : Block Allocation

- The ultimate responsibility of block allocation is given to a global authority called the Internet Corporation for Assigned Names and Numbers (ICANN). However, ICANN does not normally allocate addresses to individual Internet users. It assigns a large block of addresses to an ISP (or a larger organization).
- The number of requested addresses, N, needs to be a power of 2. The reason is that N = 2^{32-n} or n = 32 - log2N. If N is not a power of 2, we cannot have an integer value for n.
- The requested block needs to be allocated where there is an adequate number of contiguous addresses available in the address space. However, there is a restriction on choosing the first address in the block. The first address needs to be divisible by the number of addresses in the block. The reason is that the first address needs to be the prefix followed by (32 - n) number of 0s. The decimal value of the first address is then first address = (prefix in decimal) $\times$ 2^{32-n} = (prefix in decimal) $\times$ N.

Example : Block Allocation

- An ISP has requested a block of 1000 addresses.
- Since 1000 is not a power of 2, 1024 addresses are granted.
- The prefix length is calculated as n = 32 - $log_2 1024 = 22$.
- An available block, 18.14.12.0/22, is granted to the ISP.

Subnetting

- An organization (or an ISP) that is granted a range of addresses may divide the range into several subranges and assign each subrange to a subnetwork (or subnet).
- More levels of hierarchy can be created using subnetting.
- Note that nothing stops the organization from creating more levels. A subnetwork can be divided into several sub-subnetworks. A sub-subnetwork can be divided into several sub-sub-subnetworks, and so on.

Designing Subnets

- The subnetworks in a network should be carefully designed to enable the routing of packets.
- We assume the total number of addresses granted to the organization is N, the prefix length is n, the assigned number of addresses to each subnetwork is N_{sub}, and the prefix length for each subnetwork is n_{sub}.
- Then the following steps need to be carefully followed to guarantee the proper operation of the subnetworks.
- The number of addresses in each subnetwork should be a power of 2.
- The prefix length for each subnetwork should be found using the following formula:

$$n_{sub} = 32 - log_2 N_{sub} \tag{7}$$

- The starting address in each subnetwork should be divisible by the number of addresses in that subnetwork. This can be achieved if we first assign addresses to larger subnetworks.

Finding Information about Each Subnetwork

After designing the subnetworks, the information about each subnetwork, such as first and last address, can be found using the process we described to find the information about each network in the Internet.

Example of Subnetwork

An organization is granted a block of addresses with the beginning address 14.24.74.0/24. The organization needs to have 3 subblocks of addresses to use in its three subnets: one subblock of 10 addresses, one subblock of 60 addresses, and one subblock of 120 addresses. Design the subblocks.

Solution of Example

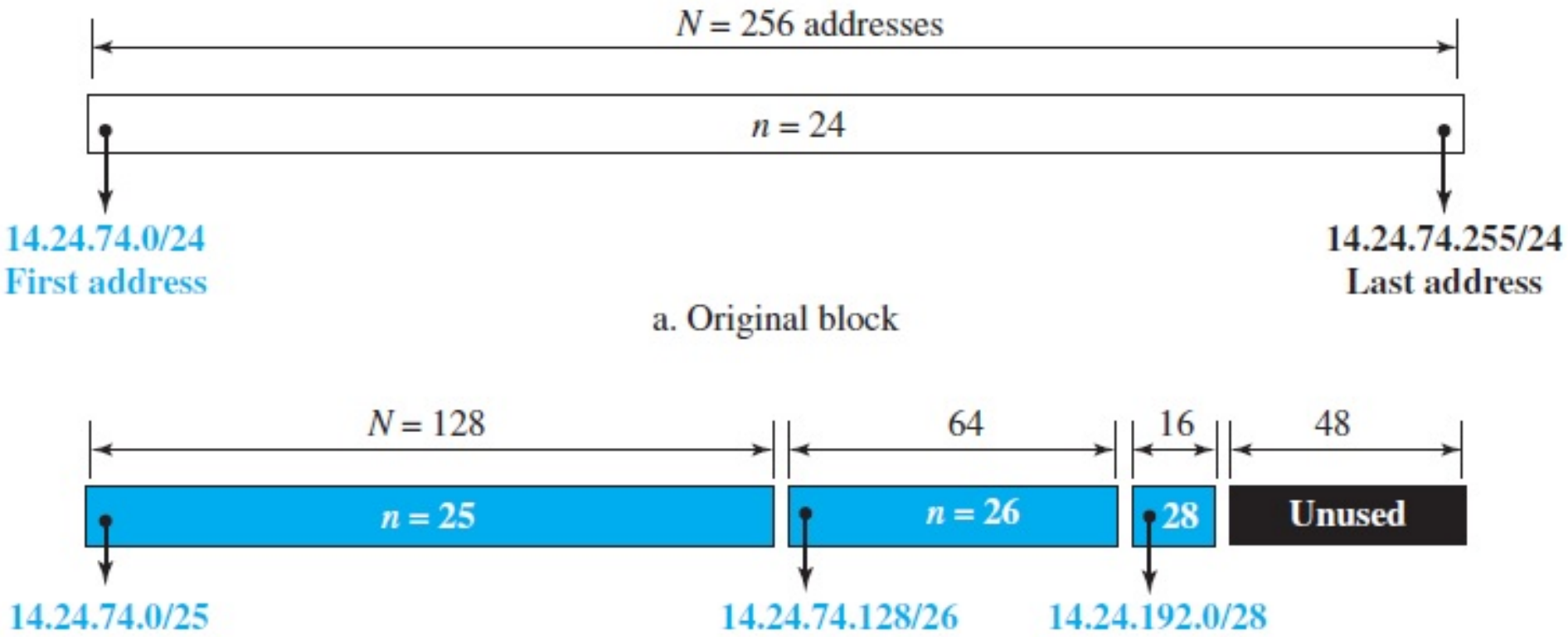
There are $2^{32-24} = 256$ addresses in this block. The first address is 14.24.74.0/24; the last address is 14.24.74.255/24. To satisfy the third requirement, we assign addresses to subblocks, starting with the largest and ending with the smallest one.

Solution of Example

- The number of addresses in the largest subblock, which requires 120 addresses, is not a power of 2. We allocate 128 addresses. The subnet mask for this subnet can be found as $n_1 = 32 - log_2 128 = 25$. The first address in this block is 14.24.74.0/25; the last address is 14.24.74.127/25.
- The number of addresses in the second largest subblock, which requires 60 addresses, is not a power of 2 either. We allocate 64 addresses. The subnet mask for this subnet can be found as $n_2 = 32 - log_2 64 = 26$. The first address in this block is 14.24.74.128/26; the last address is 14.24.74.191/26.
- The number of addresses in the smallest subblock, which requires 10 addresses, is not a power of 2 either. We allocate 16 addresses. The subnet mask for this subnet can be found as $n_3 = 32 - log_2 16 = 28$. The first address in this block is 14.24.74.192/28; the last address is 14.24.74.207/28.

Solution of Example

If we add all addresses in the previous subblocks, the result is 208 addresses, which means 48 addresses are left in reserve. The first address in this range is 14.24.74.208. The last address is 14.24.74.255. We don't know about the prefix length yet. Figure shows the configuration of blocks. We have shown the first address in each block.



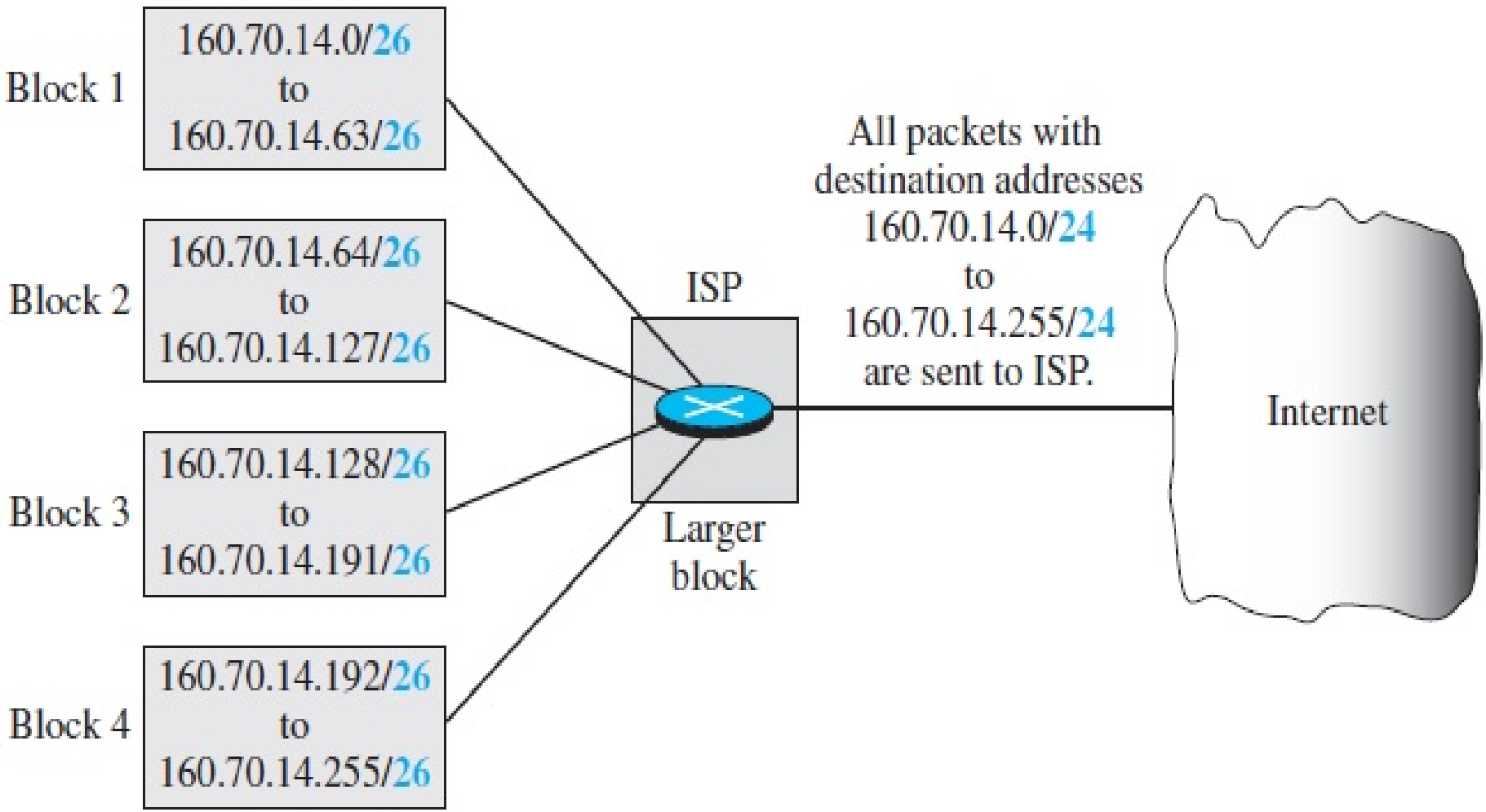
Address Aggregation

One of the advantages of the CIDR strategy is address aggregation (sometimes called address summarization or route summarization). When blocks of addresses are combined to create a larger block, routing can be done based on the prefix of the larger block. ICANN assigns a large block of addresses to an ISP. Each ISP in turn divides its assigned block into smaller subblocks and grants the subblocks to its customers.

Example

Figure shows how four small blocks of addresses are assigned to four organizations by an ISP. The ISP combines these four blocks into one single block and advertises the larger block to the rest of the world. Any packet destined for this larger block should be sent to this ISP. It is the responsibility of the ISP to forward the packet to the appropriate organization. This is similar to routing we can find in a postal network. All packages coming from outside a country are sent first to the capital and then distributed to the corresponding destination.

Address Aggregation



Special Addresses

We need to mention five IPV4 special addresses that are used for special purposes: this-host address, limited-broadcast address, loopback address, private addresses, and multicast addresses.

this-host Address

The only address in the block 0.0.0.0/32 is called the this-host address. It is used whenever a host needs to send an IP datagram but it does not know its own address to use as the source address.

Limited-broadcast Address

The only address in the block 255.255.255.255/32 is called the limited-broadcast address. It is used whenever a router or a host needs to send a datagram to all devices in a network. The routers in the network, however, block the packet having this address as the destination; the packet cannot travel outside the network.

Special Addresses

Loopback Address

The block 127.0.0.0/8 is called the loopback address. A packet with one of the addresses in this block as the destination address never leaves the host; it will remain in the host. Any address in the block is used to test a piece of software in the machine. For example, we can write a client and a server program in which one of the addresses in the block is used as the server address. We can test the programs using the same host to see if they work before running them on different computers.

Private Addresses

Four blocks are assigned as private addresses: 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16, and 169.254.0.0/16. These addresses are used in NAT (Network Address Translation).

Multicast Addresses

The block 224.0.0.0/4 is reserved for multicast addresses.

IPv6 ADDRESSING

- The main reason for migration from IPv4 to IPv6 is the small size of the address space in IPv4. The huge address space of IPv6 prevents address depletion in the future.
- An IPv6 address is 128 bits or 16 bytes (octets) long, four times the address length in IPv4.
- **IPv6 Representation:** A computer normally stores the address in binary, but it is clear that 128 bits cannot easily be handled by humans. Several notations have been proposed to represent IPv6 addresses when they are handled by humans. The following shows two of these notations: binary and colon hexadecimal.
- Binary (128 bits): 1111111011110110 . . . 1111111100000000
- Colon Hexadecimal- FEF6:BA98:7654:3210:ADEF:BBFF:2922:FF00
- Binary notation is used when the addresses are stored in a computer. The colon hexadecimal notation (or colon hex for short) divides the address into eight sections, each made of four hexadecimal digits separated by colons.

IPv6 : Abbreviation

- Although an IPv6 address, even in hexadecimal format, is very long, many of the digits are zeros. In this case, we can abbreviate the address. The leading zeros of a section can be omitted.
- Using this form of abbreviation, 0074 can be written as 74, 000F as F, and 0000 as 0. Note that 3210 cannot be abbreviated. Further abbreviation, often called zero compression, can be applied to colon hex notation if there are consecutive sections consisting of zeros only. We can remove all the zeros and replace them with a double semicolon.
- $FDEC : 0 : 0 : 0 : 0 : BBFF : 0 : FFFF -- > FDEC :: BBFF : 0 : FFFF$
- Note that this type of abbreviation is allowed only once per address. If there is more than one run of zero sections, only one of them can be compressed.

IPv6 : CIDR Notation

- IPv6 uses hierarchical addressing. For this reason, IPv6 allows slash or CIDR notation. For example, the following shows how we can define a prefix of 60 bits using CIDR. We will later show how an IPv6 address is divided into a prefix and a suffix.
- FDEC::BBFF:0:FFFF/60

Dynamic Host Configuration Protocol (DHCP)

- A large organization or an ISP can receive a block of addresses directly from ICANN and a small organization can receive a block of addresses from an ISP.
- After a block of addresses are assigned to an organization, the network administration can manually assign addresses to the individual hosts or routers.
- However, address assignment in an organization can be done automatically using the Dynamic Host Configuration Protocol (DHCP).
- DHCP is an application-layer program, using the client-server paradigm, that actually helps TCP/IP at the network layer.
- DHCP has found such widespread use in the Internet that it is often called a plug and- play protocol. In can be used in many situations. A network manager can configure DHCP to assign permanent IP addresses to the host and routers. DHCP can also be configured to provide temporary, on demand, IP addresses to hosts.

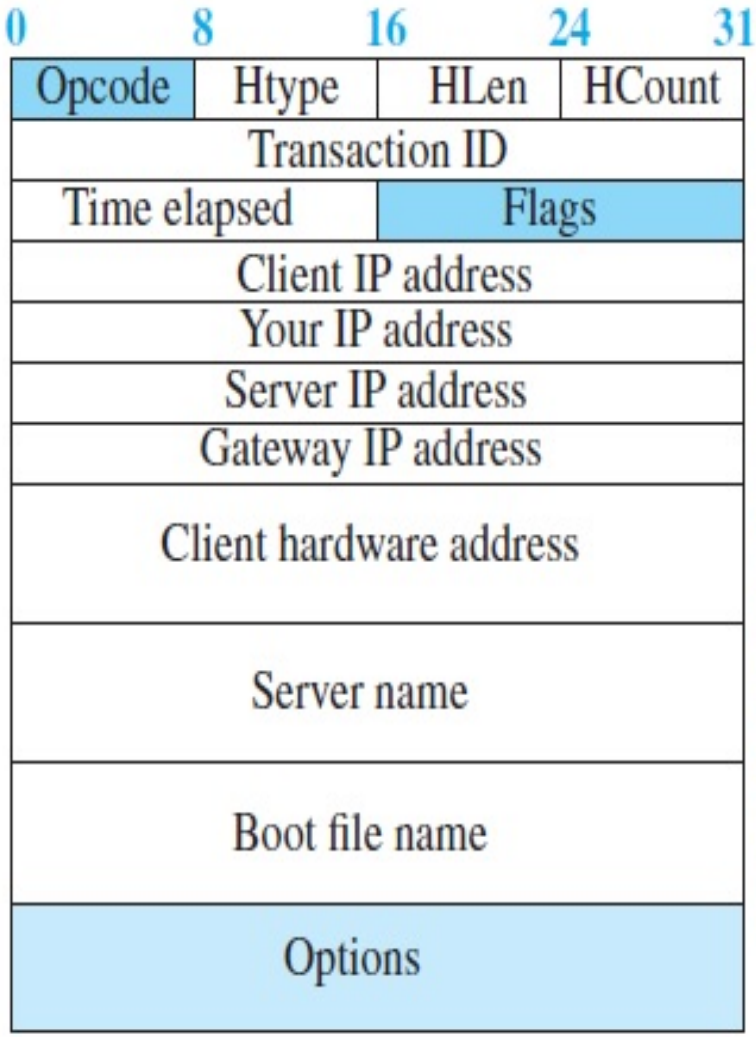
Dynamic Host Configuration Protocol (DHCP)

- The second capability can provide a temporary IP address to a traveller to connect her laptop to the Internet while she is staying in the hotel. It also allows an ISP with 1000 granted addresses to provide services to 4000 households, assuming not more than one-forth of customers use the Internet at the same time.
- In addition to its IP address, a computer also needs to know the network prefix (or address mask). Most computers also need two other pieces of information, such as the address of a default router to be able to communicate with other networks and the address of a name server to be able to use names instead of addresses.
- In other words, four pieces of information are normally needed: the computer address, the prefix, the address of a router, and the IP address of a name server. DHCP can be used to provide these pieces of information to the host.

