USN

Internal Assessment Test 2 – December 2023

Sub:	COMPUTER NETWORKS					Sub Code:	18EC71	Branch:	ECE
Date:	04-12-2023	Duration:	90 minutes	Max Marks:	50	Sem/Sec:	7th (A,B,C,D)		OBE

	ANSWER ANY 5 FULL QUESTIONS	MARKS	CO	RBT
1	Explain CSMA/CA with suitable figures and flow diagram.	10	CO2	L3
2	Explain all fields of IPV4 datagram format with a neat figure.	10	CO2, CO3	L2
3	a) Distinguish between class A, class B and class C addressing. b) Find the class of the following IP addresses: i) 237.14.2.1 ii) 208.25.54.12 iii) 129.14.6.8 iv) 114.34.2.9 c) Identify if the following 802.3 MAC addresses are unicast, multicast or broadcast. 47:20:1B:2E:08:EE EE:FF:10:01:11:00 FF:FF:FF:FF:FF:FF	3 + 4 + 3	CO2, CO3	L2

(P.T.O)

USN

Internal Assessment Test 2 – December 2023

Sub:	COMPUTER NETWORKS					Sub Code:	18EC71	Branch:	ECE
Date:	04-12-2023	Duration:	90 minutes	Max Marks:	50	Sem/Sec:	7th (A,B,C,D)		OBE

	ANSWER ANY 5 FULL QUESTIONS	MARKS	CO	RBT
1	Explain CSMA/CA with suitable figures and flow diagram.	10	CO2	L3
2	Explain all fields of IPV4 datagram format with a neat figure.	10	CO2, CO3	L2
3	a) Distinguish between class A, class B and class C addressing. b) Find the class of the following IP addresses: i) 237.14.2.1 ii) 208.25.54.12 iii) 129.14.6.8 iv) 114.34.2.9 c) Identify if the following 802.3 MAC addresses are unicast, multicast or broadcast. 47:20:1B:2E:08:EE EE:FF:10:01:11:00	3 + 4 + 3	CO2, CO3	L2

	FF:FF:FF:FF:FF:FF **(P.T.O)**			
4	Explain distance vector routing algorithm using bellman ford equations.	10	CO2, CO3	L3
5	An organization is granted a block of addresses with the beginning address 14.24.74.0/24. The organization needs to have 3 sub blocks of addresses to use in its three subnets: one sub block of 10 addresses, one sub block of 60 addresses, and one sub block of 120 addresses. Design the sub blocks. Find out the total number of unused addresses.	10	CO2, CO3	L3
6	a) Write a note on ethernet frame format with suitable figure. What is the minimum and maximum length? b) List the characteristics of Wireless LANs and explain them.	6 + 4	CO2	L1
7	a) A pure ALOHA network transmits 200-bit frames on a shared channel of 200 kbps. What is the throughput if the system (all stations together) produces a. 1000 frames per second? b. 500 frames per second? c. 250 frames per second? b) A block of addresses is granted to a small organization. One of the addresses is 205.16.37.39/28. What is the first address, last address and number of addresses in a block.	5 + 5	CO2, CO3	L3

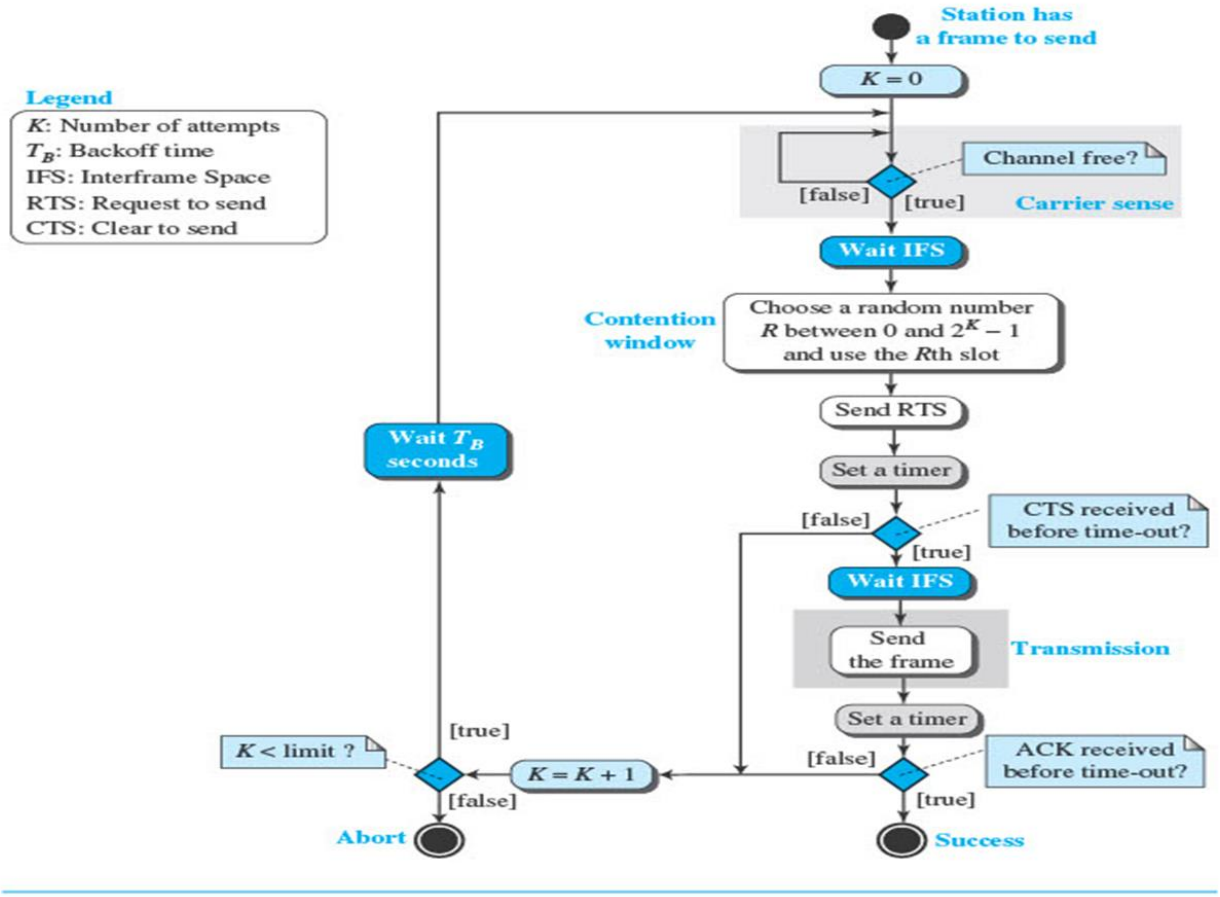
Course Instructor **Chief Course Instructor** **HOD**

