Module 4

DATA LINK LAYER

CONTENTS

IV	Link layer and Physical Layer: Introduction to link layer – Error detection (parity, checksum, and CRC), Multiple access protocols (collision and token based), IEEE 802.3 Ethernet, Switching and bridging, Media, Signal strength and interference. Data encoding. Ethernet switches , Routers MAC, ARP, FIB

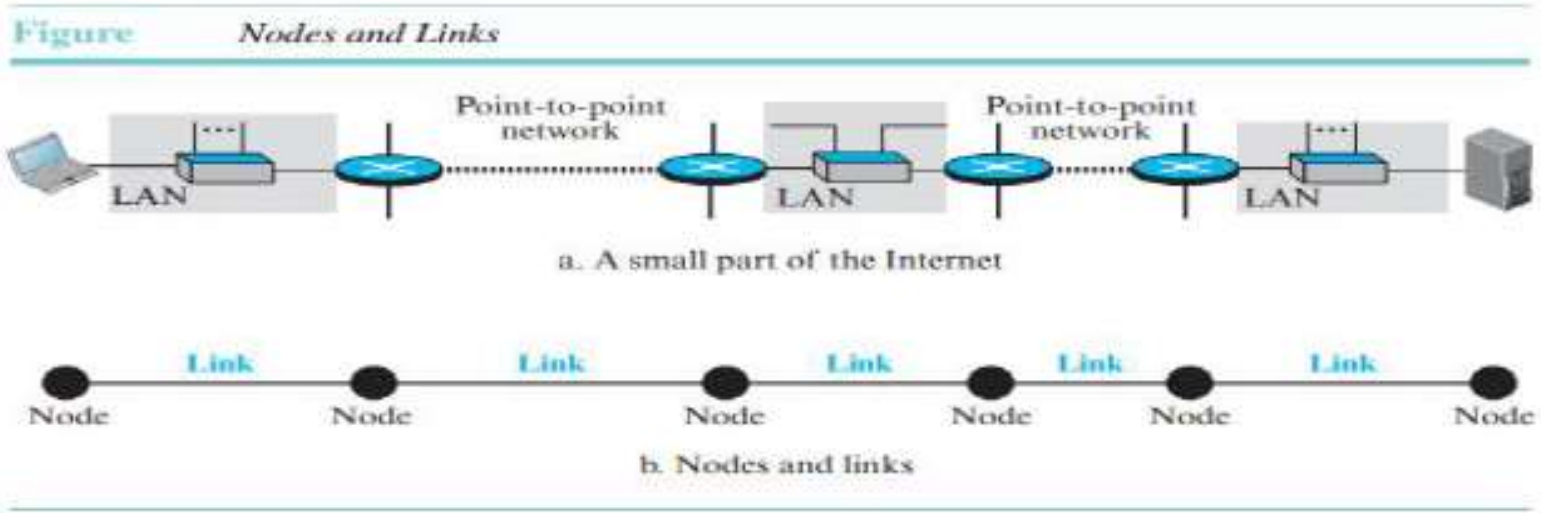
CONTENTS

- Introduction
- Framing
- Error detection methods-Parity check,CRC,Checksum

Data link layer

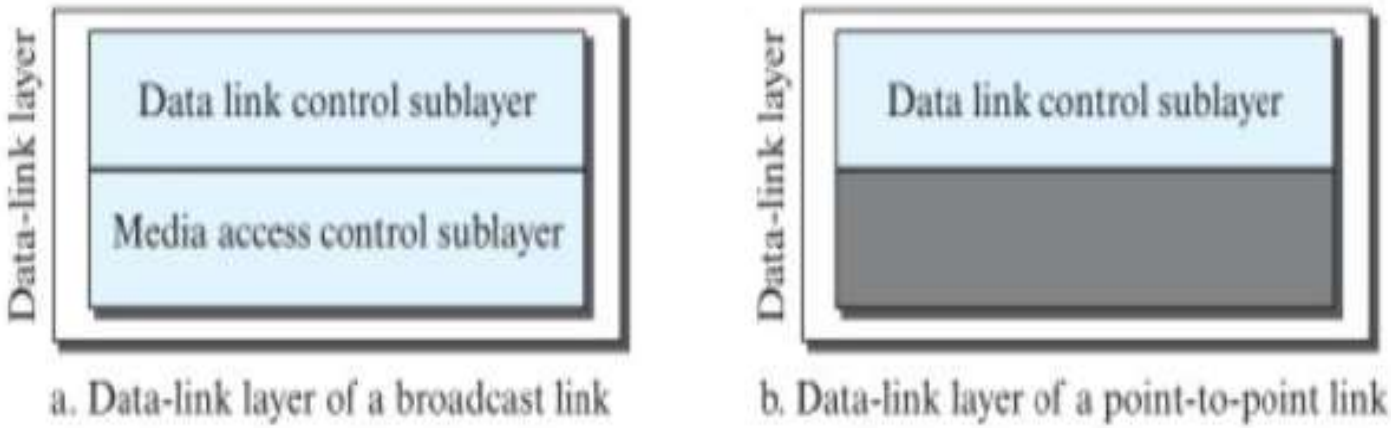
- Enables node to node / hop-to-hop communication

[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]



- Two Types of Links: point to point, broadcast links
- 2 sublayers: Logical link control(LLC), Media Access Control(MAC)

Figure 5.3 *Dividing the data-link layer into two sublayers*



LLC: it controls the framing, flow control, error control and error detection and correction functions of the data link layer.

MAC: It controls the transmission of data packets via shared channels.