DHCP Message Format

- DHCP is a client-server protocol in which the client sends a request message and the server returns a response message.
- Let us show the general format of the DHCP message in Figure. Most of the fields are explained in the figure, but we need to discuss the option field, which plays a very important role in DHCP.
- The 64-byte option field has a dual purpose. It can carry either additional information or some specific vendor information.
- The server uses a number, called a magic cookie, in the format of an IP address with the value of 99.130.83.99. When the client finishes reading the message, it looks for this magic cookie. If present, the next 60 bytes are options.
- An option is composed of three fields: a 1-byte tag field, a 1-byte length field, and a variable-length value field. There are several tag fields that are mostly used by vendors. If the tag field is 53, the value field defines one of the 8 message types shown in Figure. We show how these message types are used by DHCP.

DHCP Message Format

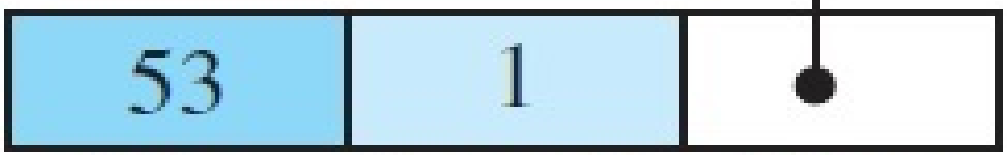


Fields:

Opcode: Operation code, request (1) or reply (2)
Htype: Hardware type (Ethernet, ...)
HLen: Length of hardware address
HCount: Maximum number of hops the packet can travel
Transaction ID: An integer set by the client and repeated by the server
Time elapsed: The number of seconds since the client started to boot
Flags: First bit defines unicast (0) or multicast (1); other 15 bits not used
Client IP address: Set to 0 if the client does not know it
Your IP address: The client IP address sent by the server
Server IP address: A broadcast IP address if client does not know it
Gateway IP address: The address of default router
Server name: A 64-byte domain name of the server
Boot file name: A 128-byte file name holding extra information
Options: A 64-byte field with dual purpose described in text

DHCP Option Format

1 DHCPDISCOVER
2 DHCPOFFER
3 DHCPREQUEST
4 DHCPDECLINE
5 DHCPACK
6 DHCPNACK
7 DHCPRELEASE
8 DHCPINFORM



Tag Length Value

DHCP Operation

- The joining host creates a DHCPDISCOVER message in which only the transaction- ID field is set to a random number. No other field can be set because the host has no knowledge with which to do so. This message is encapsulated in a UDP user datagram with the source port set to 68 and the destination port set to 67. The user datagram is encapsulated in an IP datagram with the source address set to 0.0.0.0 ("this host") and the destination address set to 255.255.255.255 (broadcast address). The reason is that the joining host knows neither its own address nor the server address.

DHCP Operation

- The DHCP server or servers (if more than one) responds with a DHCPOFFER message in which the your address field defines the offered IP address for the joining host and the server address field includes the IP address of the server. The message also includes the lease time for which the host can keep the IP address. This message is encapsulated in a user datagram with the same port numbers, but in the reverse order. The user datagram in turn is encapsulated in a datagram with the server address as the source IP address, but the destination address is a broadcast address, in which the server allows other DHCP servers to receive the offer and give a better offer if they can.

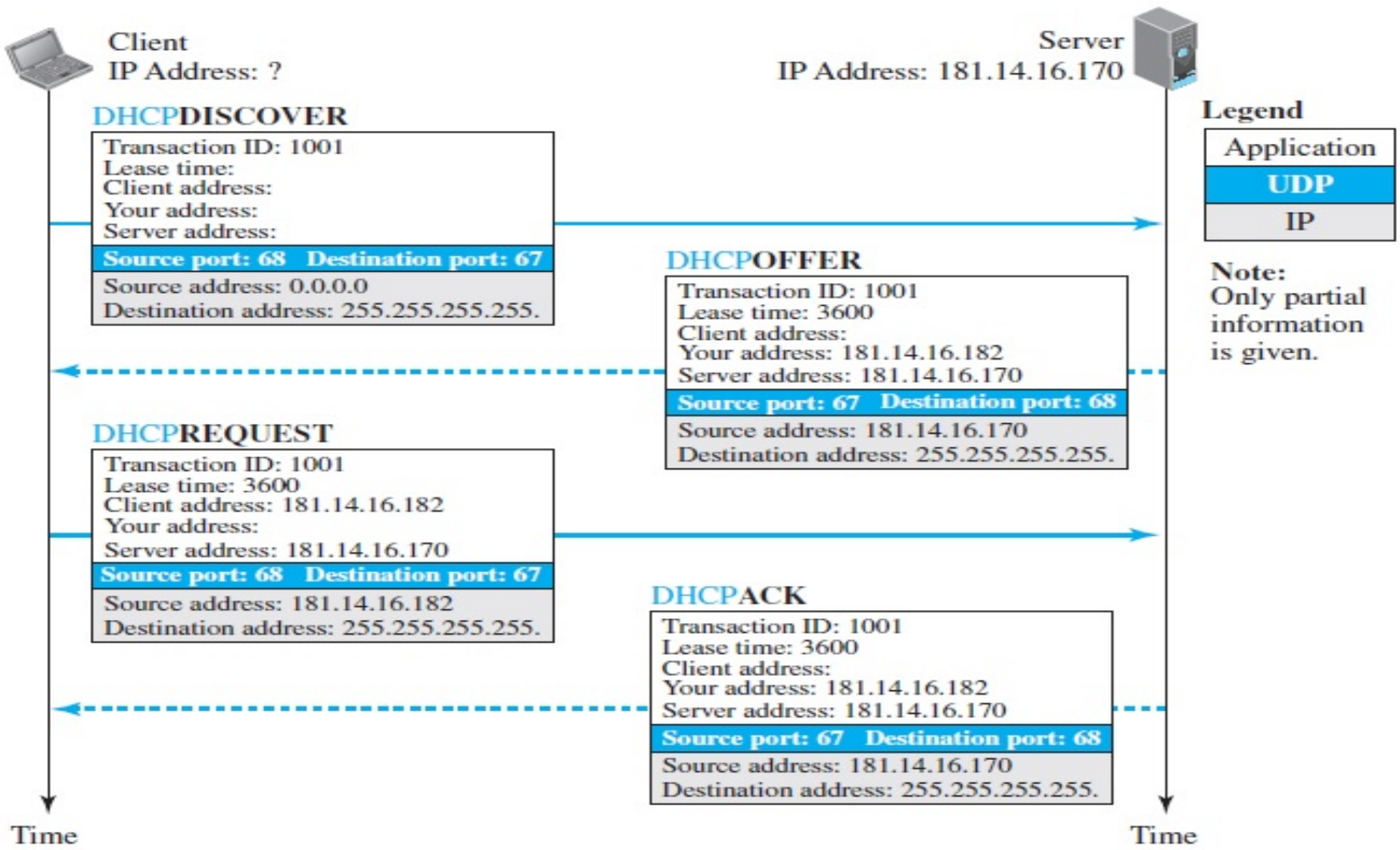
DHCP Operation

- The joining host receives one or more offers and selects the best of them. The joining host then sends a DHCPREQUEST message to the server that has given the best offer. The fields with known value are set. The message is encapsulated in a user datagram with port numbers as the first message. The user datagram is encapsulated in an IP datagram with the source address set to the new client address, but the destination address still is set to the broadcast address to let the other servers know that their offer was not accepted.

DHCP Operation

- Finally, the selected server responds with a DHCPACK message to the client if the offered IP address is valid. If the server cannot keep its offer (for example, if the address is offered to another host in between), the server sends a DHCPNACK message and the client needs to repeat the process. This message is also broadcast to let other servers know that the request is accepted or rejected.

DHCP Operation



DHCP : Two Well-Known Ports

- The DHCP uses two well-known ports (68 and 67) instead of one well-known and one ephemeral (temporary). The reason for choosing the well-known port 68 instead of an ephemeral port for the client is that the response from the server to the client is broadcast. Remember that an IP datagram with the limited broadcast message is delivered to every host on the network.
- Now assume that a DHCP client and a DAYTIME client, for example, are both waiting to receive a response from their corresponding server and both have accidentally used the same temporary port number (56017, for example). Both hosts receive the response message from the DHCP server and deliver the message to their clients. The DHCP client processes the message; the DAYTIME client is totally confused with a strange message received. Using a well-known port number prevents this problem from happening. The response message from the DHCP server is not delivered to the DAYTIME client, which is running on the port number 56017, not 68. The temporary port no. are selected from a different range than the well-known port no.

DHCP : Two Well-Known Ports

The curious reader may ask what happens if two DHCP clients are running at the same time. This can happen after a power failure and power restoration. In this case the messages can be distinguished by the value of the transaction ID, which separates each response from the other.

Using FTP

The server does not send all of the information that a client may need for joining the network. In the DHCPACK message, the server defines the pathname of a file in which the client can find complete information such as the address of the DNS server. The client can then use a file transfer protocol to obtain the rest of the needed information.

Error Control

DHCP uses the service of UDP, which is not reliable. To provide error control, DHCP uses two strategies. First, DHCP requires that UDP use the checksum. But, the use of the checksum in UDP is optional. Second, the DHCP client uses timers and a retransmission policy if it does not receive the DHCP reply to a request. However, to prevent a traffic jam when several hosts need to retransmit a request (for example, after a power failure), DHCP forces the client to use a random number to set its timers.

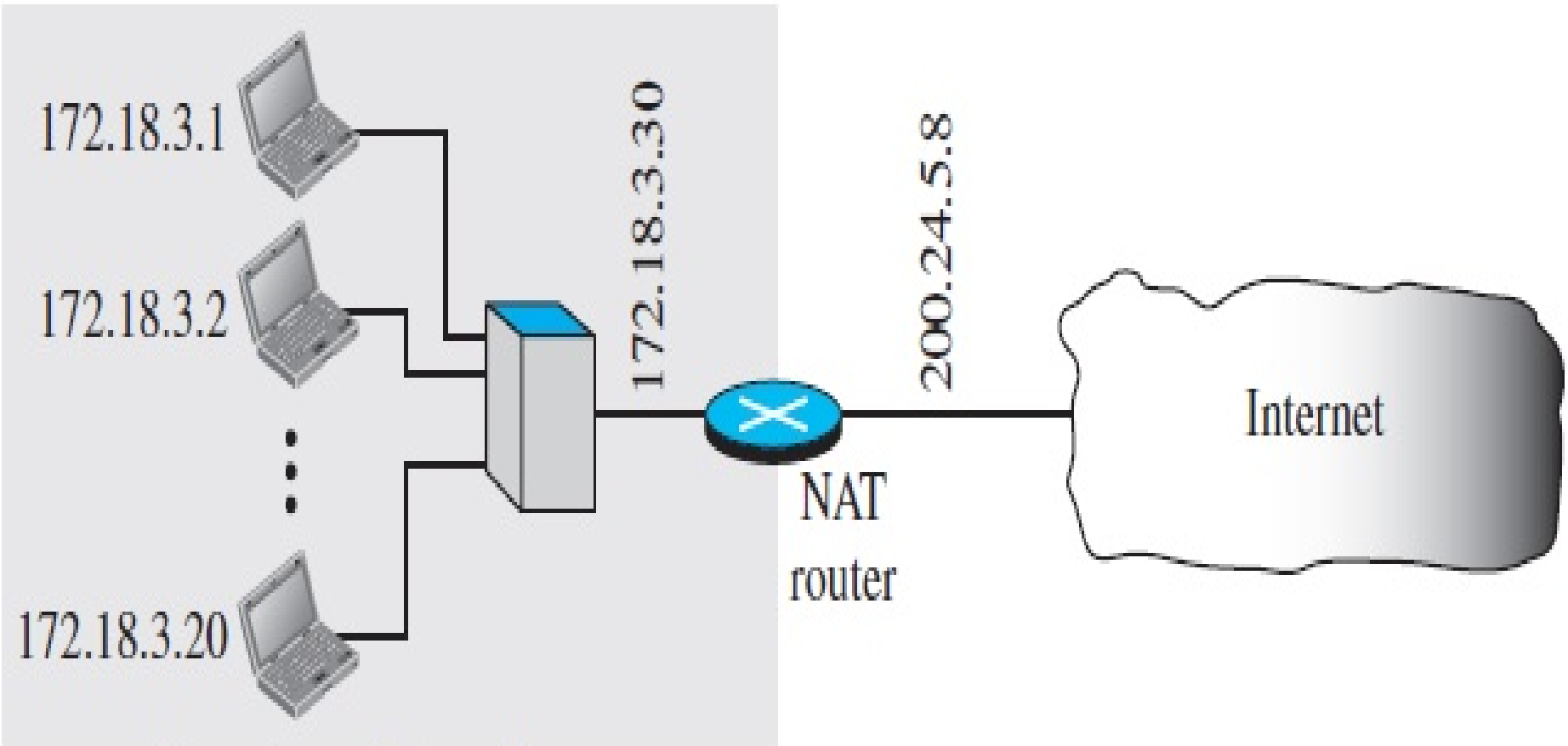
Network Address Resolution (NAT)

- Assume that an ISP has granted a small range of addresses to a small business or a household. If the business grows or the household needs a larger range, the ISP may not be able to grant the demand because the addresses before and after the range may have already been allocated to other networks. In most situations, however, only a portion of computers in a small network need access to the Internet simultaneously. This means that the number of allocated addresses does not have to match the number of computers in the network.
- For example, assume that in a small business with 20 computers the maximum number of computers that access the Internet simultaneously is only 4. Most of the computers are either doing some task that does not need Internet access or communicating with each other. This small business can use the TCP/IP protocol for both internal and universal communication. The business can use 20 (or 25) addresses from the private block addresses (discussed before) for internal communication; five addresses for universal communication can be assigned by the ISP.

Network Address Resolution (NAT)

- A technology that can provide the mapping between the private and universal addresses, and at the same time support virtual private networks, is Network Address Translation (NAT).
- The technology allows a site to use a set of private addresses for internal communication and a set of global Internet addresses (at least one) for communication with the rest of the world.
- The site must have only one connection to the global Internet through a NAT-capable router that runs NAT software.
- Figure shows a simple implementation of NAT.
- As the figure shows, the private network uses private addresses. The router that connects the network to the global address uses one private address and one global address. The private network is invisible to the rest of the Internet; the rest of the Internet sees only the NAT router with the address 200.24.5.8.

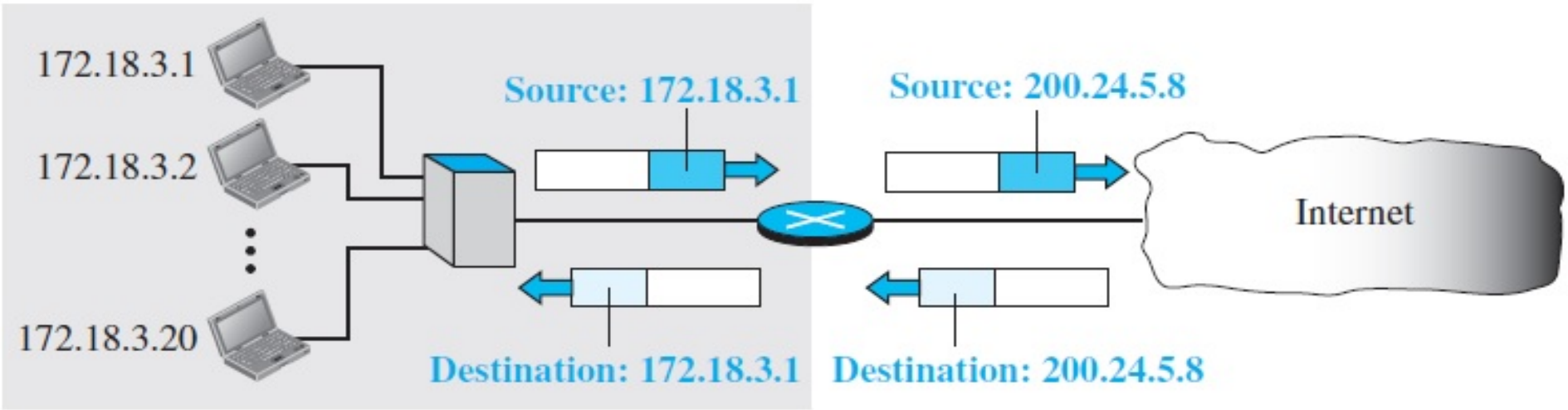
Network Address Resolution (NAT)



Network Address Resolution (NAT)

Address Translation

All of the outgoing packets go through the NAT router, which replaces the source address in the packet with the global NAT address. All incoming packets also pass through the NAT router, which replaces the destination address in the packet (the NAT router global address) with the appropriate private address. Figure shows an example of address translation.