4	Explain distance vector routing algorithm using bellman ford equations.	10	CO2, CO3	L3
5	An organization is granted a block of addresses with the beginning address 14.24.74.0/24. The organization needs to have 3 sub blocks of addresses to use in its three subnets: one sub block of 10 addresses, one sub block of 60 addresses, and one sub block of 120 addresses. Design the sub blocks. Find out the total number of unused addresses.	10	CO2, CO3	L3
6	a) Write a note on ethernet frame format with suitable figure. What is the minimum and maximum length? b) List the characteristics of Wireless LANs and explain them.	6 + 4	CO2	L1
7	a) A pure ALOHA network transmits 200-bit frames on a shared channel of 200 kbps. What is the throughput if the system (all stations together) produces a. 1000 frames per second? b. 500 frames per second? c. 250 frames per second? b) A block of addresses is granted to a small organization. One of the addresses is 205.16.37.39/28. What is the first address, last address and number of addresses in a block.	5 + 5	CO2, CO3	L3

Q1) Solution :

Carrier sense multiple access with collision avoidance (CSMA/CA) was invented for wireless networks.
Collisions are avoided through the use of CSMA/CA's three strategies: the interframe space, the contention window, and acknowledgments, as shown in Figure 12.15.
We discuss RTS and CTS frames later.

Figure 12.15 *Flow diagram of CSMA/CA*

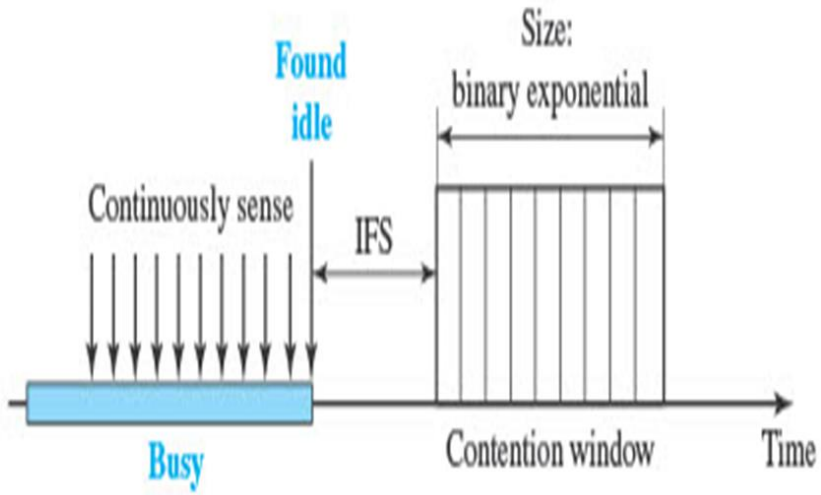


Interframe Space (IFS): First, collisions are avoided by deferring transmission even if the channel is found idle.
When an idle channel is found, the station does not send immediately.
It waits for a period of time called the interframe space or IFS.
Even though the channel may appear idle when it is sensed, a distant station may have already started transmitting.
The distant station's signal has not yet reached this station.
The IFS time allows the front of the transmitted signal by the distant station to reach this station.
After waiting an IFS time, if the channel is still idle, the station can send, but it still needs to wait a time equal to the contention window (described next).
The IFS variable can also be used to prioritize stations or frame types.
For example, a station that is assigned a shorter IFS has a higher priority.