[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

Functions

- Deals with procedures for communication between two adjacent nodes—node-to-node communication
- Data link control (DLC) functions include
  - Framing
  - Flow and error control
  - Error detection and correction

FRAMING

- Data transmission in the physical layer means moving bits in the form of a signal from the source to the destination.

- The data-link layer, on the other hand, needs to pack bits into frames, so that each frame is distinguishable from another

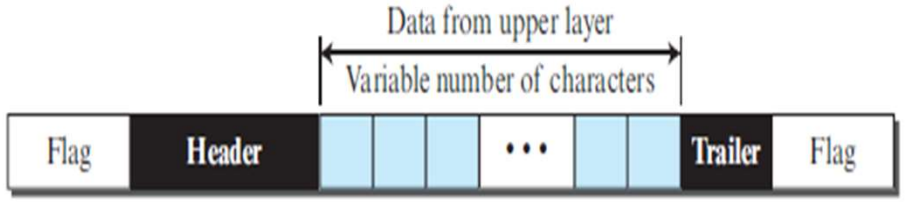
- Framing in the data-link layer separates a message from one source to a destination by adding a sender address and a destination address.

FRAMING

- Data-link layer takes the packets from the Network Layer and encapsulates them into frames by adding header and trailer.
- Frame Size:
  - fixed-size framing: the size of the frame is fixed.
  - variable-size framing: the size of each frame varies. So a mechanism is needed to define the end of one frame and the beginning of the next. Eg: Local area networks. Two approaches are used: a character-oriented approach and a bit-oriented approach.

Character-Oriented Framing

Figure 5.4 *A frame in a character-oriented protocol*

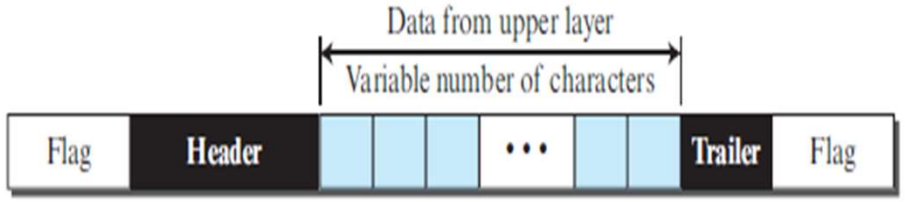


[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

- In character-oriented (or byte-oriented) framing, data to be carried are 8-bit characters
- The header which carries the source and destination addresses and other control information, and the trailer, which carries error detection redundant bits, are also multiples of 8 bits.
- To separate one frame from the next, an 8-bit (1-byte) flag is added at the beginning and the end of a frame.

Character-Oriented Framing

Figure 5.4 *A frame in a character-oriented protocol*



[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

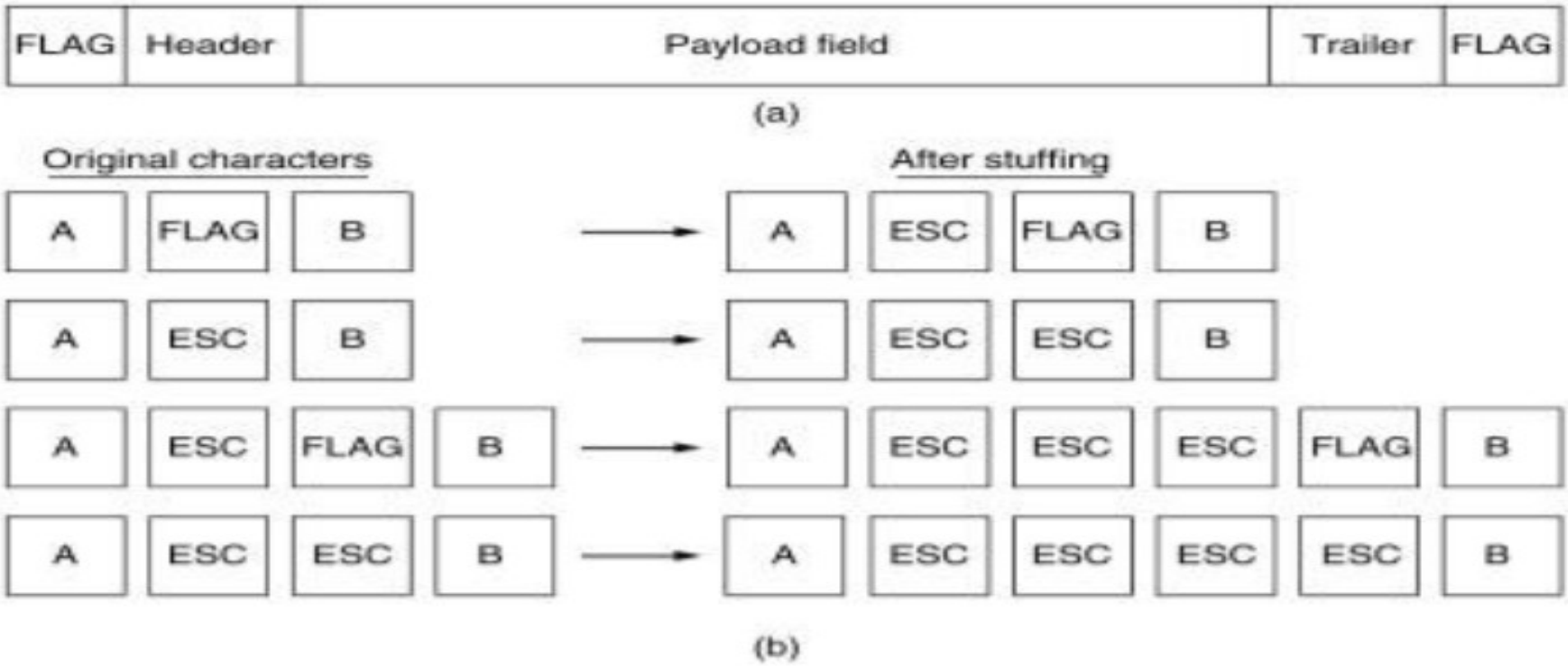
- The flag, composed of protocol-dependent special characters, signals the start or end of a frame.
- Character-oriented framing was used when only text was exchanged by the data-link layers.