Site using private addresses

Network Address Resolution (NAT)

Translation Table

It is noticed that translating the source addresses for an outgoing packet is straightforward. But Knowledge of destination address for a packet coming from the Internet to a NAT Router is difficult. There may be tens or hundreds of private IP addresses, each belonging to one specific host. The problem is solved if the NAT router has a translation table.

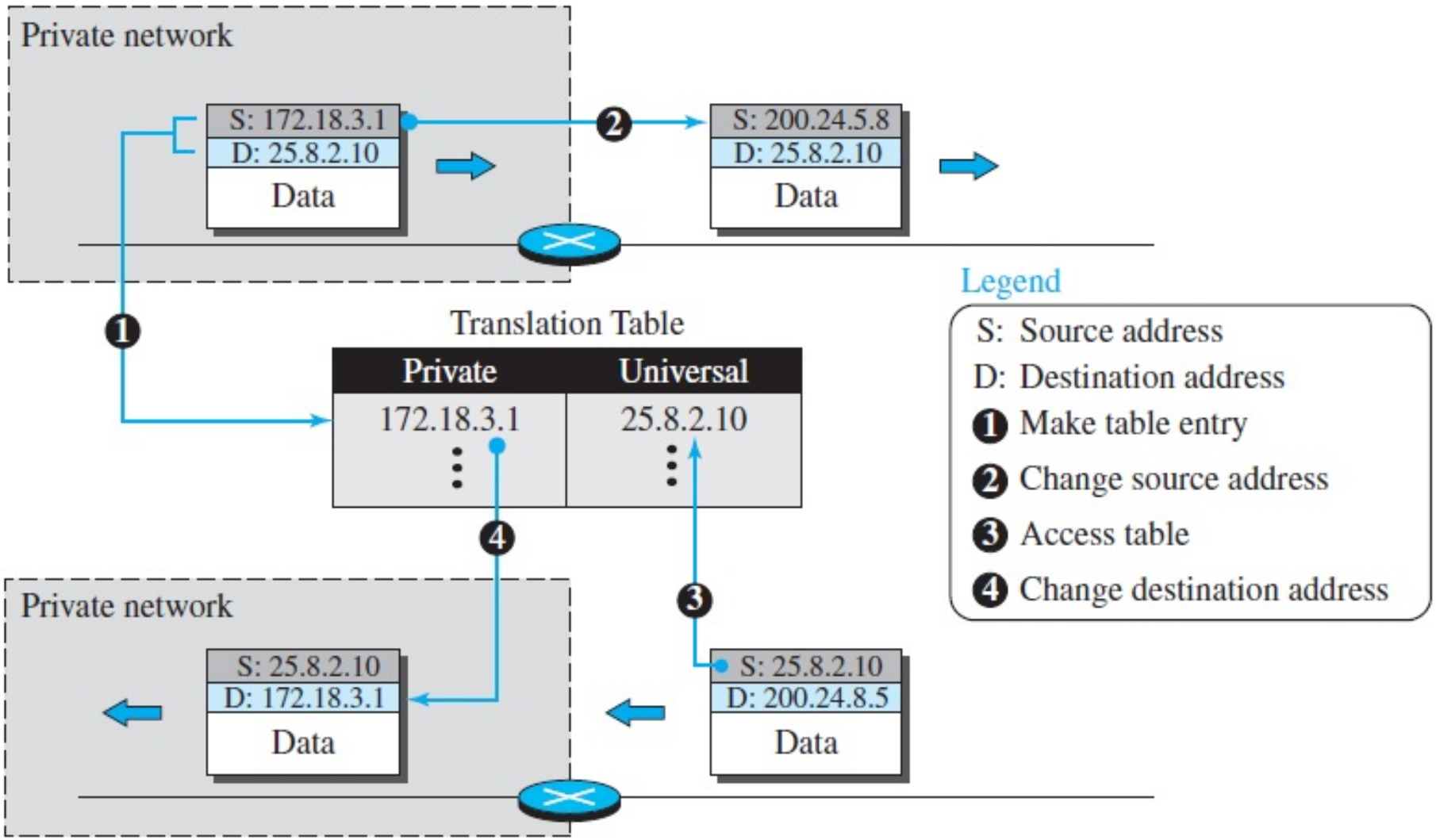
Translation Table : Using One IP Address

In its simplest form, a translation table has only two columns: the private address and the external address (destination address of the packet). When the router translates the source address of the outgoing packet, it also makes note of the destination address— where the packet is going. When the response comes back from the destination, the router uses the source address of the packet (as the external address) to find the private address of the packet. Figure shows the idea.

Translation Table : Using One IP Address

- In this strategy, communication must always be initiated by the private network. The NAT mechanism described requires that the private network start the communication.
- As we will see, NAT is used mostly by ISPs that assign a single address to a customer. The customer, however, may be a member of a private network that has many private addresses.
- In this case, communication with the Internet is always initiated from the customer site, using a client program such as HTTP, TELNET, or FTP to access the corresponding server program.
- For example, when e-mail that originates from outside the network site is received by the ISP e-mail server, it is stored in the mailbox of the customer until retrieved with a protocol such as POP.

Translation Table : Using One IP Address



Translation Table : Using a Pool of IP Addresses

- The use of only one global address by the NAT router allows only one private-network host to access a given external host.
- To remove this restriction, the NAT router can use a pool of global addresses.
- For example, instead of using only one global address (200.24.5.8), the NAT router can use four addresses (200.24.5.8, 200.24.5.9, 200.24.5.10, and 200.24.5.11).
- In this case, four private-network hosts can communicate with the same external host at the same time because each pair of addresses defines a separate connection.
- However, there are still some drawbacks. No more than four connections can be made to the same destination. No private-network host can access two external server programs (e.g., HTTP and TELNET) at the same time. And, likewise, two private-network hosts cannot access the same external server program (e.g., HTTP or TELNET) at the same time.

Translation Table : Using Both IP Addresses and Port Addresses

- To allow a many-to-many relationship between private-network hosts and external server programs, we need more information in the translation table.
- For example, suppose two hosts inside a private network with addresses 172.18.3.1 and 172.18.3.2 need to access the HTTP server on external host 25.8.3.2.
- If the translation table has five columns, instead of two, that include the source and destination port addresses and the transport-layer protocol, the ambiguity is eliminated.
- Note that when the response from HTTP comes back, the combination of source address (25.8.3.2) and destination port address (1401) defines the private network host to which the response should be directed. Note also that for this translation to work, the ephemeral port addresses (1400 and 1401) must be unique.
- Figure shows an example of such a table.

Five-column translation table

Private address	*Private port*	*External address*	*External port*	*Transport protocol*
172.18.3.1	1400	25.8.3.2	80	TCP
172.18.3.2	1401	25.8.3.2	80	TCP
⋮	⋮	⋮	⋮	⋮

Part-3

Routing

- Routing is the process of selecting a path for traffic in a network or between or across multiple networks. Broadly, routing is performed in many types of networks, including circuit-switched networks, such as the public switched telephone network (PSTN), and computer networks, such as the Internet.
- In an internet, the goal of the network layer is to deliver a datagram from its source to its destination or destinations.
- If a datagram is destined for only one destination (one-to-one delivery), we have unicast routing.
- If the datagram is destined for several destinations (one-to-many delivery), we have multicast routing.
- The routing can be possible if a router has a forwarding table to forward a packet to the appropriate next node on its way to the final destination or destinations.
- To make 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

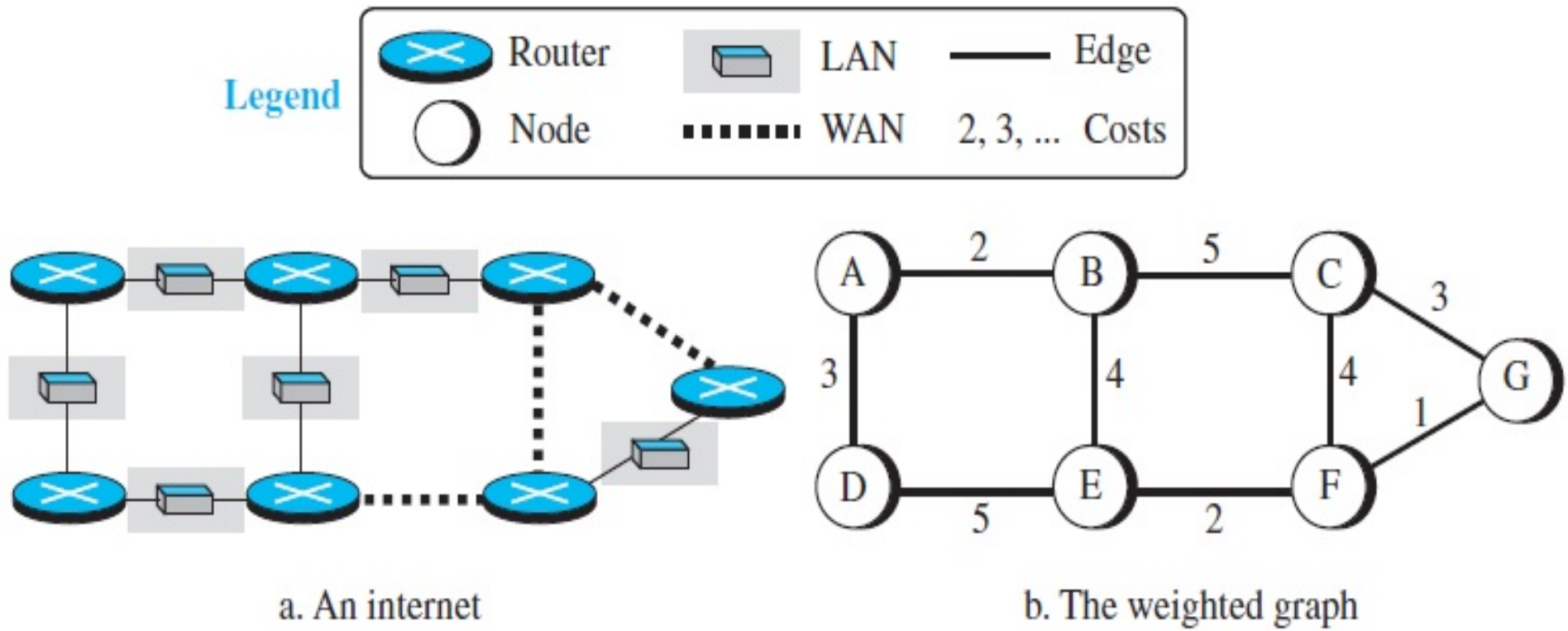
- Unicast routing in the Internet, with a large number of routers and a huge number of hosts, can be done only by using hierarchical routing: routing in several steps using different routing algorithms.
- In unicast routing, a packet is routed, hop by hop, from its source to its destination by the help 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.
- With the above explanation, 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).

Unicast Routing : An Internet as a Graph

- Although a packet needs to visit the source and the destination routers, the question is what other routers the packet should visit. In other words, there are several routes that a packet can travel from the source to the destination; what must be determined is which route the packet should take.
- 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. If a weighted graph is used to represent a geographical area, the nodes can be cities and the edges can be roads connecting the cities; the weights, in this case, are distances between cities.

Unicast Routing : An Internet as a Graph

- In routing, however, the cost of an edge has a different interpretation in different routing protocols
- For the moment, we assume that there is a cost associated with each edge. If there is no edge between the nodes, the cost is infinity. Figure shows how an internet can be modeled as a graph.



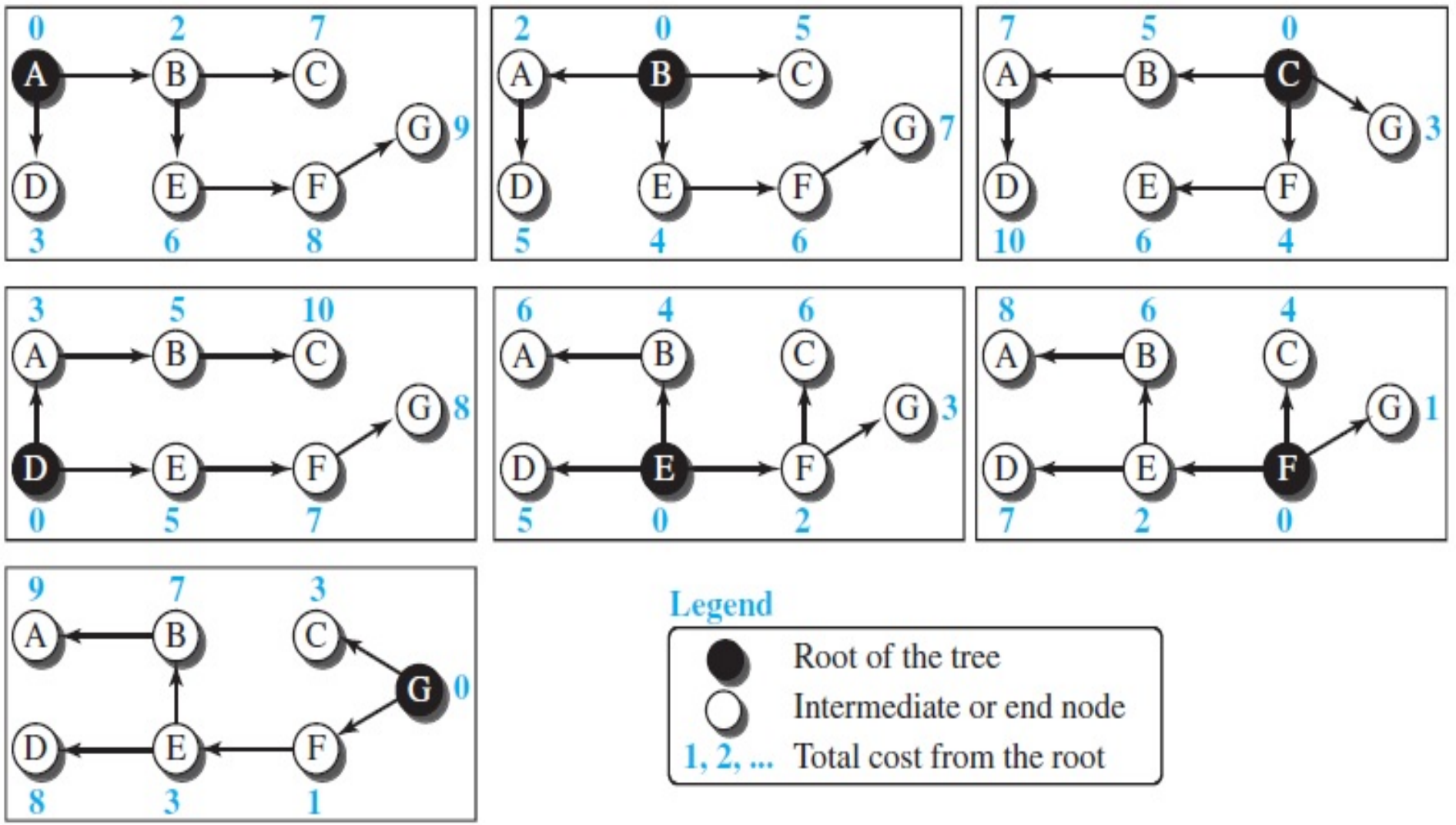
a. An internet

b. The weighted graph

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 $\times$ (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.
- In this way, we can have only one shortest-path tree for each node; we have N least-cost trees for the whole internet.
- Figure shows the seven least-cost trees for the internet.

Least-Cost Trees for nodes in the Internet



Least Cost Trees

The least-cost trees for a weighted graph can have several properties if they are created using consistent criteria.

- The least-cost route from X to Y in X's tree is the inverse of the least-cost route from Y to X in Y's tree; the cost in both directions is the same. For example, in Figure, the route from A to F in A's tree is (A $\rightarrow$ B $\rightarrow$ E $\rightarrow$ F), but the route from F to A in F's tree is (F $\rightarrow$ E $\rightarrow$ B $\rightarrow$ A), which is the inverse of the first route. The cost is 8 in each case.
- Instead of travelling from X to Z using X's tree, we can travel from X to Y using X's tree and continue from Y to Z using Y's tree. For example, in Figure, we can go from A to G in A's tree using the route (A $\rightarrow$ B $\rightarrow$ E $\rightarrow$ F $\rightarrow$ G). We can also go from A to E in A's tree (A $\rightarrow$ B $\rightarrow$ E) and then continue in E's tree using the route (E $\rightarrow$ F $\rightarrow$ G). The combination of the two routes in the second case is the same route as in the first case. The cost in the first case is 9; the cost in the second case is also 9 (6 + 3).

Routing Algorithms

Distance-Vector Routing

- The goal of Distance Vector routing algorithms is 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).
- There are two topics are covered now: the Bellman-Ford equation and the concept of distance vectors.

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.
- 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.