Contention Window: The contention window is an amount of time divided into slots.
A station that is ready to send chooses a random number of slots as its wait time. The number of slots in the window changes according to the binary exponential backoff strategy.
This means that it is set to one slot the first time and then doubles each time the station cannot detect an idle channel after the

IFS time.
This is very similar to the p-persistent method except that a random outcome defines the number of slots taken by the waiting station.
One interesting point about the contention window is that the station needs to sense the channel after each time slot.
However, if the station finds the channel busy, it does not restart the process; it just stops the timer and restarts it when the channel is sensed as idle.
This gives priority to the station with the longest waiting time. See Figure 12.16.

Figure 12.16 *Contention window*

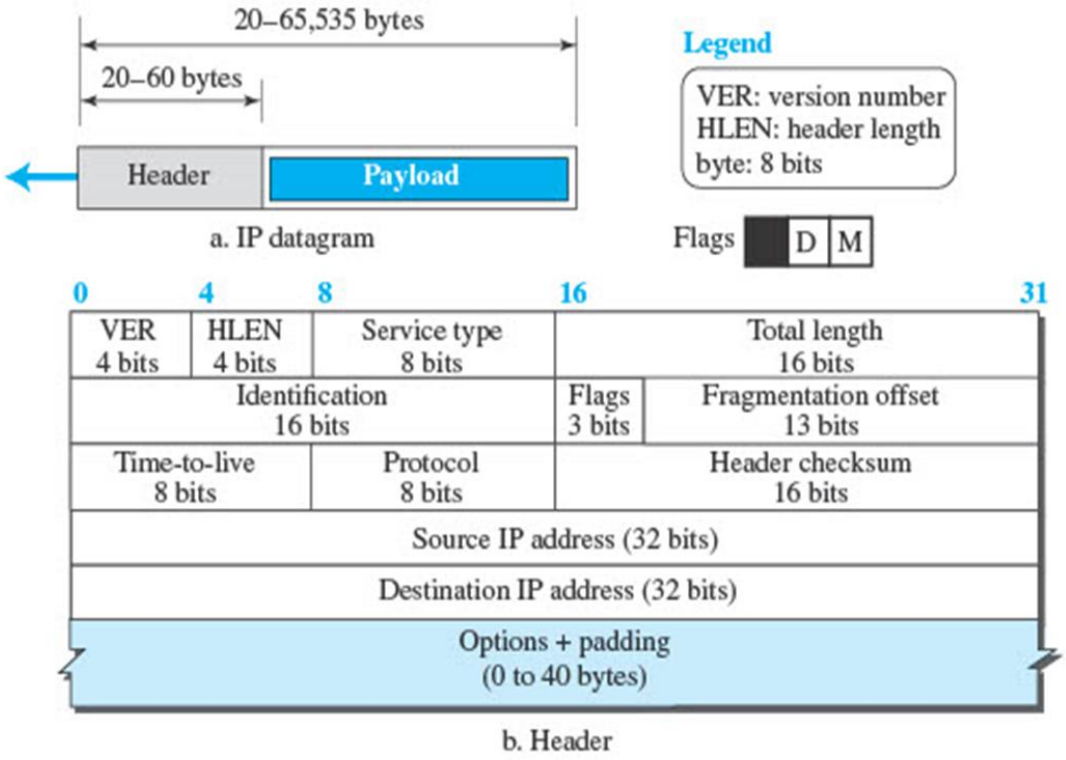


Acknowledgment: With all these precautions, there still may be a collision resulting in destroyed data.
In addition, the data may be corrupted during the transmission.
The positive acknowledgment and the time-out timer can help guarantee that the receiver has received the frame.

Q2) Solution :

In this section, we begin by discussing the first service provided by IPv4, packetizing.
We show how IPv4 defines the format of a packet in which the data coming from the upper layer or other protocols are encapsulated.
Packets used by the IP are called datagrams.
Figure 19.2 shows the IPv4 datagram format.
A datagram is a variable-length packet consisting of two parts: header and payload (data).
The header is 20 to 60 bytes in length and contains information essential to routing and delivery.
It is customary in TCP/IP to show the header in 4-byte sections.

Figure 19.2 *IP datagram*



Discussing the meaning and rationale for the existence of each field is essential to understanding the operation of IPv4; a brief description of each field is in order.
Version Number: The 4-bit version number (VER) field defines the version of the IPv4 protocol, which, obviously, has the value of 4.
Header Length: The 4-bit header length (HLEN) field defines the total length of the datagram header in 4-byte words. The IPv4 datagram has a variable-length header. When a device receives a datagram, it needs to know when the header stops and the data, which is encapsulated in the packet, starts. However, to make the value of the header length (number of bytes) fit in a 4-bit header length, the total length of the header is calculated as 4-byte words. The total length is divided by 4 and the value is inserted in the field. The receiver needs to multiply the value of this field by 4 to find the total length.
Service Type: In the original design of the IP header, this field was referred to as type of service (TOS), which defined how the datagram should be handled. In the late 1990s, IETF redefined the field to provide differentiated services (DiffServ).