Byte stuffing

- The flag could be selected to be any character not used for text communication.
- We also send other types of information such as graphs, audio, and video, any pattern used for the flag could also be part of the information.
- *Byte stuffing is the process of adding one extra byte whenever there is a flag in the text*

Byte stuffing

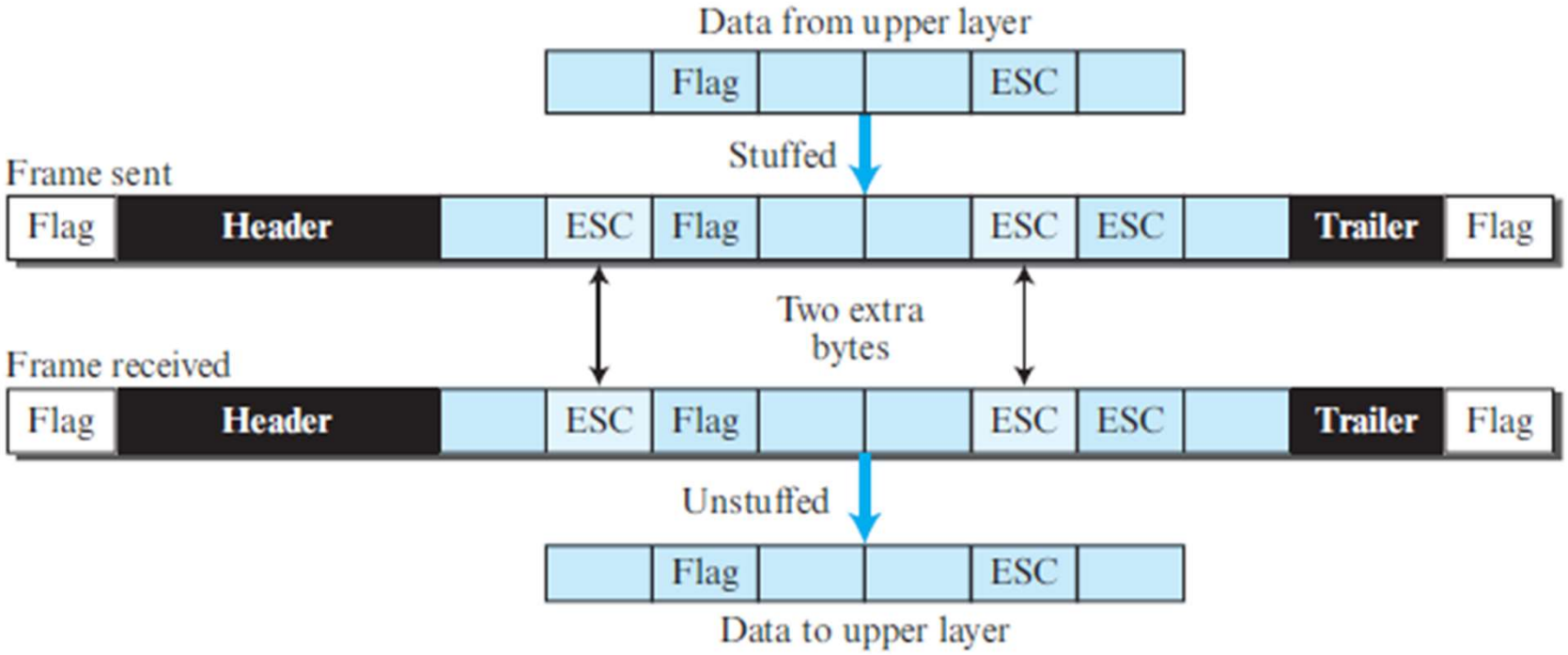
- In byte stuffing (or character stuffing), a special byte is added to the data section of the frame when there is a character with the same pattern as the flag.
- The data section is stuffed with an extra byte.
- This byte is usually called the escape character (ESC) and has a predefined bit pattern.
- Whenever the receiver encounters the ESC character, it removes it from the data section and treats the next character as data, not as a delimiting flag.



Framing with Byte stuffing

[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

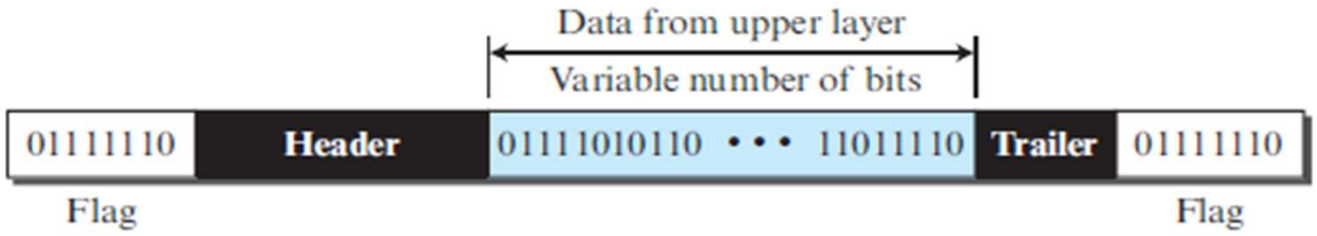
Figure 5.5 *Byte stuffing and unstuffing*