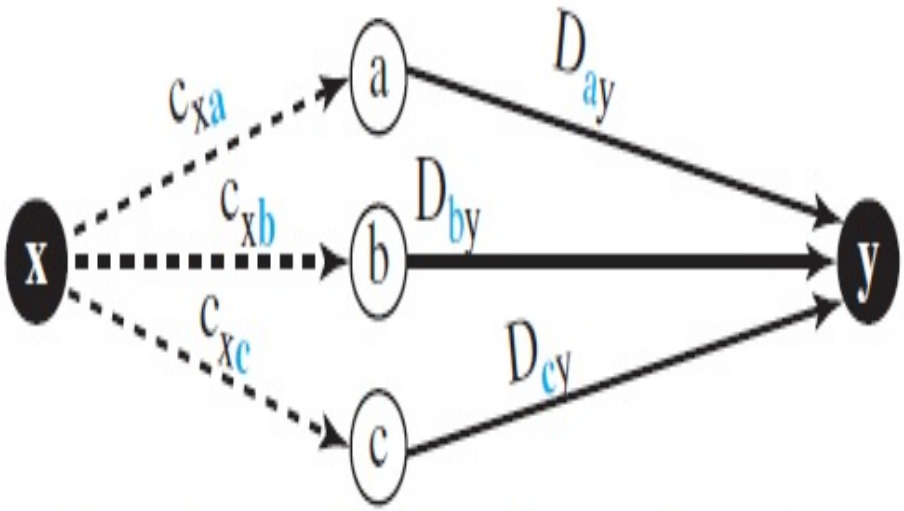
$$D_{xy} = min\{(c_{xa} + D_{ay}), (c_{xb} + D_{by}), (c_{xc} + D_{cy}), \dots\} \quad (8)$$

- 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:

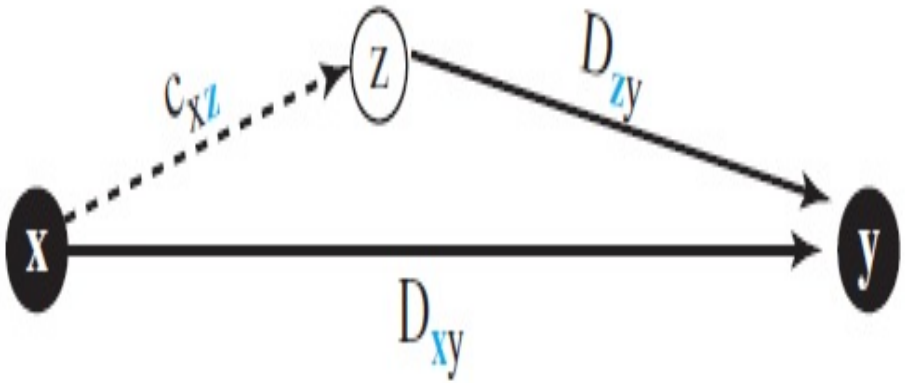
$$D_{xy} = min\{(D_{xy}, (c_{xz} + D_{zy})\} \quad (9)$$

Bellman-Ford Equation

Figure shows the idea graphically for both cases.



a. General case with three intermediate nodes



b. Updating a path with a new route

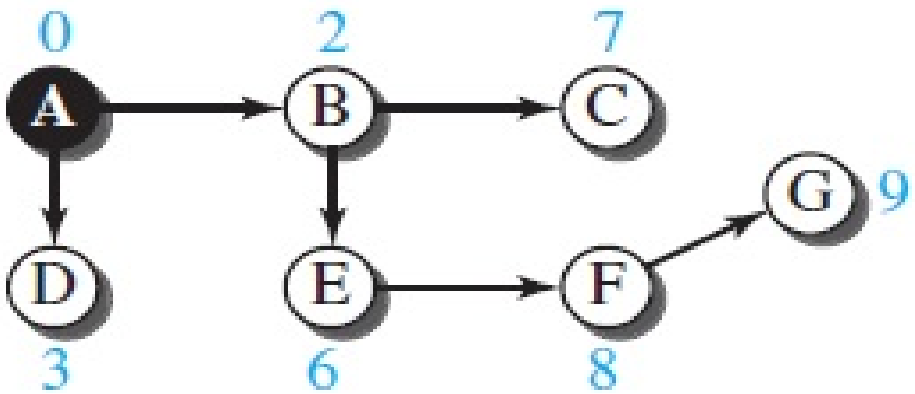
Bellman-Ford Equation

- We can say that the Bellman-Ford equation enables us to build a new least-cost path from previously established least-cost paths.
- In Figure, we can think of (a→y), (b→y), and (c→y) as previously established least-cost paths and (x→y) as the new least-cost path.
- We can even think of this equation as the builder of a new least-cost tree from previously established least-cost trees if we use the equation repeatedly. In other words, the use of this equation in distance-vector routing is a witness that this method also uses least-cost trees.

Distance Vectors

- 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(next) shows the tree for node A in the internet in Figure (Previous) and the corresponding distance vector.
- The name of the distance vector defines the root, the indexes define the destinations, and the value of each cell defines the least cost from the root to the destination.
- A distance vector does not give the path to the destinations as the least-cost tree does; it gives only the least costs to the destinations.

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

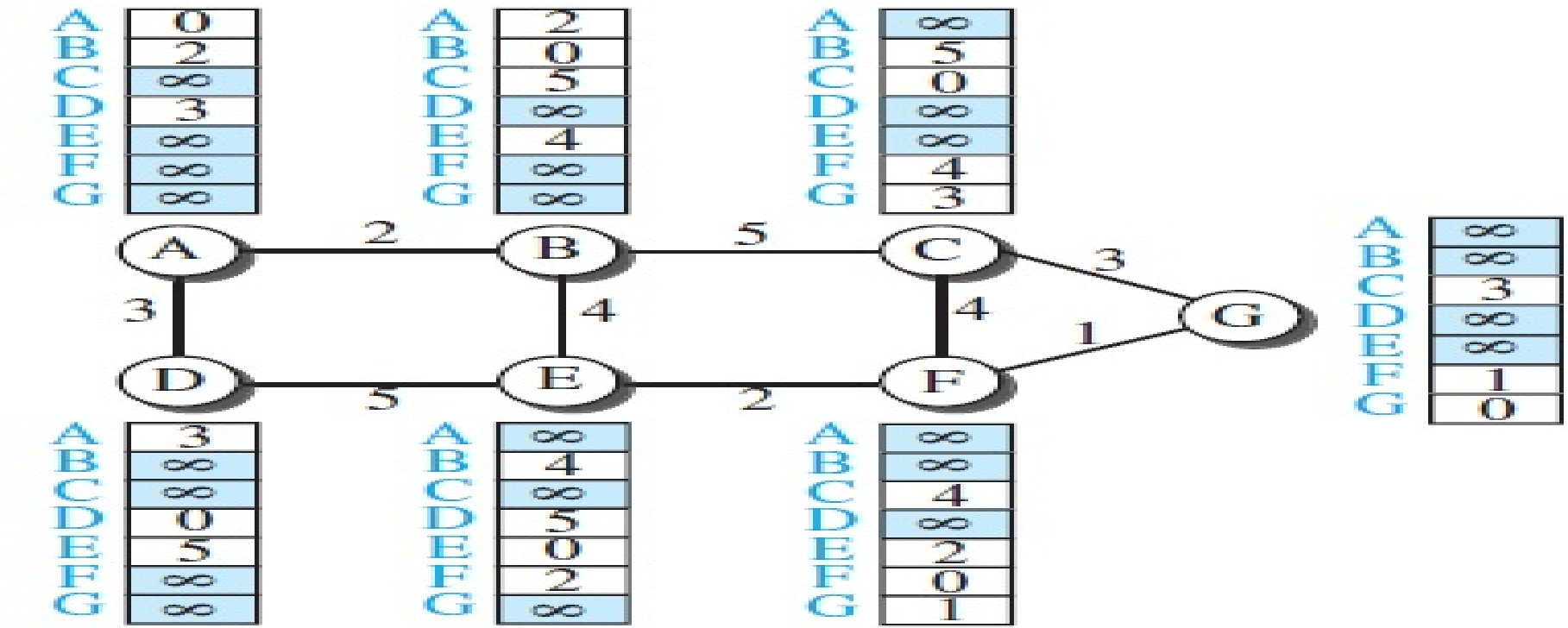
b. Distance vector for node A

Distance Vectors

- A distance vector can represent least-cost paths in a least-cost tree.
- 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.
- The node sends some greeting messages out of its interfaces and discovers the identity of the immediate neighbors and the distance between itself and each neighbor.
- It then makes a simple distance vector by inserting the discovered distances in the corresponding cells and leaves the value of other cells as infinity.
- Do these distance vectors represent least-cost paths? They do, considering the limited information a node has. When we know only one distance between two nodes, it is the least cost.

Distance Vectors

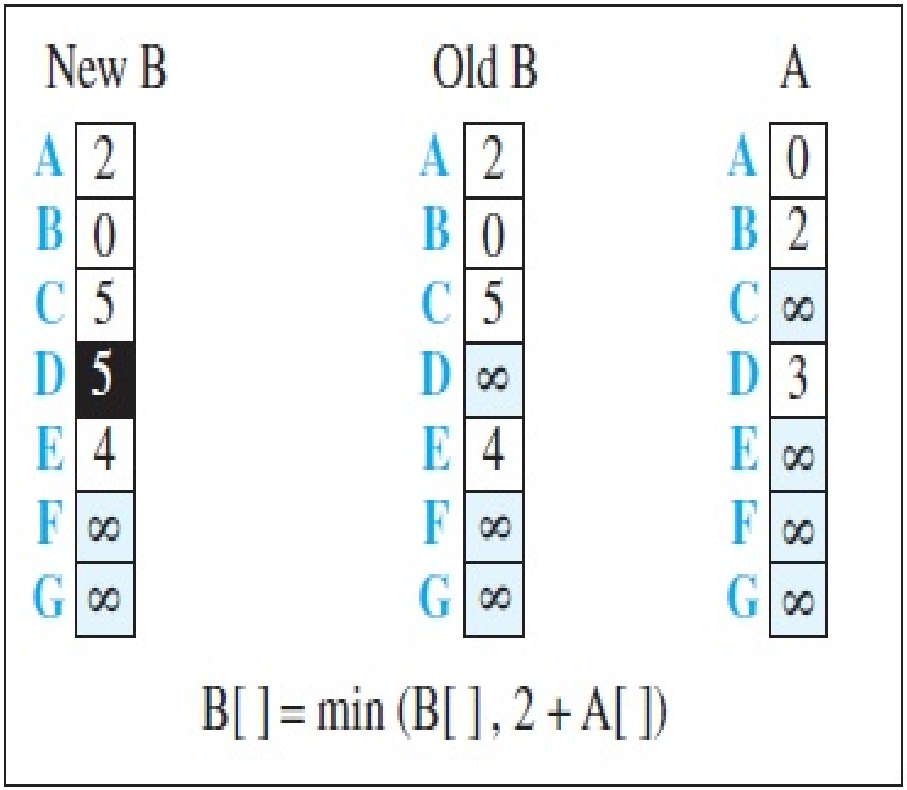
Figure shows all distance vectors for our internet. However, we need to mention that these vectors are made asynchronously, when the corresponding node has been booted; the existence of all of them in a figure does not mean synchronous creation of them.



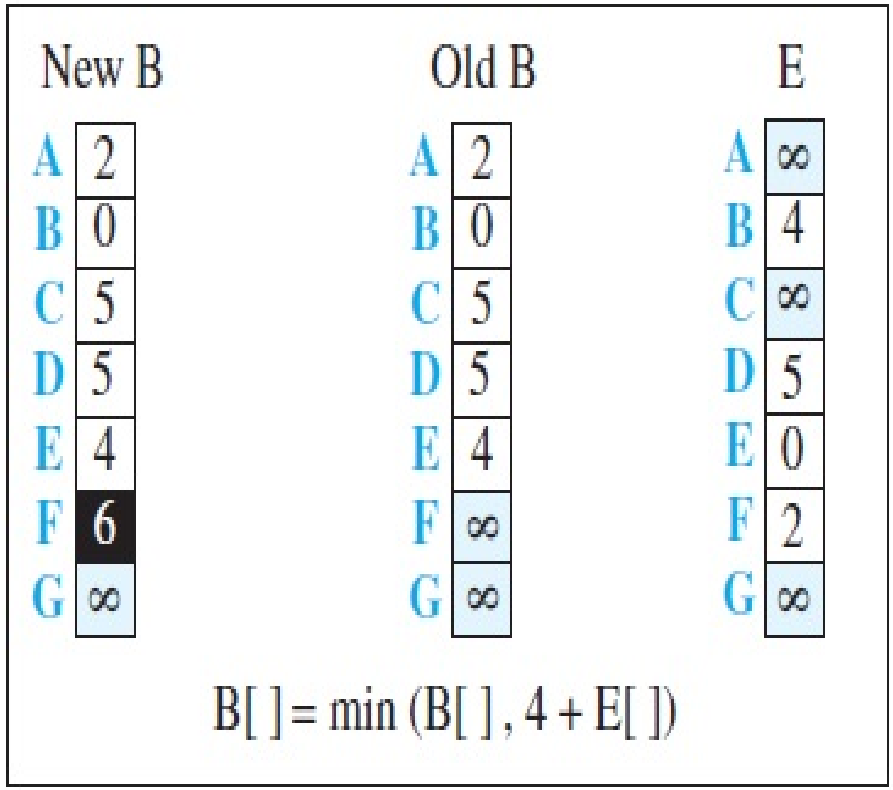
Distance Vectors

- These rudimentary vectors cannot help the internet to effectively forward a packet. For example, node A thinks that it is not connected to node G because the corresponding cell shows the least cost of infinity.
- To improve these 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 (second case). However, we need to understand that we need to update, not only one least cost, but N of them in which N is the number of the nodes in the internet.
- If we are using a program, we can do this using a loop; if we are showing the concept on paper, we can show the whole vector instead of the N separate equations.
-

Updating Distance Vector



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

Updating Distance Vector

- We show the whole vector instead of seven equations for each update in Figure. The figure shows two asynchronous events, happening one after another with some time in between.
- In the first event, node A has sent its vector to node B. Node B updates its vector using the cost $c_{BA} = 2$. In the second event, node E has sent its vector to node B. Node B updates its vector using the cost $c_{EA} = 4$.
- After the first event, node B has one improvement in its vector: its least cost to node D has changed from infinity to 5 (via node A).
- After the second event, node B has one more improvement in its vector; its least cost to node F has changed from infinity to 6 (via node E).

Distance Vector Routing Algorithm

- We hope that we have convinced the reader that exchanging vectors eventually stabilizes the system and allows all nodes to find the ultimate least cost between themselves and any other node.
- We need to remember that after updating a node, it immediately sends its updated vector to all neighbors. Even if its neighbors have received the previous vector, the updated one may help more.
- we can give a simplified pseudocode for the distance-vector routing algorithm, as shown in Table. The algorithm is run by its node independently and asynchronously.
- Lines 4 to 11 initialize the vector for the node. Lines 14 to 23 show how the vector can be updated after receiving a vector from the immediate neighbor. The for loop in lines 17 to 20 allows all entries (cells) in the vector to be updated after receiving a new vector. Note that the node sends its vector in line 12, after being initialized, and in line 22, after it is updated.