When we discuss differentiated services in Chapter 30, we will be in a better situation to define the bits in this field.
The use of 4-byte words for the length header is also logical because the IP header always needs to be aligned in 4-byte boundaries.
Total Length: This 16-bit field defines the total length (header plus data) of the IP datagram in bytes. A 16-bit number can define a total length of up to 65,535 (when all bits are 1s). However, the size of the datagram is normally much less than this. This field helps the receiving device to know when the packet has completely arrived. To find the length of the data coming from the upper layer, subtract the header length from the total length. The header length can be found by multiplying the

value in the HLEN field by 4.
Length of data = total length − (HLEN) × 4

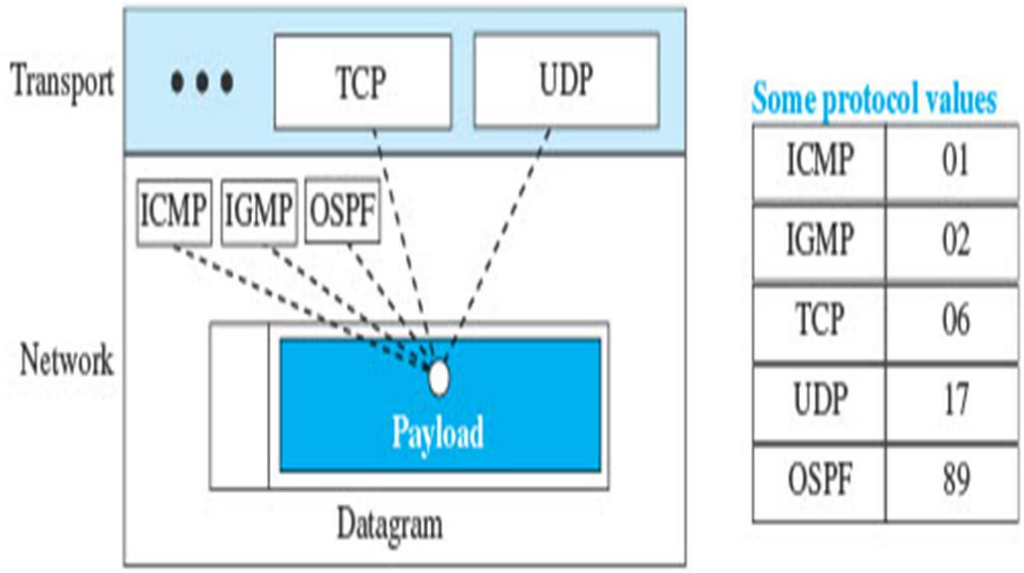
Though a size of 65,535 bytes might seem large, the size of the IPv4 datagram may increase in the near future as the underlying technologies allow even more throughput (greater bandwidth).
One may ask why we need this field anyway.
When a machine (router or host) receives a frame, it drops the header and the trailer, leaving the datagram.
Why include an extra field that is not needed?

The answer is that in many cases we really do not need the value in this field.
However, there are occasions in which the datagram is not the only thing encapsulated in a frame; it may be that padding has been added.
For example, the Ethernet protocol has a minimum and maximum restriction on the size of data that can be encapsulated in a frame (46 to 1500 bytes).
If the size of an IPv4 datagram is less than 46 bytes, some padding will be added to meet this requirement.
In this case, when a machine decapsulates the datagram, it needs to check the total length field to determine how much is really data and how much is padding.

Identification, Flags, and Fragmentation Offset: These three fields are related to the fragmentation of the IP datagram when the size of the datagram is larger than the underlying network can carry. We discuss the contents and importance of these fields when we talk about fragmentation in the next section.
Time-to-live: Due to some malfunctioning of routing protocols (discussed later) a datagram may be circulating in the Internet, visiting some networks over and over without reaching the destination. This may create extra traffic in the Internet. The time-to-live (TTL) field is used to control the maximum number of hops (routers) visited by the datagram. When a source host sends the datagram, it stores a number in this field. This value is approximately two times the maximum number of routers between any two hosts. Each router that processes the datagram decrements this number by one. If this value, after being decremented, is zero, the router discards the datagram.
Protocol: In TCP/IP, the data section of a packet, called the payload, carries the whole packet from another protocol. A datagram, for example, can carry a packet belonging to any transport-layer protocol such as UDP or TCP. A datagram can also carry a packet from other protocols that directly use the service of the IP, such as some routing protocols or some auxiliary protocols.

The Internet authority has given any protocol that uses the service of IP a unique 8-bit number which is inserted in the protocol field.
When the payload is encapsulated in a datagram at the source IP, the corresponding protocol number is inserted in this field; when the datagram arrives at the destination, the value of this field helps to define to which protocol the payload should be delivered.
In other words, this field provides multiplexing at the source and demultiplexing at the destination, as shown in Figure 19.3.
Note that the protocol fields at the network layer play the same role as the port numbers at the transport layer (Chapters 23 and 24).
However, we need two port numbers in a transport-layer packet because the port numbers at the source and destination are different, but we need only one protocol field because this value is the same for each protocol no matter whether it is located at the source or the destination.