[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

Bit-Oriented Framing

Figure 5.6 *A frame in a bit-oriented protocol*

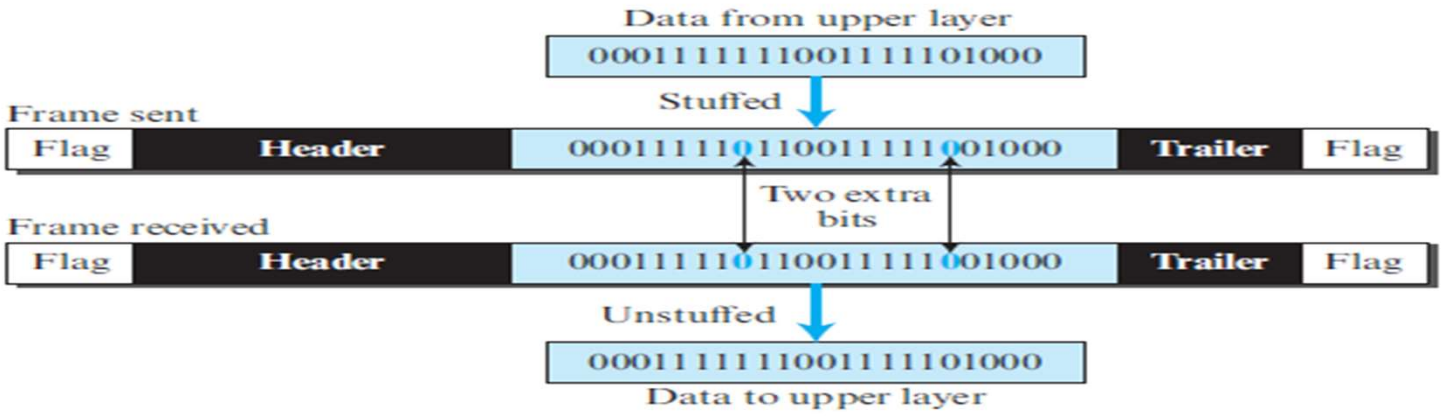


[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

- In addition to headers (and possible trailers), we still need a delimiter to separate one frame from the other.
- Most protocols use a special 8-bit pattern flag, 01111110, as the delimiter to define the beginning and the end of the frame

- This flag can create the same type of problem as we saw in the character-oriented protocols. That is, if the flag pattern appears in the data, we need to somehow inform the receiver that this is not the end of the frame. We do this by stuffing one single bit (instead of 1 byte) to prevent the pattern from looking like a flag. The strategy is called bit stuffing.
- Bit stuffing is the process of adding one extra 0 whenever five consecutive 1s follow a 0 in the data, so that the receiver does not mistake the pattern 01111110 for a flag.

Figure 5.7 *Bit stuffing and unstuffing*



[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

Error Detection and Correction

Error Detection and Correction

- At the data-link layer, if a frame is corrupted between the two nodes, it needs to be corrected before it continues its journey to other nodes.

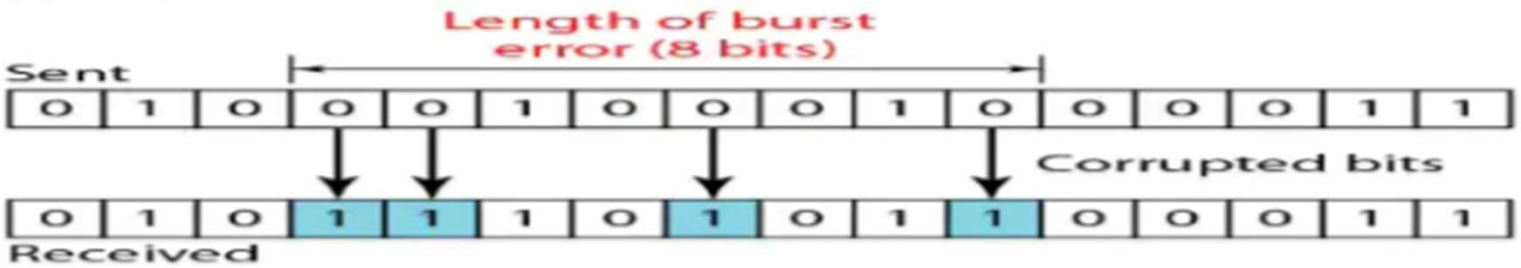
Types of Errors:

- single-bit error: only 1 bit of a given data unit is changed from 1 to 0 or from 0 to 1.
- burst error: 2 or more bits in the data unit have changed from 1 to 0 or from 0 to 1.
- The number of bits affected depends on the data rate and duration of noise.

Types of Error Single Bit Errors

In a single bit error, only 1 bit in the data unit changes:

Burst Errors



[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

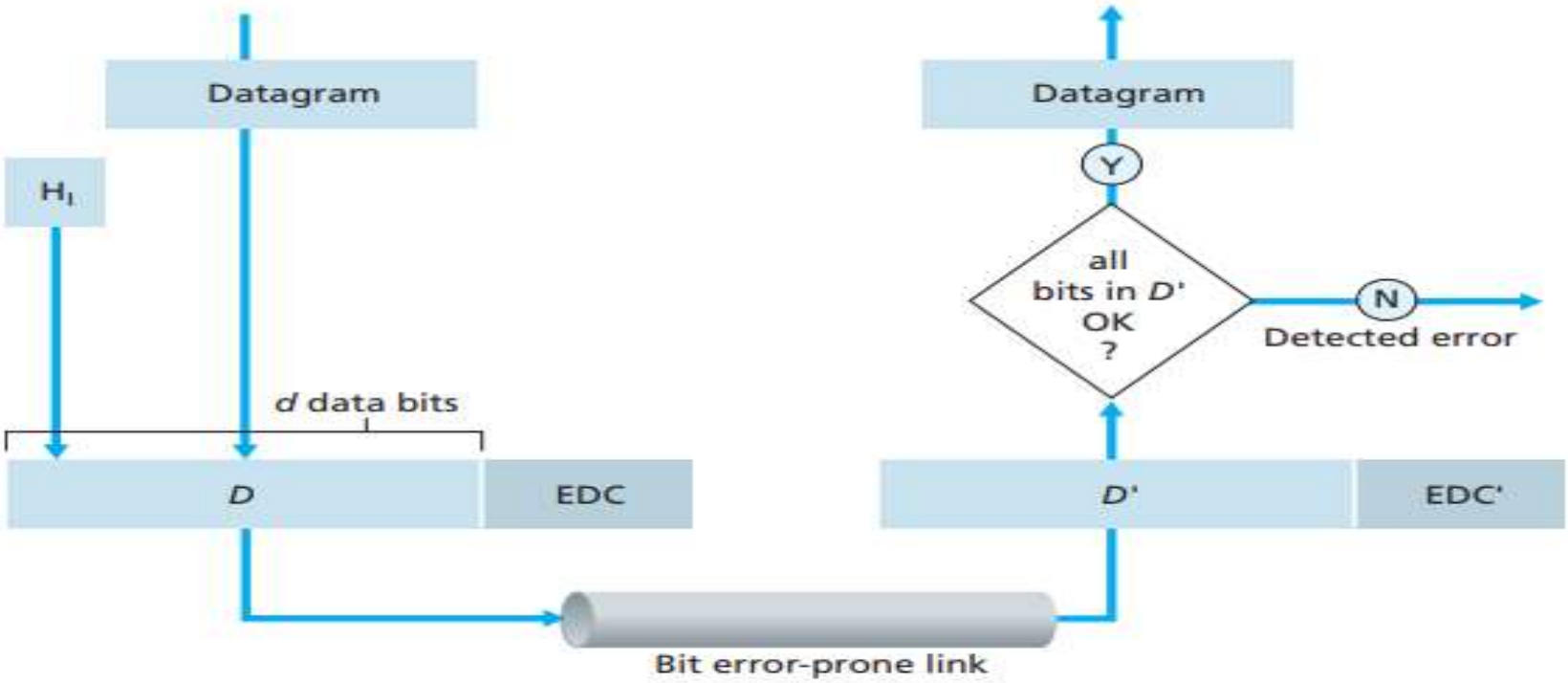
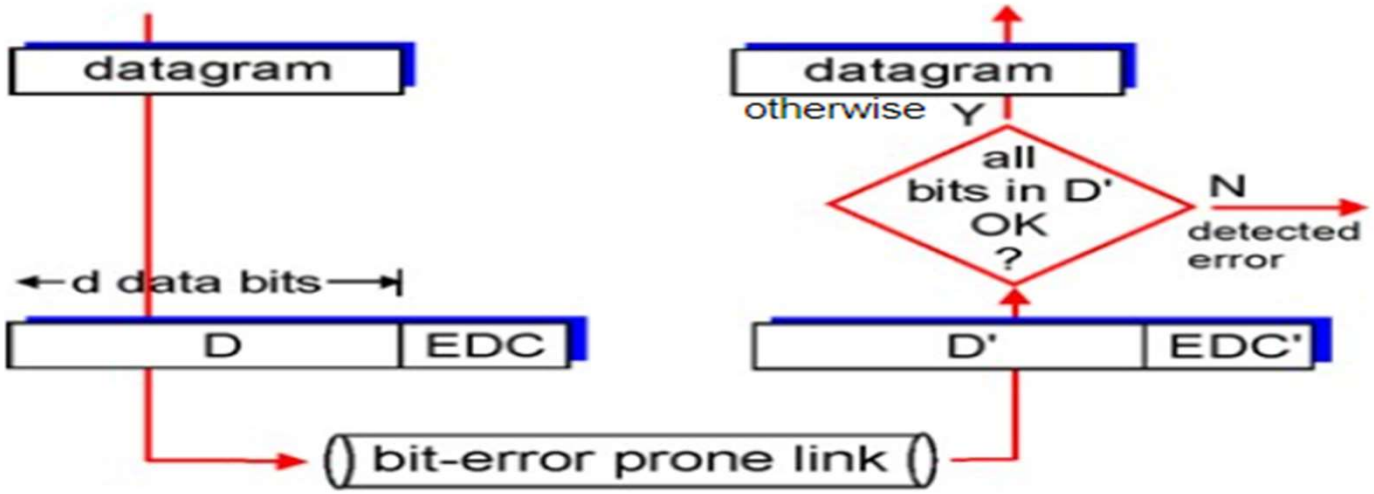


Figure 5.3 ◆ Error-detection and -correction scenario

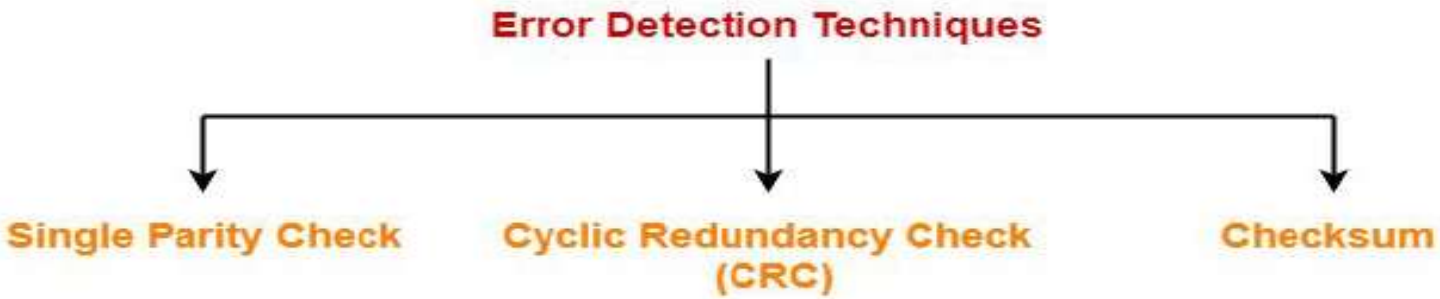
James F Kurose and Keith W Ross, "Computer Networking: A Top - Down Approach", Pearson Education; 6 th Edition (2017)