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 D_w 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] + D_w[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  } // End of Distance Vector
```

Distance-Vector Routing Algorithm

Count to Infinity

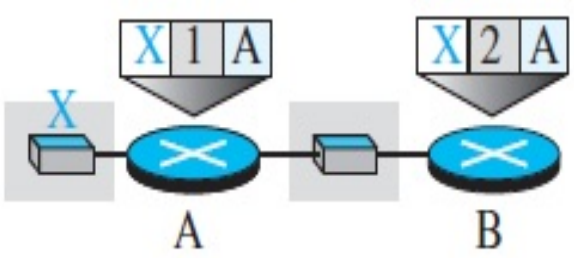
- A problem with distance-vector routing is that any decrease in cost (good news) propagates quickly, but any increase in cost (bad news) will propagate slowly.
- For a routing protocol to work properly, if a link is broken (cost becomes infinity), every other router should be aware of it immediately, but in distance-vector routing, this takes some time.
- The problem is referred to as count to infinity. It sometimes takes several updates before the cost for a broken link is recorded as infinity by all routers.

Example of Count to Infinity : Two Node Loop

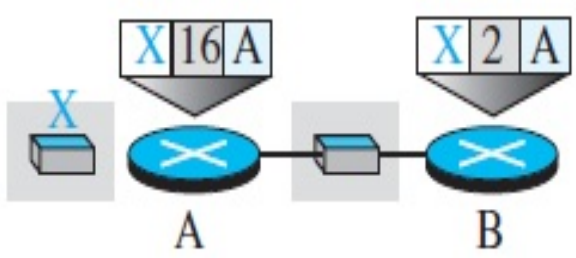
- One example of count to infinity is the two-node loop problem. To understand the problem, let us look at the scenario depicted in Figure.
- The figure shows a system with three nodes. We have shown only the portions of the forwarding table needed for our discussion.
- At the beginning, both nodes A and B know how to reach node X. But suddenly, the link between A and X fails. Node A changes its table. If A can send its table to B immediately, everything is fine.
- However, the system becomes unstable if B sends its forwarding table to A before receiving A's forwarding table. Node A receives the update and, assuming that B has found a way to reach X, immediately updates its forwarding table. Now A sends its new update to B. Now B thinks that something has been changed around A and updates its forwarding table.
- The cost of reaching X increases gradually until it reaches infinity. At this moment, both A and B know that X cannot be reached. However, during this time the system is not stable. Node A thinks that the route to X is via B; node B thinks that the route to X is via A.

Example of Count to Infinity : Two Node Loop

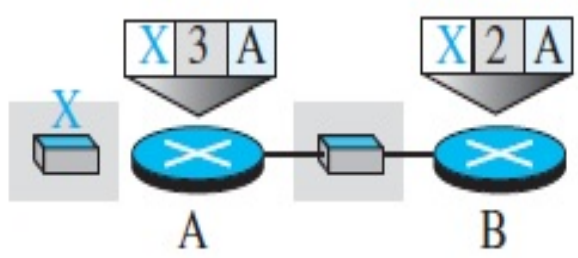
- If A receives a packet destined for X, the packet goes to B and then comes back to A. Similarly, if B receives a packet destined for X, it goes to A and comes back to B. Packets bounce between A and B, creating a two-node loop problem.
- A few solutions have been proposed for instability of this kind.



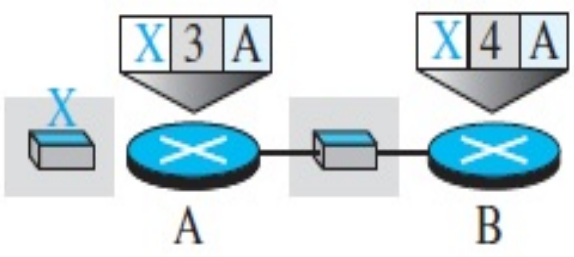
a. Before failure



b. After link failure

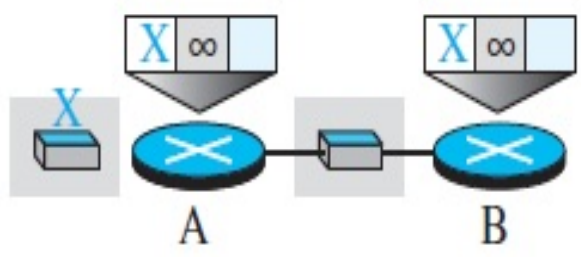


c. After A is updated by B



d. After B is updated by A

• • •



e. Finally

A solution of Two Loop Problem: Split Horizon

- One solution to instability is called split horizon.
- In this strategy, each node sends only part of its table through each interface, instead of flooding the table through each interface.
- If, according to its table, node B thinks that the optimum route to reach X is via A, it does not need to advertise this piece of information to A; the information has come from A (A already knows). Taking information from node A, modifying it, and sending it back to node A is what creates the confusion.
- In our scenario, node B eliminates the last line of its forwarding table before it sends it to A. In this case, node A keeps the value of infinity as the distance to X. Later, when node A sends its forwarding table to B, node B also corrects its forwarding table. The system becomes stable after the first update: both node A and node B know that X is not reachable.

A solution of Two Loop Problem: Split Horizon with Poison Reverse

- Using the split-horizon strategy has one drawback. Normally, the corresponding protocol uses a timer, and if there is no news about a route, the node deletes the route from its table. When node B in the previous scenario eliminates the route to X from its advertisement to A, node A cannot guess whether this is due to the split-horizon strategy (the source of information was A) or because B has not received any news about X recently.
- In the poison reverse strategy B can still advertise the value for X, but if the source of information is A, it can replace the distance with infinity as a warning: "Do not use this value; what I know about this route comes from you."
- **Three-Node Instability:** The two-node instability can be avoided using split horizon combined with poison reverse. However, if the instability is between three nodes, stability cannot be guaranteed.

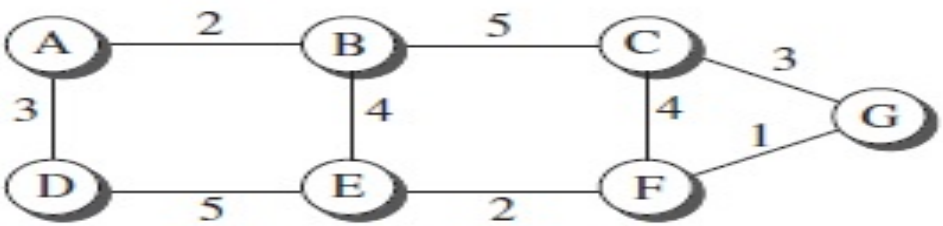
Link-State Routing

- A routing algorithm that directly creates 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 Routing

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 shows an example of an LSDB for the graph. 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.



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

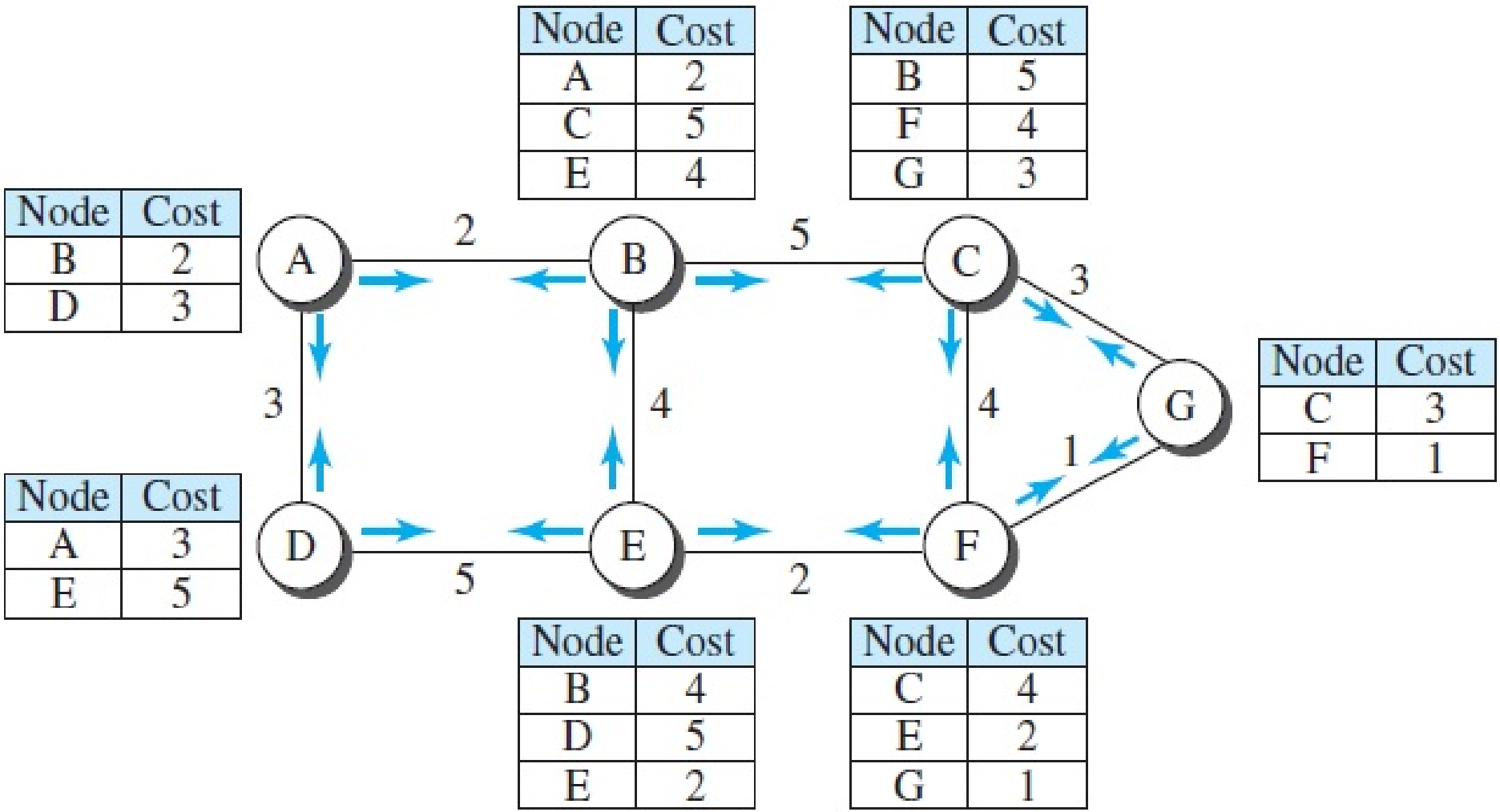
Link State Routing

- With the help of flooding process each node can create this LSDB that contains information about the whole internet.
- Each node can send some greeting messages to all its immediate neighbors (those nodes to which it is connected directly) to collect two pieces of information for each neighboring node: the identity of the node and the cost of the link. The combination of these two pieces of information is called the LS packet (LSP); the LSP is sent out of each interface, as shown in Figure.
- When a node receives an LSP from one of its interfaces, it compares the LSP with the copy it may already have. If the newly arrived LSP is older than the one it has (found by checking the sequence number), it discards the LSP. If it is newer or the first one received, the node discards the old LSP (if there is one) and keeps the received one.

Link State Routing

- It then sends a copy of it out of each interface except the one from which the packet arrived. This guarantees that flooding stops somewhere in the network (where a node has only one interface). We need to convince ourselves that, after receiving all new LSPs, each node creates the comprehensive LSDB as shown in Figure. This LSDB is the same for each node and shows the whole map of the internet. In other words, a node can make the whole map if it needs to, using this LSDB.
- We can compare the link-state routing algorithm with the distance-vector routing algorithm. In the distance-vector routing algorithm, each router tells its neighbors what it knows about the whole internet; in the link-state routing algorithm, each router tells the whole internet what it knows about its neighbors.

Link State Routing: LSPs created and sent out by each node to build LSDB