Figure 19.3 *Multiplexing and demultiplexing using the value of the protocol field*



IP is not a reliable protocol; it does not check whether the payload carried by a datagram is corrupted during the transmission.
IP puts the burden of error checking of the payload on the protocol that owns the payload, such as UDP or TCP.
The datagram header, however, is added by IP, and its error-checking is the responsibility of IP.
Errors in the IP header can be a disaster.
For example, if the destination IP address is corrupted, the packet can be delivered to the wrong host.
If the protocol field is corrupted, the payload may be delivered to the wrong protocol.
If the fields related to the fragmentation are corrupted, the datagram cannot be reassembled correctly at the destination, and so on.
For these reasons, IP adds a header checksum field to check the header, but not the payload.
We need to remember that, since the value of some fields, such as TTL, which are related to fragmentation and options, may change from router to router, the checksum needs to be recalculated at each router.
As we discussed in Chapter 10, checksum in the Internet normally uses a 16-bit field, which is the complement of the sum of other fields calculated using 1s complement arithmetic.

These 32-bit source and destination address fields define the IP address of the source and destination respectively.
The source host should know its IP address.
The destination IP address is either known by the protocol that uses the service of IP or is provided by the DNS as described in Chapter 26.
Note that the value of these fields must remain unchanged during the time the IP datagram travels from the source host to the destination host.
IP addresses were discussed in Chapter 18.

A datagram header can have up to 40 bytes of options.
Options can be used for network testing and debugging.
Although options are not a required part of the IP header, option processing is required of the IP software.
This means that all implementations must be able to handle options if they are present in the header.
The existence of options in a header creates some burden on the datagram handling; some options can be changed by routers, which forces each router to recalculate the header checksum.
There are one-byte and multi-byte options that we will briefly discuss later in the chapter.
The complete discussion is posted at the book website.

Payload, or data, is the main reason for creating a datagram.

Payload is the packet coming from other protocols that use the service of IP.
Comparing a datagram to a postal package, payload is the content of the package; the header is only the information written on the package.

Q3)a) Solution :

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 27 = 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 214 = 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 221 = 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.

Q3)b) Solution :

i) 237.14.2.1 is class D
ii) 208.25.54.12 is class C
iii) 129.14.6.8 is class B
iv) 114.34.2.9 is class A

Q3) c) Solution :

47:20:1B:2E:08:EE is multicast

EE:FF:10:01:11:00 is unicast

FF:FF:FF:FF:FF:FF is broadcast

Q4) Solution :

The distance-vector (DV) routing uses the goal we discussed in the introduction, to find the best route.
In distance-vector routing, the first thing each node creates is its own least-cost tree with the rudimentary information it has about its immediate neighbors.
The incomplete trees are exchanged between immediate neighbors to make the trees more and more complete and to represent the whole internet.
We can say that in distance-vector routing, a router continuously tells all of its neighbors what it knows about the whole internet (although the knowledge can be incomplete).
Before we show how incomplete least-cost trees can be combined to make complete ones, we need to discuss two important topics: the Bellman-Ford equation and the concept of distance vectors, which we cover next.

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 Dij is the shortest distance and cij is the cost between nodes i and j.

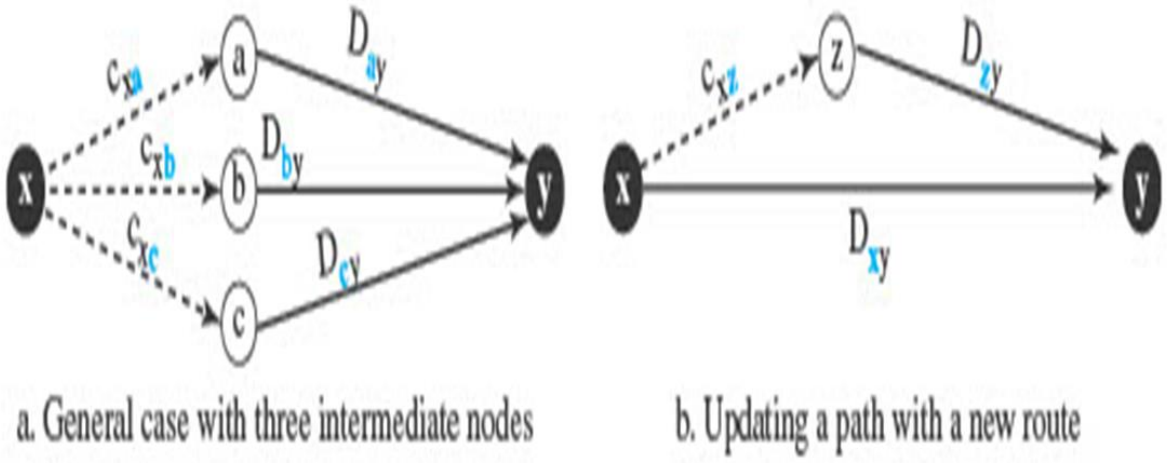
$$D_{xy} = \min\{(c_{xa} + D_{ay}), (c_{xb} + D_{by}), (c_{xc} + D_{cy}), \ldots\}$$

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:

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

Figure 20.3 shows the idea graphically for both cases.

Figure 20.3 *Graphical idea behind Bellman-Ford equation*



a. General case with three intermediate nodes

b. Updating a path with a new route

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 20.3, 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, but this use may be in the background.
We will shortly show how we use the Bellman-Ford equation and the concept of distance vectors to build least-cost paths for each node in distance-vector routing, but first we need to discuss the concept of a distance vector.