Error detection

- EDC:
  - Error Detection and Correction bits
- D:
  - Data protected by error checking, may include header fields
- Error detection not 100% reliable!
  - Protocol may miss some errors, but rarely
  - Larger EDC field yields better detection and correction
    - Also reduces link throughput



Error Detection techniques:-



Calculating bit string parity

- A bit string has **odd parity** if the number of 1s in the string is odd.
  - 100011, 1, 000010 have odd parity
- A bit string has **even parity** if the number if 1s in the string is even.
  - 01100, 000, 11001001 have even parity
- Assume 0 is an even number

i) Parity checks:

- the simplest form of error detection
- uses a single parity bit.
- Let the information to be sent, D has d bits.
- Even parity:
  - the sender simply includes one additional bit and chooses its value such that the total number of 1s in the d + 1 bits (the original information plus a parity bit) is even.

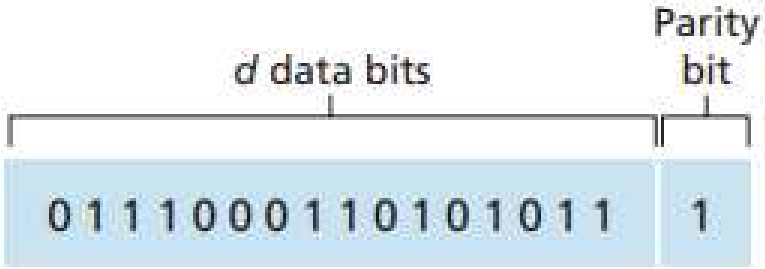


Figure 5.4 ♦ One-bit even parity

James F Kurose and Keith W Ross, "Computer Networking: A Top - Down Approach", Pearson Education; 6 th Edition (2017)

- odd parity:
  - the parity bit value is chosen such that there is an odd number of 1s.
- Example:

  i) if the original data is 1010001,

  For even parity; parity bit is 1 and the code word is 10100011

  For odd parity; parity bit is 0 and the code word is 10100010

  ii) if the original data is 1101001,

  For even parity; parity bit is 0 and the code word is 11010010

  For odd parity; parity bit is 1 and the code word is 11010011

- This technique is guaranteed to detect an odd number of bit errors (one, three, five and so on).
- But this technique can not detect an even number of bit errors (two, four, six and so on).