Link State Routing : Formation of Least-Cost Trees : Dijkstra Algorithm

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. After this node is added to the tree, the cost of all other nodes not in the tree needs to be updated because the paths may have been changed.
3. The node repeats step 2 until all nodes are added to the tree.

We need to convince ourselves that the above three steps finally create the least-cost tree.

Link State Routing : Dijkstra Algorithm

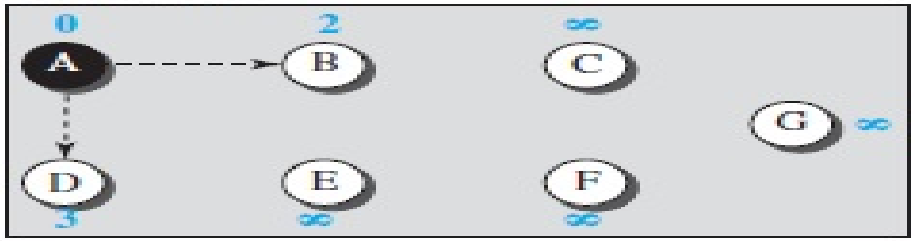
```
1   Dijkstra's Algorithm ( )
2   {
3       // Initialization
4       Tree = {root}                       // Tree is made only of the root
5       for (y = 1 to N)                    // N is the number of nodes
6       {
7           if (y is the root)
8               D[y] = 0                    // 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 neighbors of w
20          for (every node x, which is a 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  } // End of Dijkstra
```

Link State Routing : Least Cost Tree : Dijkstra Algorithm

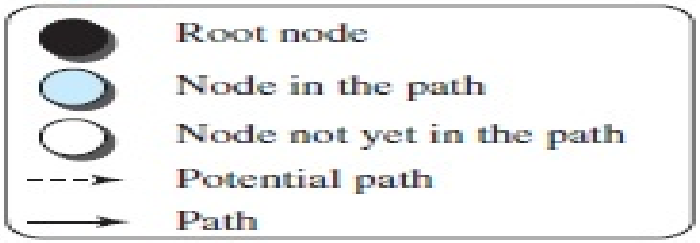
- Lines 4 to 13 implement step 1 in the algorithm. Lines 16 to 23 implement step 2 in the algorithm. Step 2 is repeated until all nodes are added to the tree.
- Next Figure shows the formation of the least-cost tree for the graph in Previous Figure using Dijkstra's algorithm. We need to go through an initialization step and six iterations to find the least-cost tree.

Link State Routing : Dijkstra Algorithm

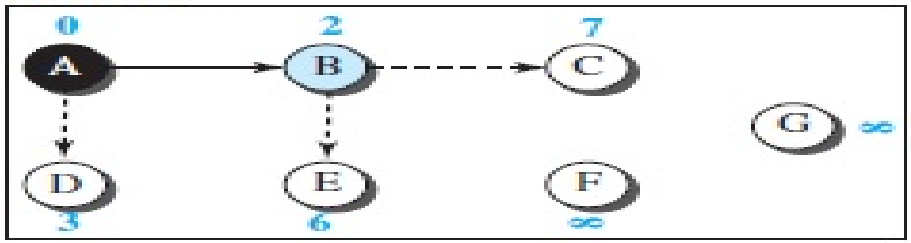
Initialization



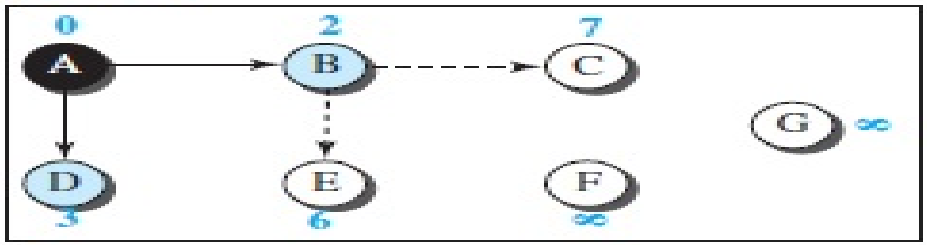
Legend



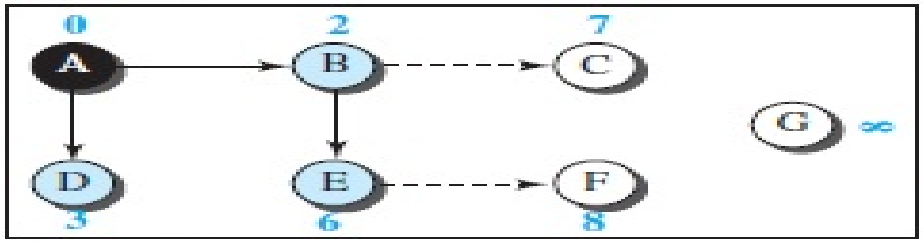
Iteration 1



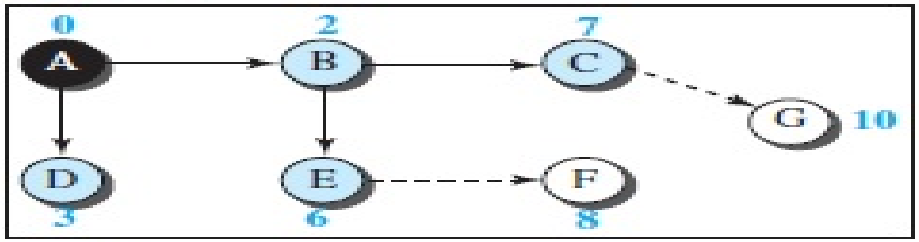
Iteration 2



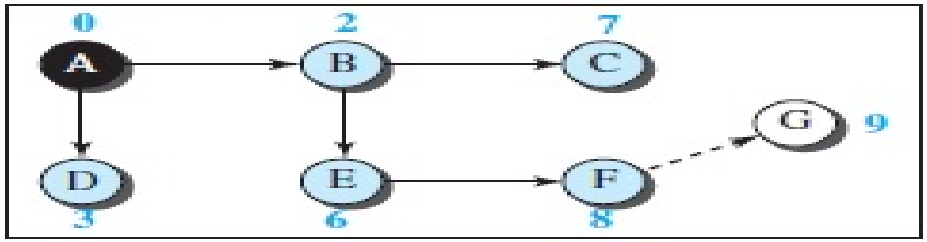
Iteration 3



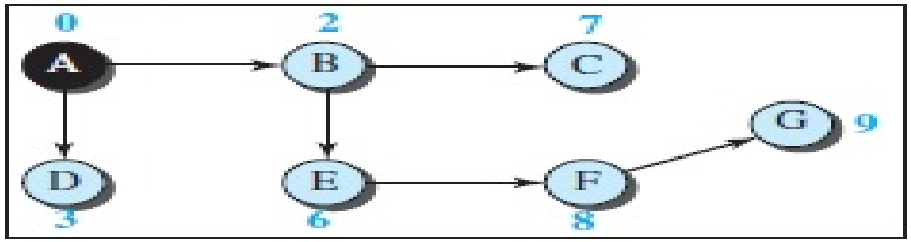
Iteration 4



Iteration 5



Iteration 6



Requirement of 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. For example, a router may belong to an organization that does not provide enough security or it may belong to a commercial rival of the sender which might inspect the packets for obtaining information.
- Least-cost routing does not prevent a packet from passing through an area when that area is in the least-cost path. 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.
- Aside from safety and security, there are occasions, as discussed in the next section, in which the goal of routing is merely reachability: to allow the packet to reach its destination more efficiently without assigning costs to the route.
- To respond to these demands, a third routing algorithm, called path-vector (PV) routing has been devised.

Path vector Routing

- Path-vector routing does not have the drawbacks of LS or DV routing because it is not based on least-cost routing.
- In this routing, The best route is determined by the source using the policy it imposes on the route. In other words, the source can control the path.
- Although path-vector routing is not actually used in an internet, but it is mostly designed to route a packet between ISPs.

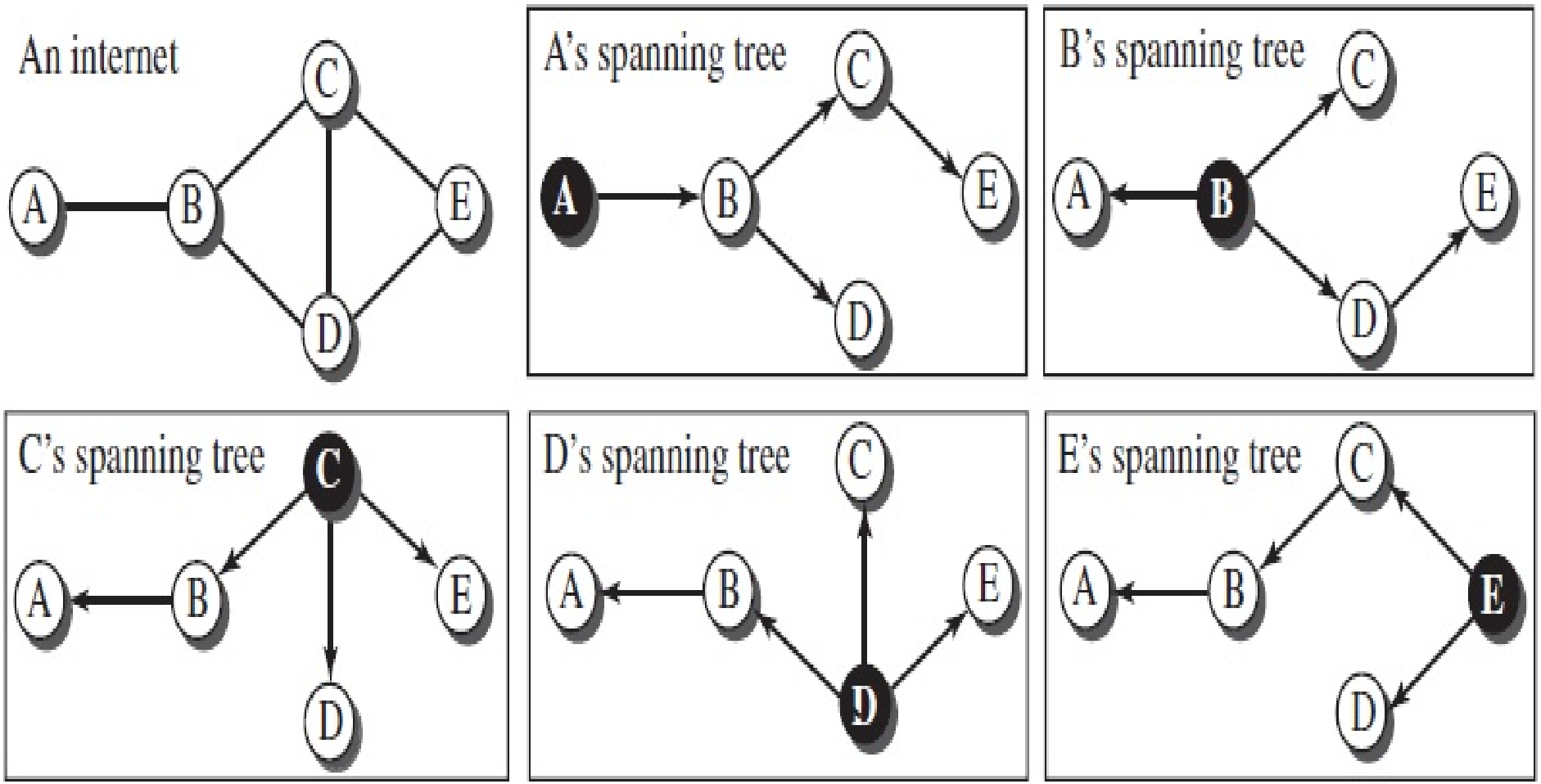
Spanning Trees

- 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.

Path vector Routing : Spanning Trees

- 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 shows a small internet with only five nodes. Each source has created its own spanning tree that meets its policy. The policy imposed by all sources is to use the minimum number of nodes to reach a destination. The spanning tree selected by A and E is such that the communication does not pass through D as a middle node. Similarly, the spanning tree selected by B is such that the communication does not pass through C as a middle node.

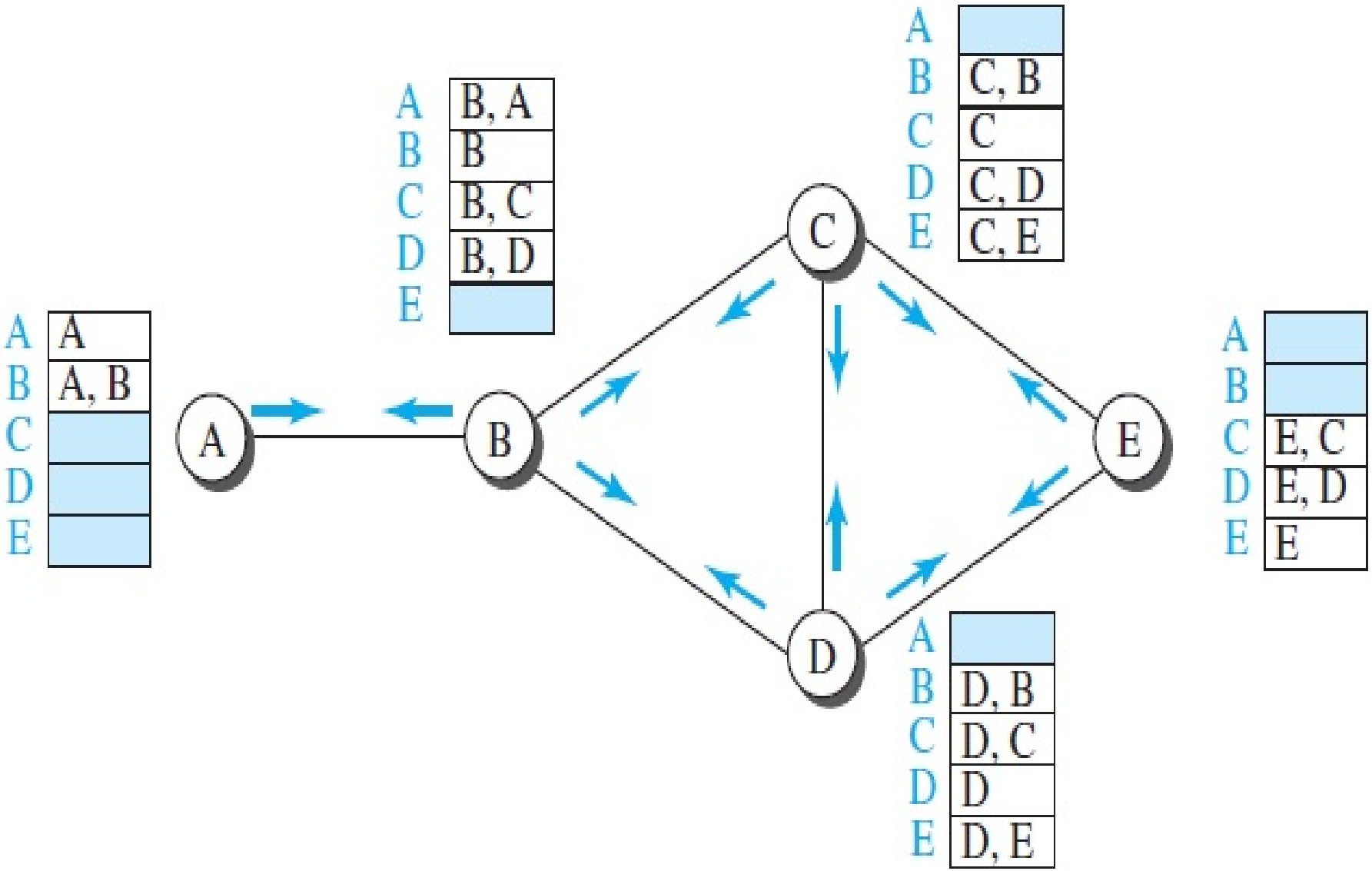
Spanning trees in path-vector routing



Path Vector Routing : Creation of Spanning Trees

- 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.
- Next Figure shows all of these path vectors for our internet in pervious Figure. Note, however, that we do not mean that all of these tables are created simultaneously; they are created when each node is booted. The figure also shows how these path vectors are sent to immediate neighbors after they have been created (arrows).

Path Vectors Made at Booting Time



Path Vector Routing : Creation of Spanning Trees

- Each node, after the creation of the initial path vector, sends it to all its immediate neighbors. Each node, when it receives a path vector from a neighbor, updates its path vector using an equation similar to the Bellman-Ford, but applying its own policy instead of looking for the least cost. We can define this equation as

$$Path\,(x, y) = best\ Path\ (x, y),\ [(x + Path\ (v, y)]\ for\ all\ v's\ in\ internet. \tag{10}$$

- In this equation, the operator $(+)$ means to add x to the beginning of the path. We also need to be cautious to avoid adding a node to an empty path because an empty path means one that does not exist.

Path Vector Routing : Creation of Spanning Trees

- The policy is defined by selecting the best of multiple paths. Path-vector routing also imposes one more condition on this equation: If Path (v, y) includes x, that path is discarded to avoid a loop in the path. In other words, x does not want to visit itself when it selects a path to y.
- Figure shows the path vector of node C after two events. In the first event, node C receives a copy of B's vector, which improves its vector: now it knows how to reach node A. In the second event, node C receives a copy of D's vector, which does not change its vector. As a matter of fact the vector for node C after the first event is stabilized and serves as its forwarding table.

Path Vector Routing : Updating path vectors

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

	New C	Old C	B
A	C, B, A		B, A
B	C, B	C, B	B
C	C	C	B, C
D	C, D	C, D	B, D
E	C, E	C, E	

C[ ] = best (C[ ], C + B[ ])

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

	New C	Old C	D
A	C, B, A	C, B, A	
B	C, B	C, B	D, B
C	C	C	D, C
D	C, D	C, D	D
E	C, E	C, E	D, E

C[ ] = best (C[ ], C + D[ ])

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

Path Vector Algorithm

- Based on the initialization process and the equation used in updating each forwarding table after receiving path vectors from neighbors, we can write a simplified version of the path vector algorithm as shown in Table.
- Lines 4 to 12 show the initialization for the node. Lines 17 to 24 show how the node updates its vector after receiving a vector from the neighbor. The update process is repeated forever. We can see the similarities between this algorithm and the DV algorithm.

Path Vector Algorithm for a Node