Now we can give a simplified pseudocode for the distance-vector routing algorithm, as shown in Table 20.1.
The algorithm is run by its node independently and asynchronously.

Table 20.1 Distance-Vector Routing Algorithm for a Node

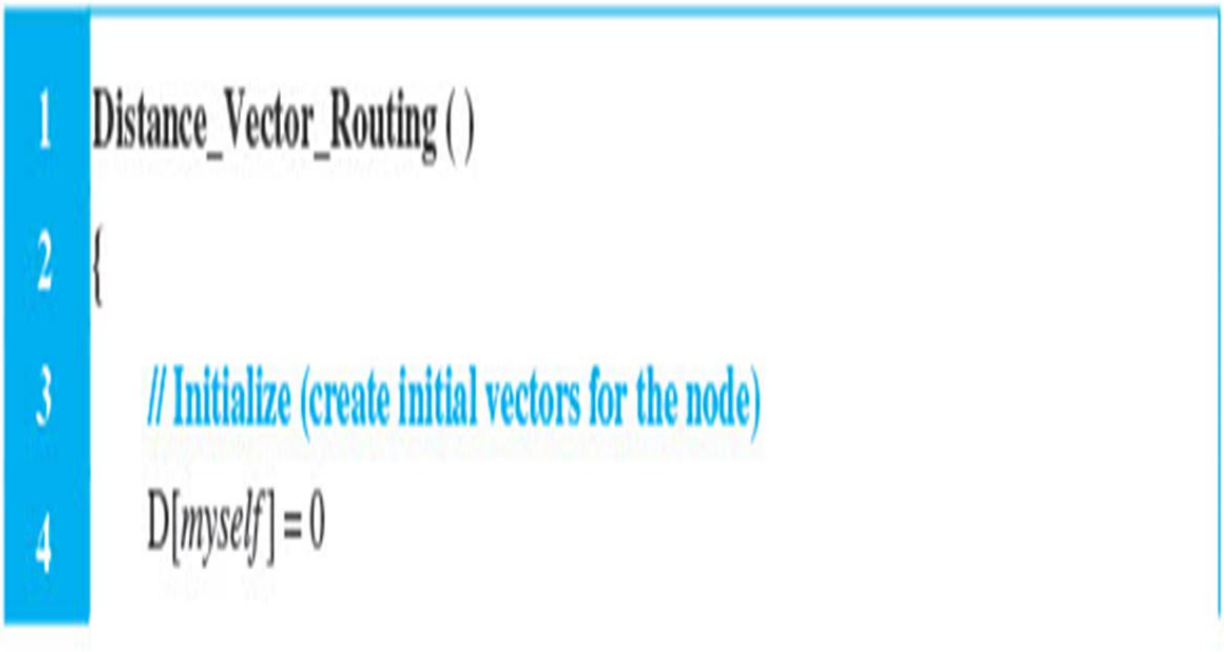


Table 20.1 *Distance-Vector Routing Algorithm for a Node (continued)*

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

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.

Q5) Solution :

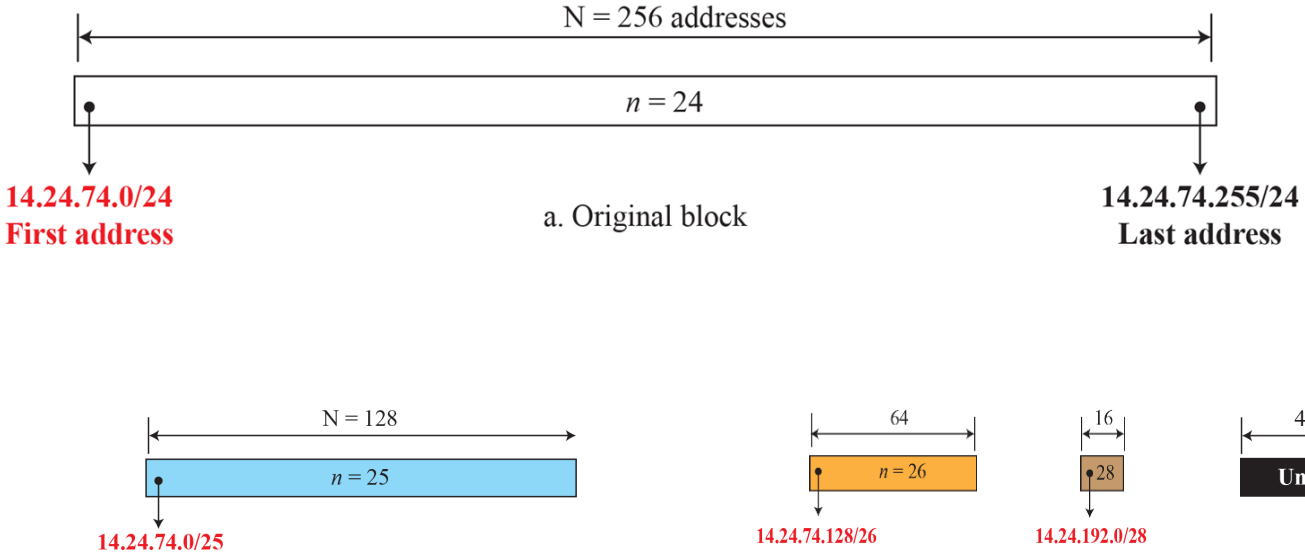
There are 232– 24 = 256 addresses in this block. The first address is 14.24.74.0/24; the last address is 14.24.74.255/24. To satisfy the third requirement, we assign addresses to subblocks, starting with the largest and ending with the smallest one.