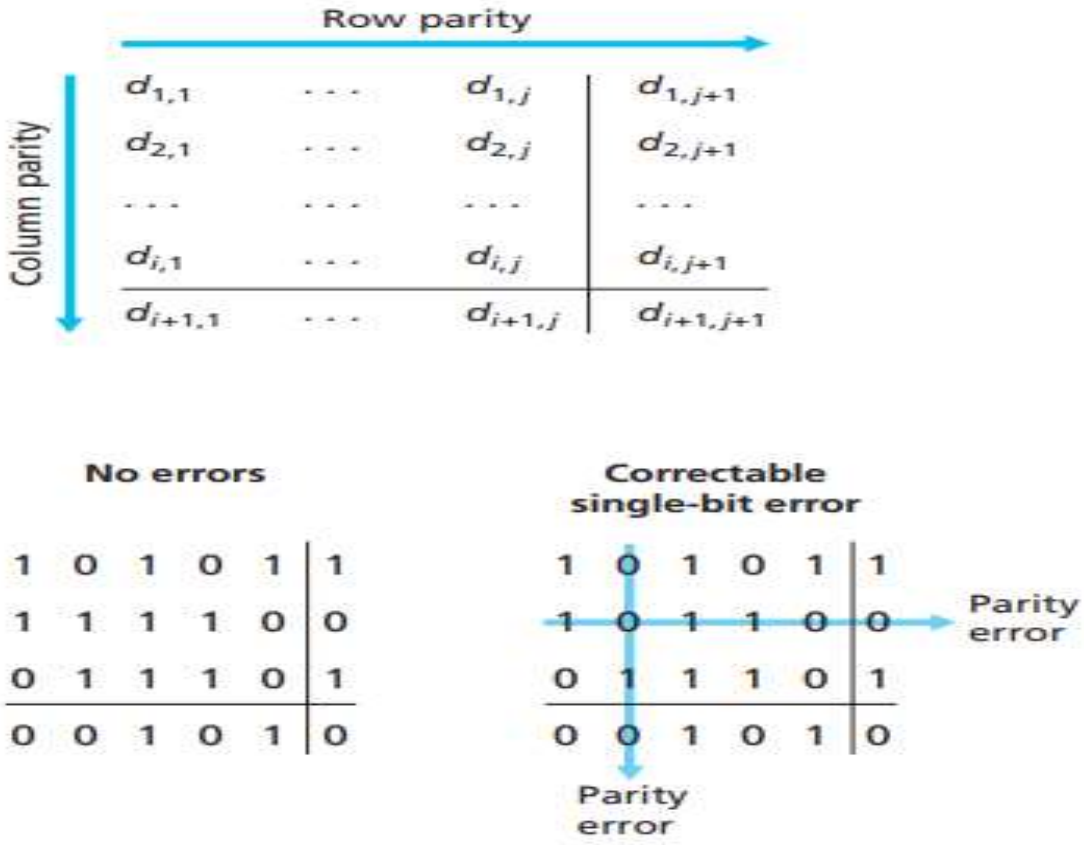
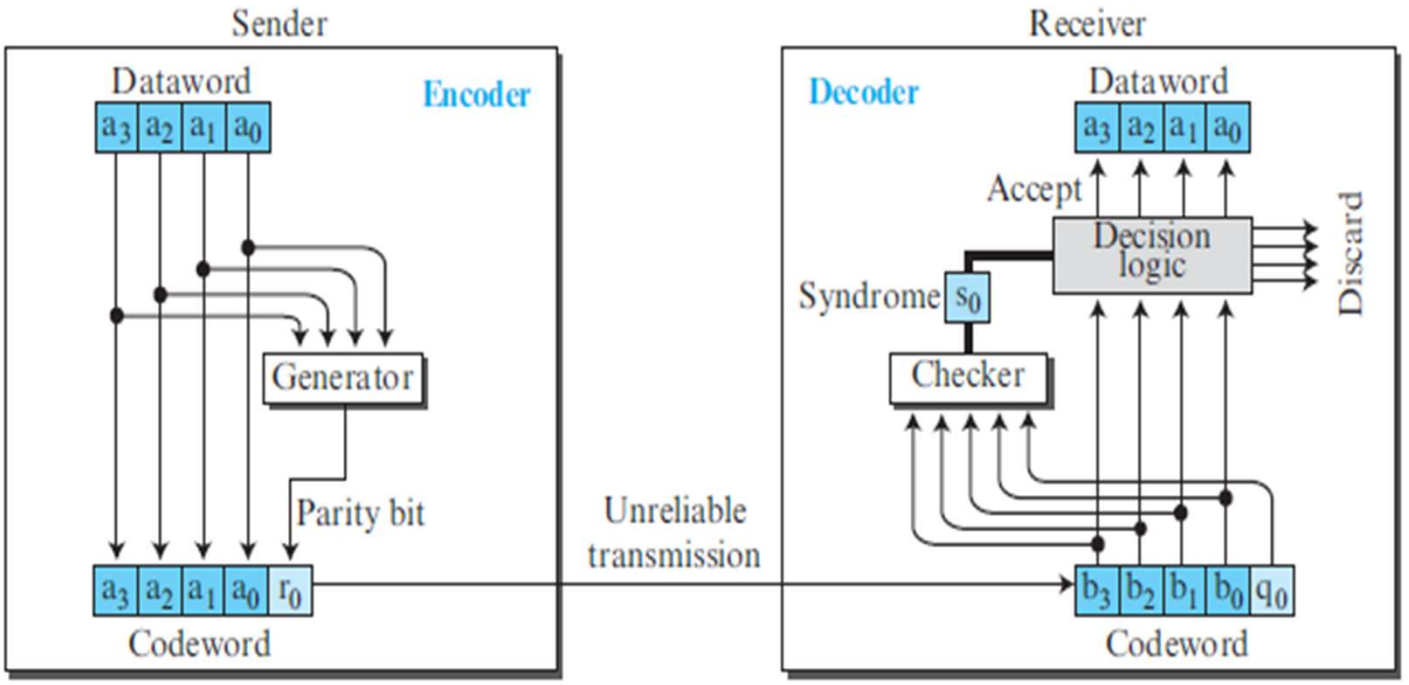


Figure 5.5 ◆ Two-dimensional even parity

James F Kurose and Keith W Ross, "Computer Networking: A Top - Down Approach", Pearson Education; 6 th Edition (2017)

The ability of the receiver to both detect and correct errors is known as forward error correction (FEC).

Figure 5.11 *Encoder and decoder for simple parity-check code*



- A parity-check code can detect an odd number of errors

[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

Parity check code

- Assume we are transmitting blocks of *k* bits.
  - A block (w) of length (k) is encoded as (wa), where the value of the **parity bit** (a) is chosen so that (wa) has even parity.

- Example:
  - If w = 10110, we send wa = 101101, which has even parity

  - With no bit flips in the transmission, the receiver gets the bit string exactly as it was sent by the sender. Bit string has even parity.
  - If there are an odd number of bit flips in the transmission, the receiver gets a bit string with odd parity. Retransmission is requested.
  - If there are an even number of bit flips in the transmission, the receiver gets a bit string with even parity. The error(s) go undetected.

  - Another solution?