```
1   Path_Vector_Routing ( )
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
```

Unicast Routing Protocol

- 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.

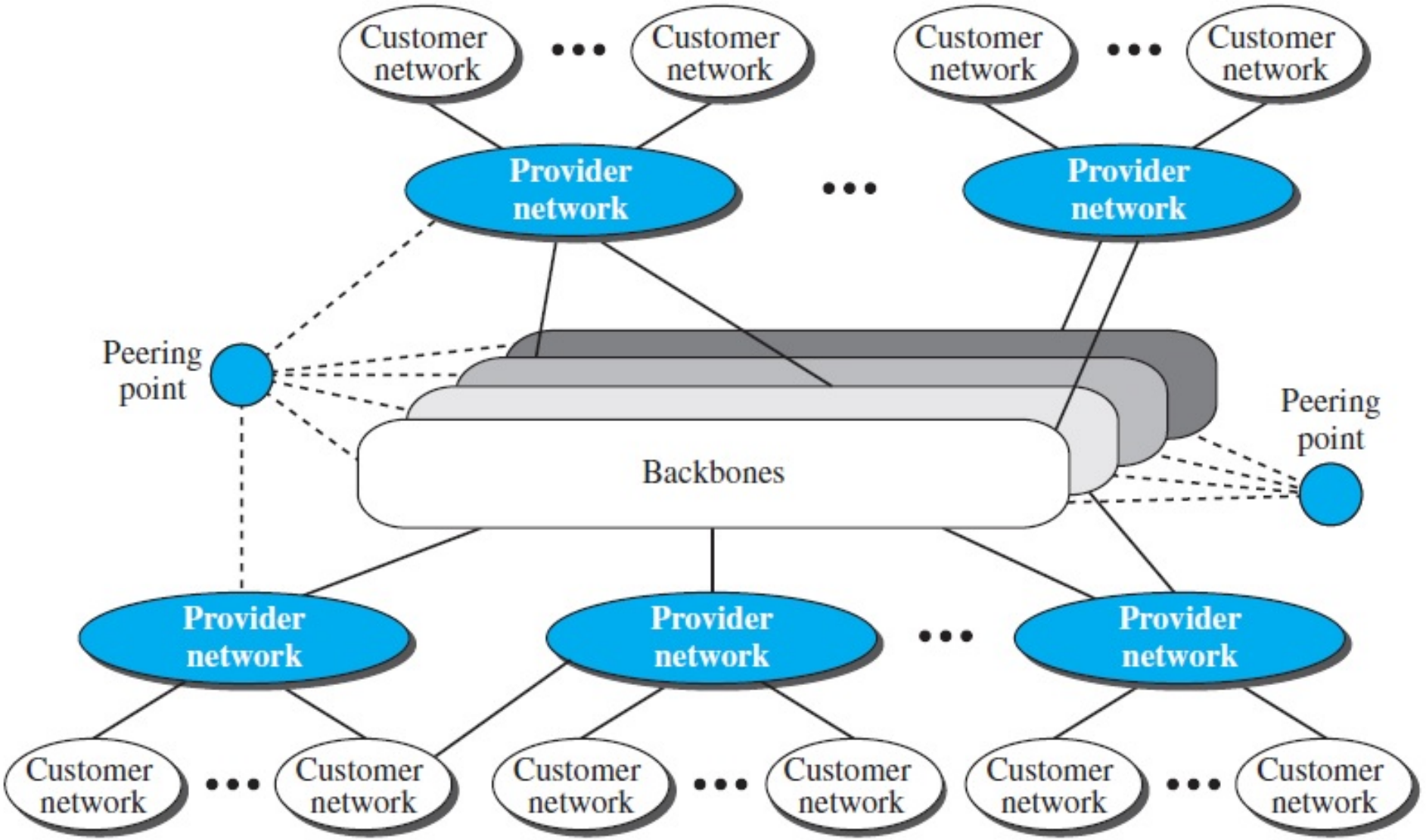
Internet Structure

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.

Internet Structure

- There are several backbones run by private communication companies that provide global connectivity. These backbones are connected by some peering points that allow connectivity between backbones. At a lower level, there are some provider networks that use the backbones for global connectivity but provide services to Internet customers.
- Finally, there are some customer networks that use the services provided by the provider networks. 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.

Internet Structure



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 a 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 Previous Figure . As the figure shows, each ISP is run by an administrative authority. The administrator needs to have control in its system. The organization must be able to use as many subnets and routers as it needs, may desire that the routers be from a particular manufacturer, may wish to run a specific routing algorithm to meet the needs of the organization, and may want to impose some policy on the traffic passing through its ISP.

Hierarchical Routing

- Hierarchical routing means considering each ISP as an autonomous system (AS). Each AS can run a routing protocol that meets its needs, but the global Internet runs a global protocol to glue all ASs together.
- The routing protocol run in each AS is referred to as intra-AS routing protocol, intradomain routing protocol, or interior gateway protocol (IGP); the global routing protocol is referred to as inter-AS routing protocol, interdomain routing protocol, or exterior gateway protocol (EGP).
- We can have several intradomain routing protocols, and each AS is free to choose one, but it should be clear that we should have only one interdomain protocol that handles routing between these entities. Presently, the two common intradomain routing protocols are RIP and OSPF; the only interdomain routing protocol is BGP.

Autonomous Systems

- 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. Each ASN is a 16-bit unsigned integer that uniquely defines an AS.
- The autonomous systems, however, are not categorized according to their size; they are categorized according to the way they are connected to other ASs. We have stub ASs, multihomed ASs, and transient ASs.

Autonomous Systems

The type of AS affects the operation of the interdomain routing protocol in relation to that AS.

- **Stub AS:** A stub AS has only one connection to another AS. The data traffic can be either initiated or terminated in a stub AS; the data cannot pass through it. A good example of a stub AS is the customer network, which is either the source or the sink of data.
- **Multihomed AS:** A multihomed AS can have more than one connection to other ASs, but it does not allow data traffic to pass through it. A good example of such an AS is some of the customer ASs that may use the services of more than one provider network, but their policy does not allow data to be passed through them.
- **Transient AS:** A transient AS is connected to more than one other AS and also allows the traffic to pass through. The provider networks and the backbone are good examples of transient ASs.

Routing Information Protocol (RIP)

- The Routing Information Protocol (RIP) is one of the most widely used intradomain routing protocols based on the distance-vector routing algorithm.
- 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.

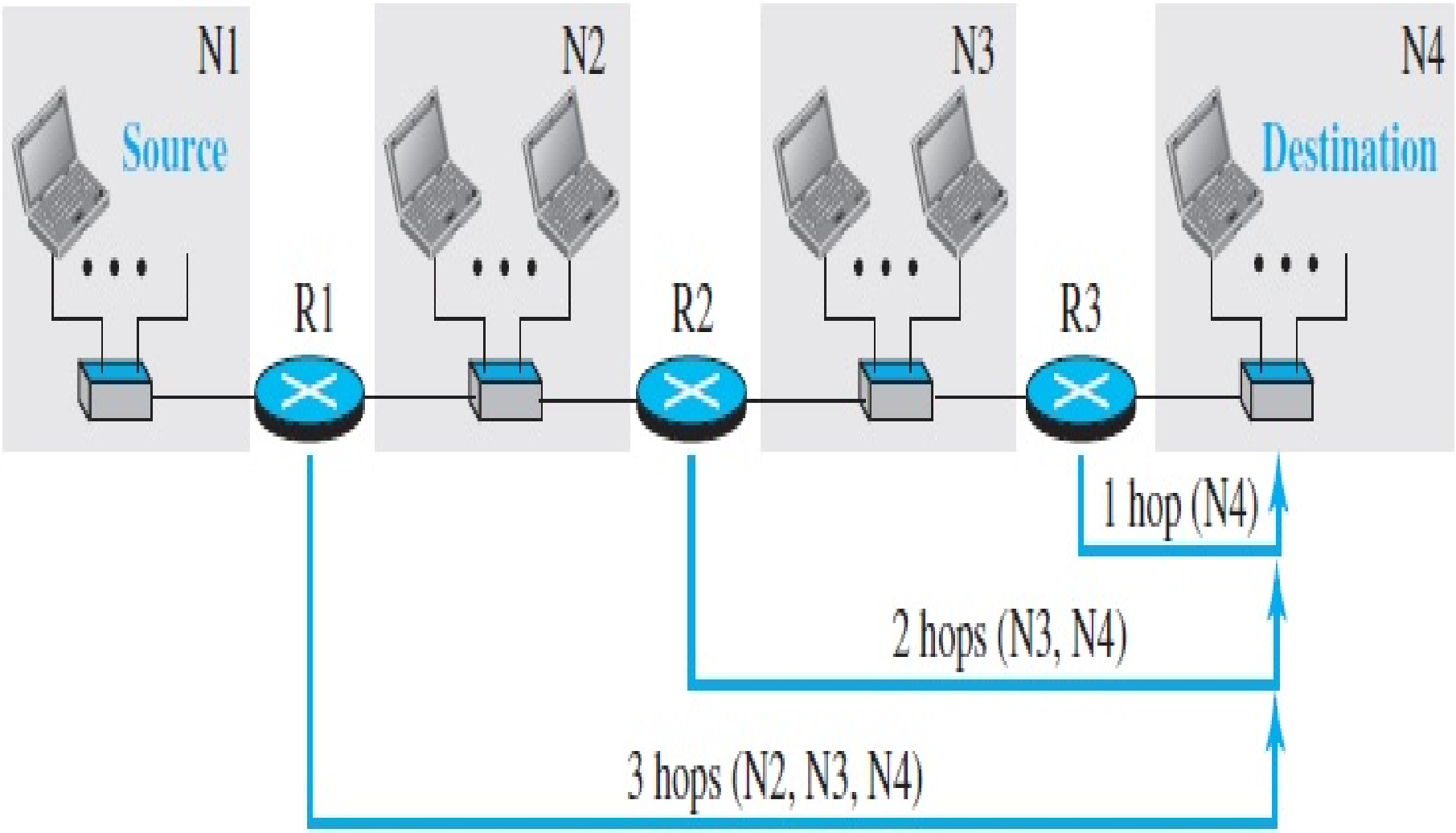
Hop Count

- A router in this protocol basically implements the distance-vector routing algorithm shown in Table. 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.

Hop Count

- Second, to make the implementation of the cost simpler (independent from performance factors of the routers and links, such as delay, bandwidth, and so on), the cost is defined as the number of hops, which means the number of networks (subnets) a packet needs to travel through from the source router to the final destination host.
- Note that the network in which the source host is connected is not counted in this calculation because the source host does not use a forwarding table; the packet is delivered to the default router.
- Figure shows the concept of hop count advertised by three routers from a source host to a destination host. 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 autonomous systems in which the diameter of the AS is not more than 15 hops.

Hop counts in RIP



Forwarding Tables

- Although the distance-vector algorithm 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.
- A forwarding table in RIP is a three-column table in which the first column is the address of the destination network, the second column is the address of the next router to which the packet should be forwarded, and the third column is the cost (the number of hops) to reach the destination network.
- Next Figure shows the three forwarding tables for the routers in Previous Figure. 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.

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

Forwarding Tables

- 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 based on the second property of these trees.
- 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 $\rightarrow$ R2 $\rightarrow$ R3 $\rightarrow$ N4.
- The third column in forwarding table is not needed for forwarding the packet, but it is needed for updating the forwarding table when there is a change in the route.

RIP Implementation

- 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 in the background), named routed (abbreviation for route daemon and pronounced route-dee).
- This means that, although RIP is a routing protocol to help IP 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.
- RIP has gone through two versions: RIP-1 and RIP-2. The second version is backward compatible with the first section; it allows the use of more information in the RIP messages that were set to 0 in the first version. We discuss only RIP-2 in this section.

RIP Algorithm

RIP implements the same algorithm as the distance-vector routing algorithm. However, some changes need to be made to the algorithm to enable a router to update its forwarding table:

- Instead of sending only distance vectors, a router needs to send the whole contents of its forwarding table in a response message.
- The receiver adds one hop to each cost and changes the next router field to the address of the sending router. We call each route in the modified forwarding table the received route and each route in the old forwarding table the old route.
- The received router selects the old routes as the new ones except in the following three cases:

1. If the received route does not exist in the old forwarding table, it should be added to the route.
2. If the cost of the received route is lower than the cost of the old one, the received route should be selected as the new one.

RIP Algorithm

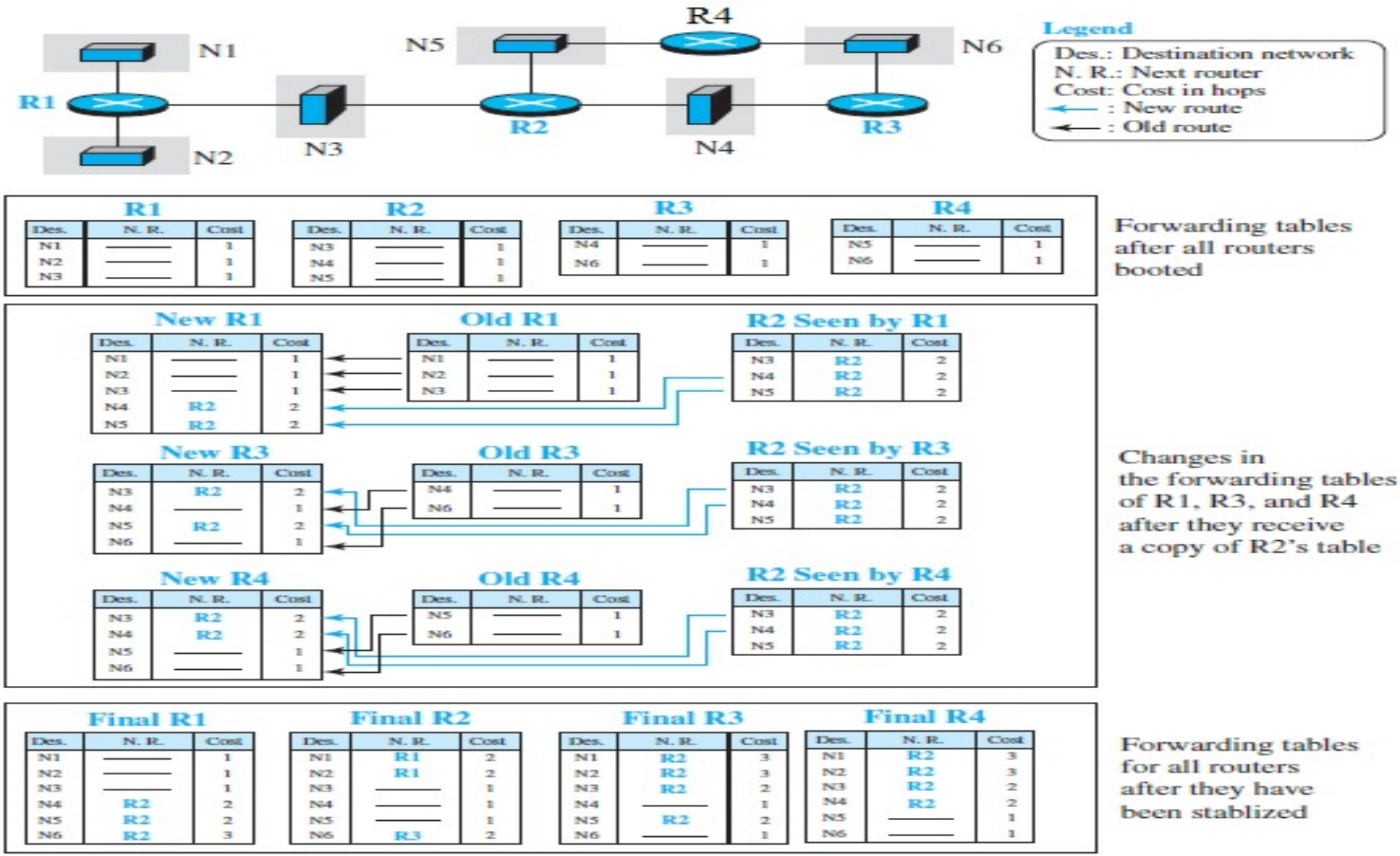
3 If the cost of the received route is higher than the cost of the old one, but the value of the next router is the same in both routes, the received route should be selected as the new one. This is the case where the route was actually advertised by the same router in the past, but now the situation has been changed. For example, suppose a neighbor has previously advertised a route to a destination with cost 3, but now there is no path between this neighbor and that destination. The neighbor advertises this destination with cost value infinity (16 in RIP). The receiving router must not ignore this value even though its old route has a lower cost to the same destination.

- The new forwarding table needs to be sorted according to the destination route (mostly using the longest prefix first).

RIP Algorithm : Example

- Figure 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.

RIP Algorithm : Example



Timers in RIP

- RIP uses three timers to support its operation. The periodic timer controls the advertising of regular update messages. Each router has one periodic timer that is randomly set to a number between 25 and 35 seconds (to prevent all routers sending their messages at the same time and creating excess traffic).