a. 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 n1 = 32 − log2 128 = 25. The first address in this block is 14.24.74.0/25; the last address is 14.24.74.127/25.

b. 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 n2 = 32 − log2 64 = 26. The first address in this block is 14.24.74.128/26; the last address is 14.24.74.191/26.

c. The number of addresses in the smallest subblock, which requires 10 addresses, is not a power of 2. We allocate 16 addresses. The subnet mask for this subnet can be found as n1 = 32 − log2 16 = 28. The first address in this block is 14.24.74.192/28; the last address is 14.24.74.207/28.

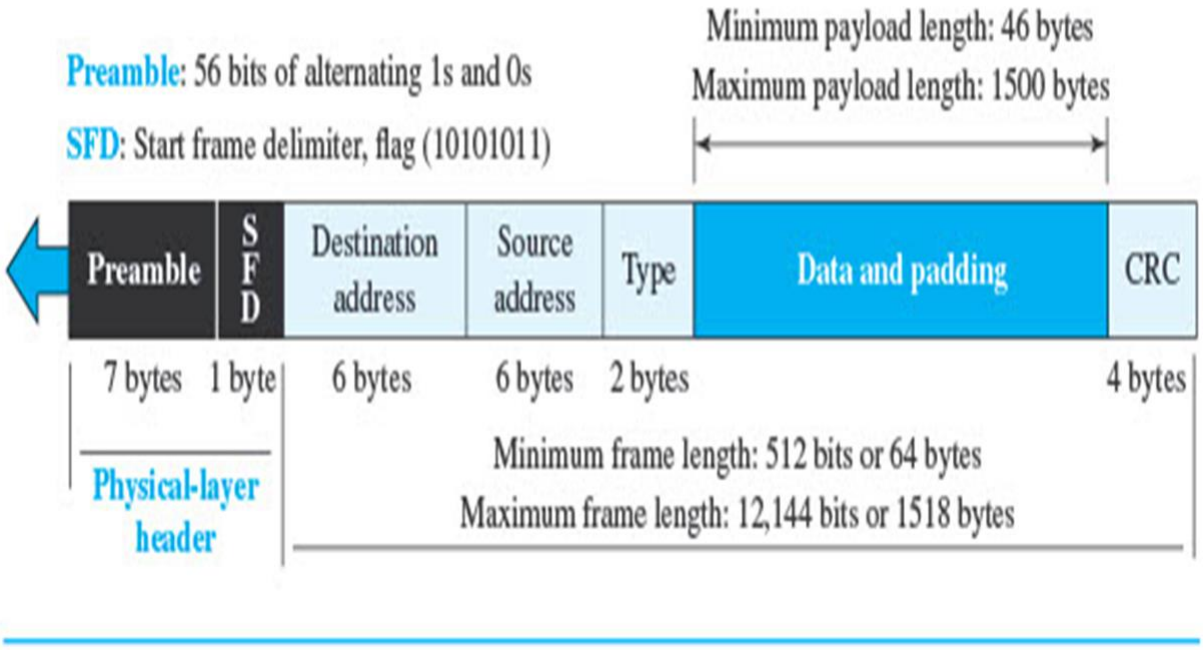
If we add all addresses in the previous subblocks, the result is 208 addresses, which means 48 addresses are left in reserve. The first address in this range is 14.24.74.208. The last address is 14.24.74.255. We don't know about the prefix length yet. Figure 18.23 shows the configuration of blocks. We have shown the first address in each block.



Q6) a) Solution :

The Ethernet frame contains seven fields, as shown in Figure 13.3.
Preamble: This field contains 7 bytes (56 bits) of alternating 0s and 1s that alert the receiving system to the coming frame and enable it to synchronize its clock if it's out of synchronization.
The pattern provides only an alert and a timing pulse.
The 56-bit pattern allows the stations to miss some bits at the beginning of the frame.
The preamble is actually added at the physical layer and is not (formally) part of the frame.

Figure 13.3 *Ethernet frame*



Start frame delimiter (SFD): This field (1 byte: 10101011) signals the beginning of the frame.
The SFD warns the station or stations that this is the last chance for synchronization.
The last 2 bits are (11)2 and alert the receiver that the next field is the destination address.
This field is actually a flag that defines the beginning of the frame.
We need to remember that an Ethernet frame is a variable-length frame.
It needs a flag to define the beginning of the frame.
The SFD field is also added at the physical layer.

Destination address (DA): This field is six bytes (48 bits) and contains the link-layer address of the destination station or stations to receive the packet.
We will discuss addressing shortly.
When the receiver sees its own link-layer address, or a multicast address for a group that the receiver is a member of, or a broadcast address, it decapsulates the data from the frame and passes the data to the upper layer protocol defined by the value of the type field.

Source address (SA): This field is also six bytes and contains the link-layer address of the sender of the packet. We will discuss addressing shortly.
Type: This field defines the upper-layer protocol whose packet is encapsulated in the frame.
This protocol can be IP, ARP, OSPF, and so on.
In other words, it serves the same purpose as the protocol field in a datagram and the port number in a segment or user datagram.
It is used for multiplexing and demultiplexing.

Data: This field carries data encapsulated from the upper-layer protocols.
It is a minimum of 46 and a maximum of 1500 bytes. We discuss the reason for these minimum and maximum values shortly.
If the data coming from the upper layer is more than 1500 bytes, it should be fragmented and encapsulated in more than one frame.
If it is less than 46 bytes, it needs to be padded with extra 0s.
A padded data frame is delivered to the upper-layer protocol as it is (without removing the padding), which means that it is the responsibility of the upper layer to remove or, in the case of the sender, to add the padding.
The upper-layer protocol needs to know the length of its data. For example, a datagram has a field that defines the length of the data.

CRC: The last field contains error detection information, in this case a CRC-32.
The CRC is calculated over the addresses, types, and data field.
If the receiver calculates the CRC and finds that it is not zero (corruption in transmission), it discards the frame.

Ethernet has imposed restrictions on both the minimum and maximum lengths of a frame.
The minimum length restriction is required for the correct operation of CSMA/CD, as we will see shortly.
An Ethernet frame needs to have a minimum length of 512 bits or 64 bytes.
Part of this length is the header and the trailer.
If we count 18 bytes of header and trailer (6 bytes of source address, 6 bytes of destination address, 2 bytes of length or type, and 4 bytes of CRC), then the minimum length of data from the upper layer is 64 − 18 = 46 bytes.
If the upper-layer packet is less than 46 bytes, padding is added to make up the difference.

The standard defines the maximum length of a frame (without preamble and SFD field) as 1518 bytes.
If we subtract the 18 bytes of header and trailer, the maximum length of the payload is 1500 bytes.
The maximum length restriction has two historical reasons.
First, memory was very expensive when Ethernet was designed; a maximum length restriction helped to reduce the size of the buffer.
Second, the maximum length restriction prevents one station from monopolizing the shared medium, blocking other stations that have data to send.

Q6) b) Solution :

There are several characteristics of wireless LANs that either do not apply to wired LANs or the existence of which is negligible and can be ignored.
We discuss some of these characteristics here to pave the way for discussing wireless LAN protocols.

Attenuation

The strength of electromagnetic signals decreases rapidly because the signal disperses in all directions; only a small portion of it reaches the receiver.
The situation becomes worse with mobile senders that operate on batteries and normally have small power supplies.

Interference

Another issue is that a receiver may receive signals not only from the intended sender, but also from other senders if they are using the same frequency band.

Multipath Propagation

A receiver may receive more than one signal from the same sender because electromagnetic waves can be reflected back from obstacles such as walls, the ground, or objects.
The result is that the receiver receives some signals at different phases (because they travel different paths).
This makes the signal less recognizable.

Error

With the above characteristics of a wireless network, we can expect that errors and error detection are more serious issues in a wireless network than in a wired network.
If we think about the error level as the measurement of signal-to-noise ratio (SNR), we can better understand why error detection and error correction and retransmission are more important in a wireless network.
We discussed SNR in more detail in Chapter 3, but it is enough to say that it measures the ratio of good stuff to bad stuff (signal to noise).
If SNR is high, it means that the signal is stronger than the noise (unwanted signal), so we may be able to convert the signal to actual data.
On the other hand, when SNR is low, it means that the signal is corrupted by the noise and the data cannot be recovered.

Q7) a) Solution :

The frame transmission time is 200/200 kbps or 1 ms.

a. If the system creates 1000 frames per second, or 1 frame per millisecond, then G = 1. In this case S = G × e−2G = 0.135 (13.5 percent). This means that the throughput is 1000 × 0.135 = 135 frames. Only 135 frames out of 1000 will probably survive.

b. If the system creates 500 frames per second, or 1/2 frames per millisecond, then G = 1/2. In this case S = G × e−2G = 0.184 (18.4 percent). This means that the throughput is 500 × 0.184 = 92 and that only 92 frames out of 500 will probably survive. Note that this is the maximum throughput case, percentage-wise.

c. If the system creates 250 frames per second, or 1/4 frames per millisecond, then G = 1/4. In this case S = G × e−2G = 0.152 (15.2 percent). This means that the throughput is 250 × 0.152 = 38. Only 38 frames out of 250 will probably survive

Q7)b) Solution:

205.16.37.39 is 11001101 00010000 00100101 00100111

No. of addresses is 2^(32-28) = 2^4 =16

As suffix is 4 (i.e. 32-28 where prefix is 28), last 4 binary digit LSBs should be made all 0s for 1st address and all 1s for last address.
Hence ,
1st address is 205.16.37.32/28

And last address is 205.16.37.47/28

**********END**********