- The timer counts down; when zero is reached, the update message is sent, and the timer is randomly set once again. The expiration timer governs the validity of a route. When a router receives update information for a route, the expiration timer is set to 180 seconds for that particular route. Every time a new update for the route is received, the timer is reset.

Timers in RIP

- If there is a problem on an internet and no update is received within the allotted 180 seconds, the route is considered expired and the hop count of the route is set to 16, which means the destination is unreachable. Every route has its own expiration timer.
- The garbage collection timer is used to purge a route from the forwarding table. When the information about a route becomes invalid, the router does not immediately purge that route from its table. Instead, it continues to advertise the route with a metric value of 16. At the same time, a garbage collection timer is set to 120 seconds for that route. When the count reaches zero, the route is purged from the table. This timer allows neighbors to become aware of the invalidity of a route prior to purging.

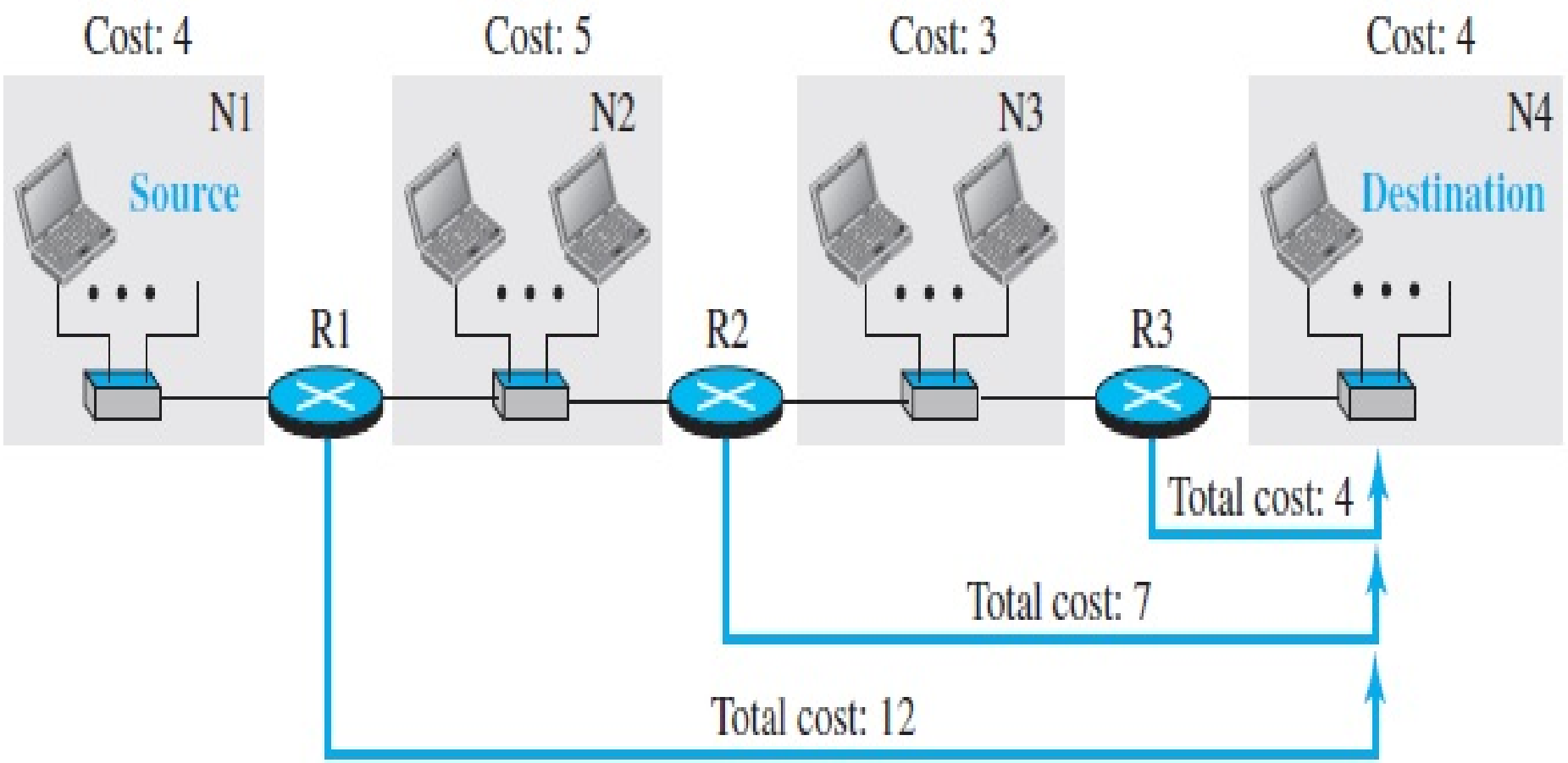
Open Shortest Path First (OSPF)

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

Metric in OSPF

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. An administration can also decide to use the hop count as the cost. An interesting point about the cost in OSPF is that different service types (TOSs) can have different weights as the cost. Figure shows the idea of the cost from a router to the destination host network. We can compare the figure with demonstrate Figure for the RIP.

Open Shortest Path First (OSPF): Metric in OSPF



OSPF :Forwarding Tables

- Each OSPF router can create a forwarding table after finding the shortest-path tree between itself and the destination using Dijkstra's algorithm.
- Next Figure shows the forwarding tables for the simple AS in Previous Figure . Comparing the forwarding tables for the OSPF and RIP in the same AS, we find that the only difference is the cost values.
- In other words, if we use the hop count for OSPF, the tables will be exactly the same. The reason for this consistency is that both protocols use the shortest-path trees to define the best route from a source to a destination.

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

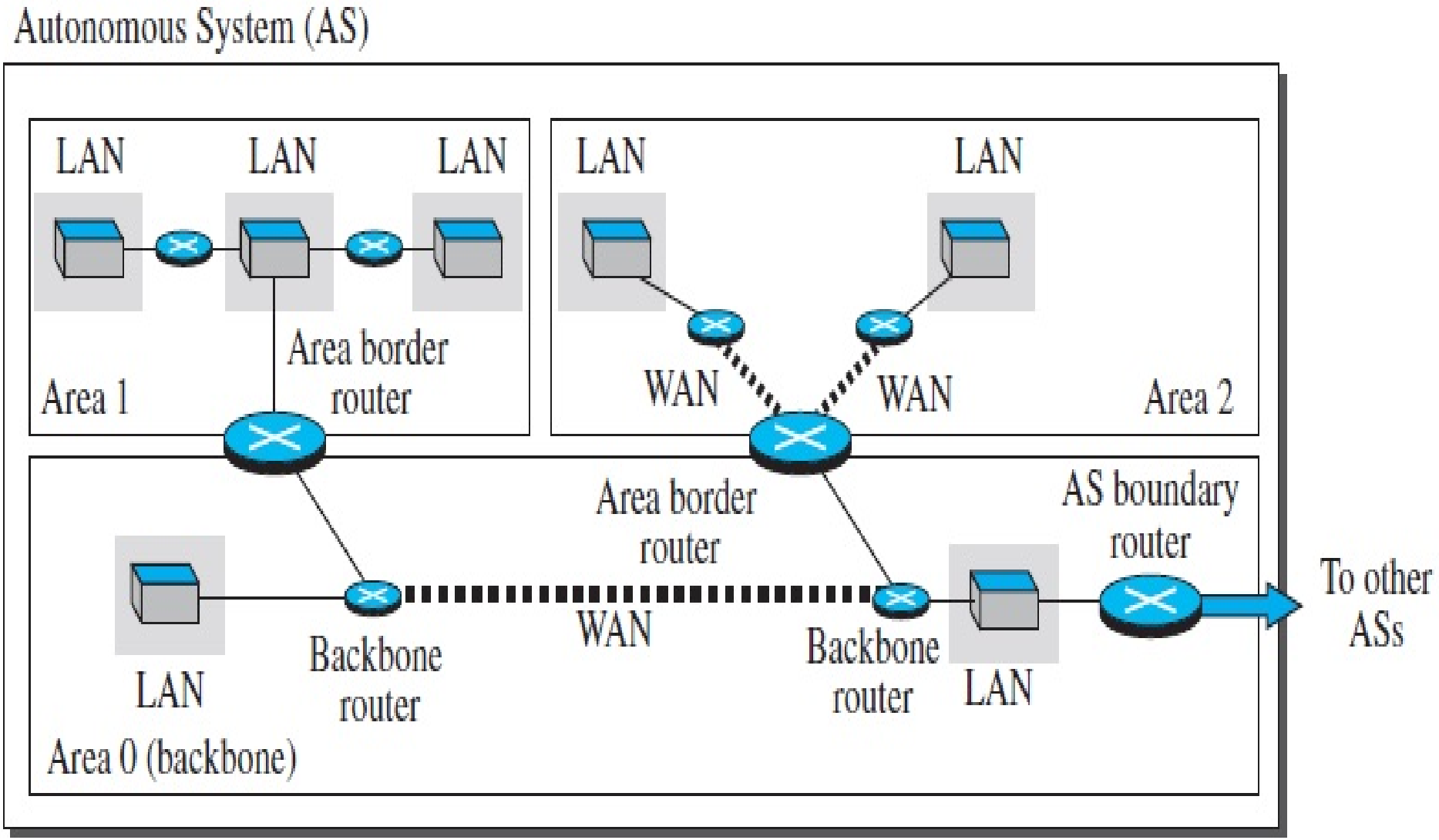
OSPF : Areas

- Compared with RIP, which is normally used in small ASs, OSPF was designed to be able to handle routing in a small or large autonomous system.
- However, the formation of shortest-path trees in OSPF requires that all routers flood the whole AS with their LSPs to create the global LSDB.
- Although this may not create a problem in a small AS, it may have created a huge volume of traffic in a large AS.
- To prevent this, the AS needs to be divided into small sections called areas. Each area acts as a small independent domain for flooding LSPs. In other words, OSPF uses another level of hierarchy in routing: the first level is the autonomous system, the second is the area.

OSPF : Areas

- However, each router in an area needs to know the information about the link states not only in its area but also in other areas.
- For this reason, one of the areas in the AS is designated as the backbone area, responsible for gluing the areas together. The routers in the backbone area are responsible for passing the information collected by each area to all other areas.
- In this way, a router in an area can receive all LSPs generated in other areas. For the purpose of communication, each area has an area identification. The area identification of the backbone is zero.
- Figure shows an autonomous system and its areas.

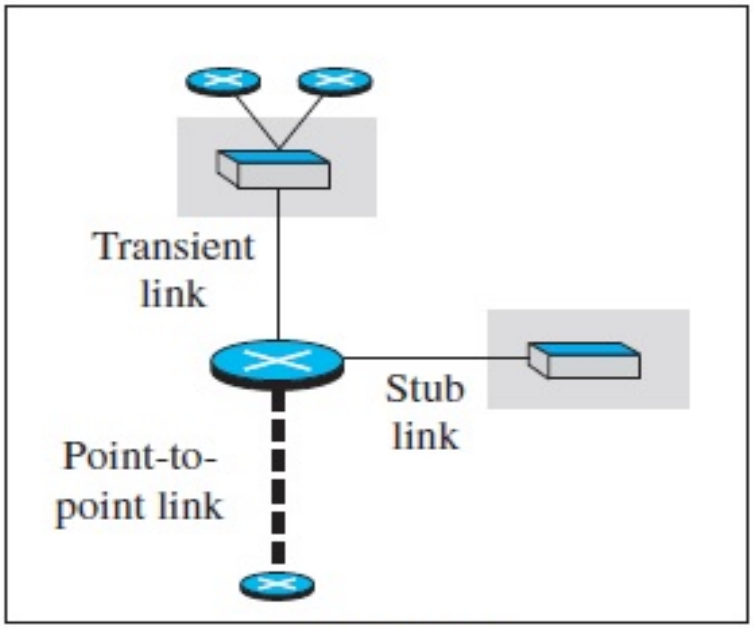
OSPF : Areas in an Autonomous System



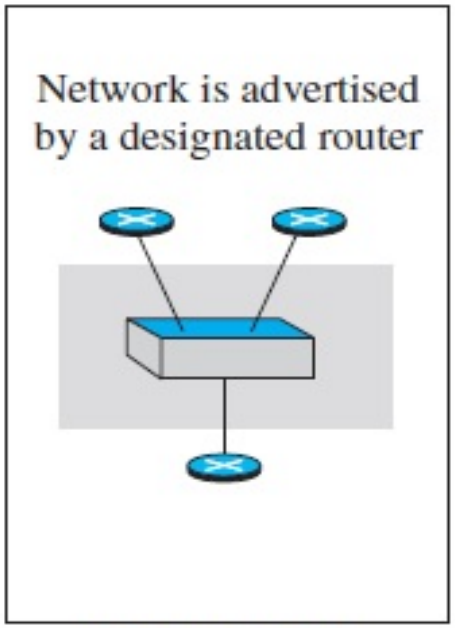
OSPF : Link-State Advertisement

- 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.
- In 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. This means we need different types of advertisements, each capable of advertising different situations.
- We can have five types of link-state advertisements: router link, network link, summary link to network, summary link to AS border router, and external link. Figure shows these five advertisements and their uses.

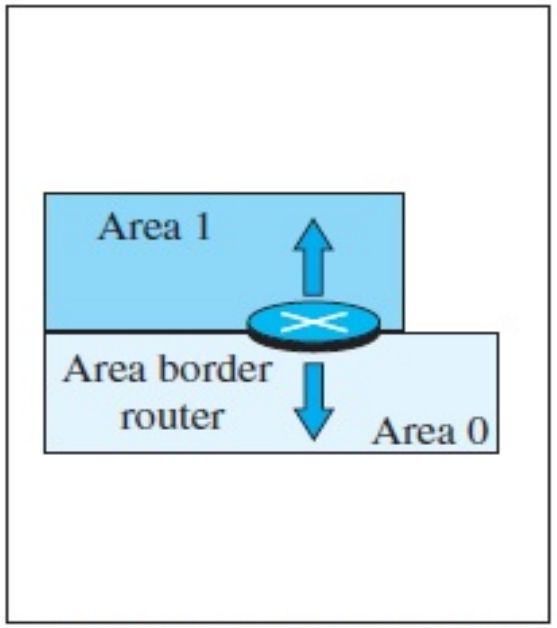
OSPF : Five different LSPs



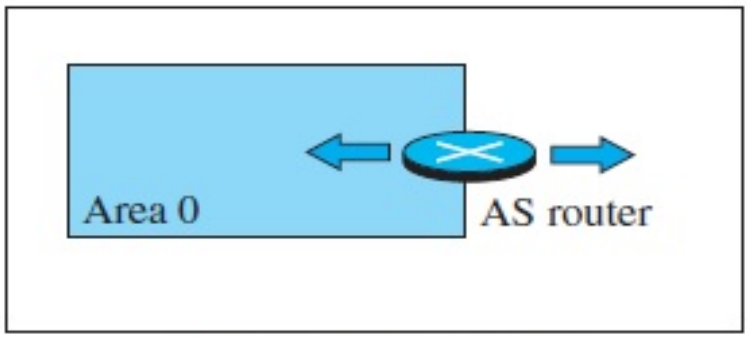
a. Router link



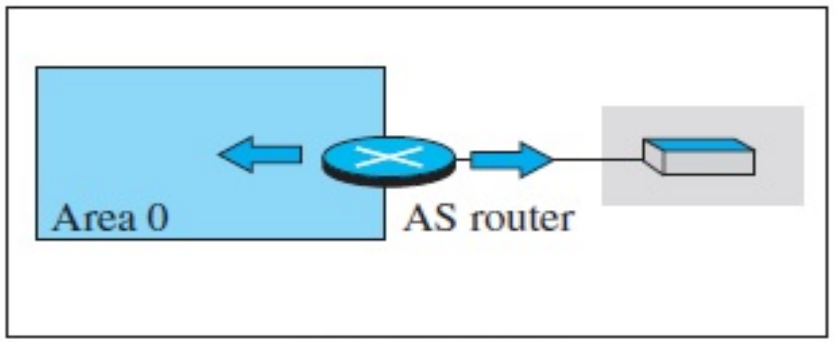
b. Network link



c. Summary link to network



d. Summary link to AS



e. External link

OSPF : Five different LSPs

- **Router link:** A router link advertises the existence of a router as a node. In addition to giving the address of the announcing router, this type of advertisement can define one or more types of links that connect the advertising router to other entities. A transient link announces a link to a transient network, a network that is connected to the rest of the networks by one or more routers. This type of advertisement should define the address of the transient network and the cost of the link. A stub link advertises a link to a stub network, a network that is not a through network. Again, the advertisement should define the address of the network and the cost. A point-to-point link should define the address of the router at the end of the point-to-point line and the cost to get there.

OSPF : Five different LSPs

- **Network link:** A network link advertises the network as a node. However, since a network cannot do announcements itself (it is a passive entity), one of the routers is assigned as the designated router and does the advertising. In addition to the address of the designated router, this type of LSP announces the IP address of all routers (including the designated router as a router and not as speaker of the network), but no cost is advertised because each router announces the cost to the network when it sends a router link advertisement.
- **Summary link to network:** This is done by an area border router; it advertises the summary of links collected by the backbone to an area or the summary of links collected by the area to the backbone. This type of information exchange is needed to glue the areas together.

OSPF : Five different LSPs

- **Summary link to AS:** This is done by an AS router that advertises the summary links from other ASs to the backbone area of the current AS, information which later can be disseminated to the areas so that they will know about the networks in other ASs. The need for this type of information exchange is better understood when we discuss inter-AS routing (BGP
- **External link:** This is also done by an AS router to announce the existence of a single network outside the AS to the backbone area to be disseminated into the areas.

OSPF Implementation

- OSPF is implemented as a program in the network layer, using 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.

OSPF Algorithm

OSPF implements the link-state routing algorithm. However, some changes and augmentations need to be added to the algorithm.

- After each router has created the shortest-path tree, the algorithm needs to use it to create the corresponding routing algorithm.
- The algorithm needs to be augmented to handle sending and receiving all five types of messages.

Border Gateway Protocol Version 4 (BGP4)

- 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.

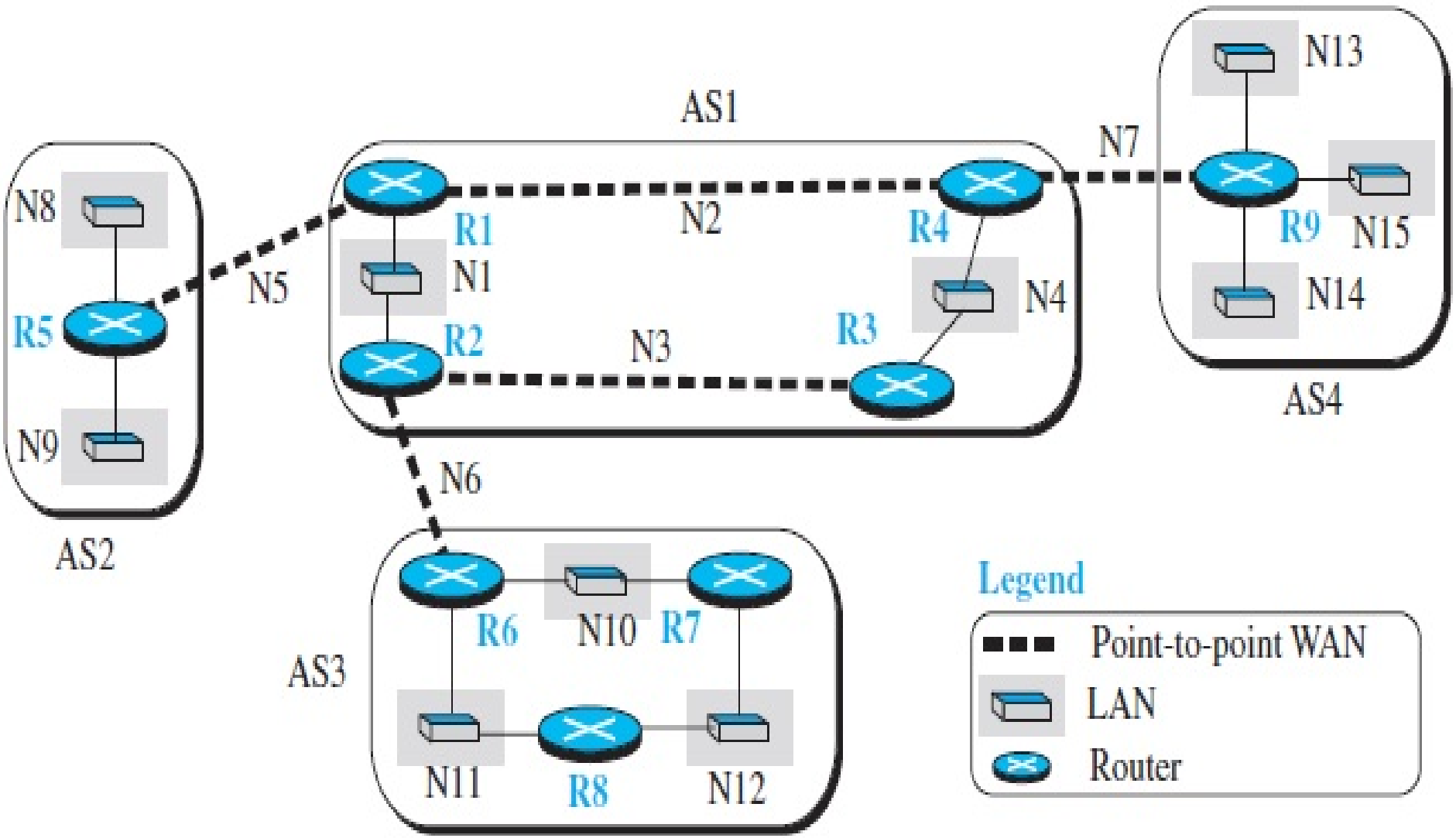
Introduction

BGP, and in particular BGP4, is a complex protocol. We introduce the basics of BGP and its relationship with intradomain routing protocols (RIP or OSPF). Figure 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.

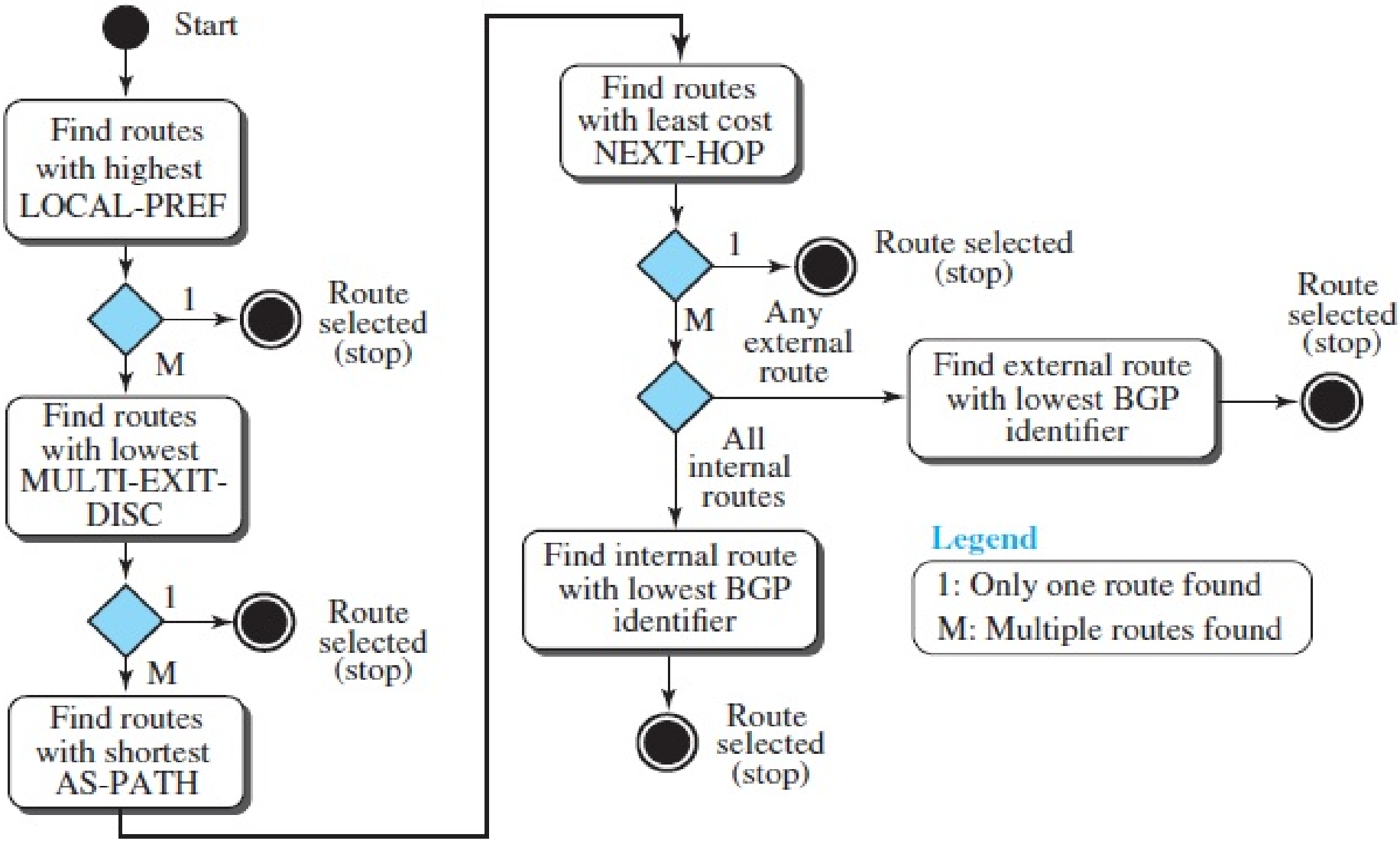
Border Gateway Protocol Version 4 (BGP4): Introduction

- Each autonomous system in this figure uses one of the two common intradomain protocols, RIP or OSPF. Each router in each AS knows how to reach a network that is in its own AS, but it does not know how to reach a network in another AS.
- To enable each router to route a packet to any network in the internet, we first install a variation of BGP4, called external BGP (eBGP), on each border router (the one at the edge of each AS which is connected to a router at another AS). We then install the second variation of BGP, called internal BGP (iBGP), on all routers. This means that the border routers will be running three routing protocols (intradomain, eBGP, and iBGP), but other routers are running two protocols (intradomain and iBGP). We discuss the effect of each BGP variation separately.

Border Gateway Protocol Version 4 (BGP4): Introduction



BGP: Flow diagram for route selection



The End