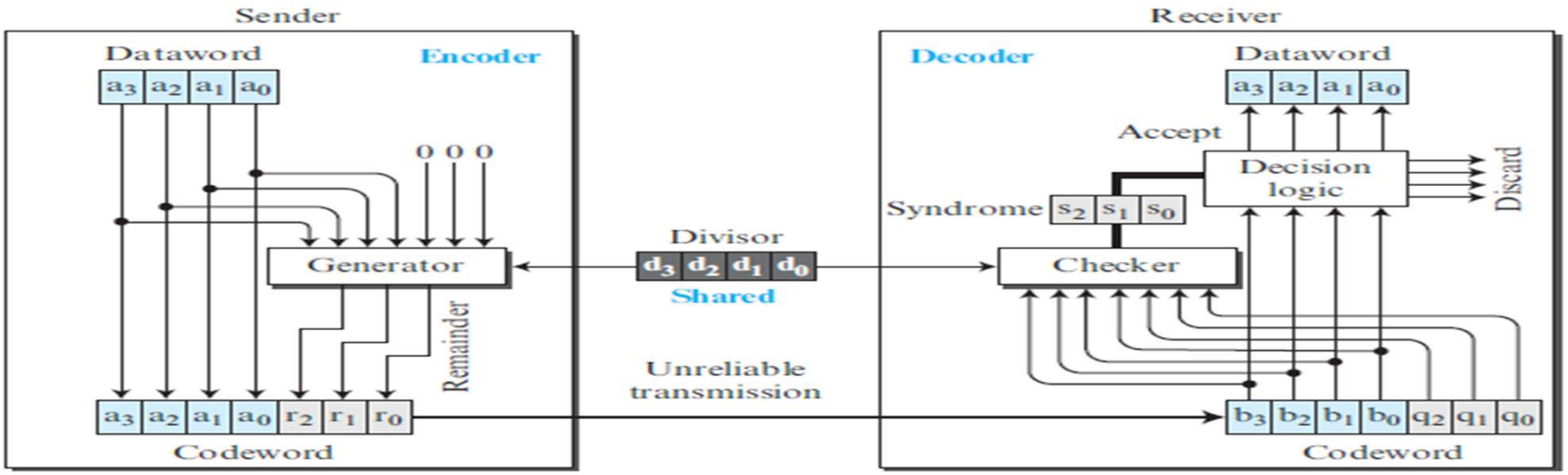
Cyclic Codes

- In a cyclic code, if a codeword is cyclically shifted (rotated), the result is another codeword.
- For example, if 1011000 is a codeword and we cyclically left-shift, then 0110001 is also a codeword.
- In this case, if we call the bits in the first word a0 to a6, and the bits in the second word b0 to b6, we can shift the bits by using the following:
- b1 = a0 b2 = a1 b3 = a2 b4 = a3 b5 = a4 b6 = a5 b0 = a6

Cyclic Redundancy Check

- We can create cyclic codes to correct errors.
- A type of cyclic codes called the cyclic redundancy check (CRC) that is used in networks such as LANs and WANs.

Figure 5.12 *CRC encoder and decoder*



[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

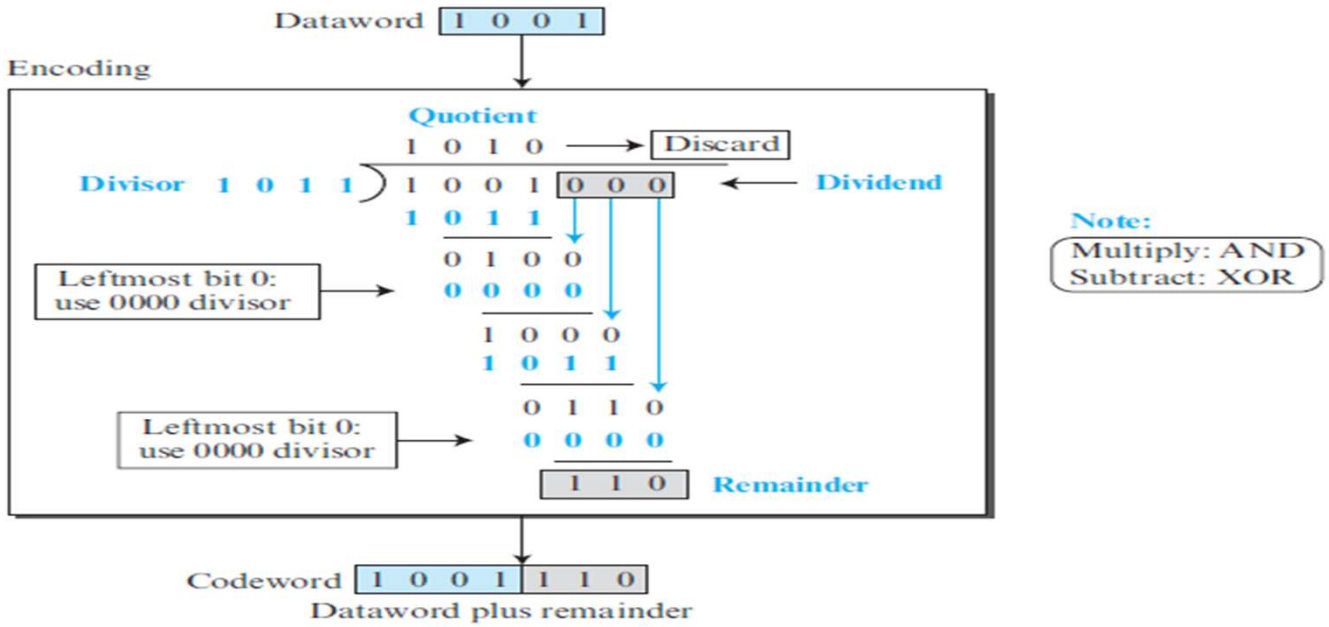
Steps

- Dataword has k bits (4 here);
- The codeword has n bits (7 here).
- The size of the dataword is augmented by adding n – k (3 here) 0s to the right-hand side of the word.
- The n-bit result is fed into the encoder/generator.
- The generator uses a divisor of size n – k + 1 (4 here), predefined and agreed upon.
- The generator divides the augmented dataword by the divisor (modulo-2 division). The quotient of the division is discarded;
- The remainder ($r_2r_1r_0$) is appended to the dataword to create the codeword.
- The decoder receives the codeword (possibly corrupted in transition).
- A copy of all n bits is fed to the checker, which is a replica of the generator.
- The remainder produced by the checker is a syndrome of n – k (3 here) bits. If the syndrome bits are all 0s, the 4 leftmost bits of the codeword are accepted as the dataword (interpreted as no error);
- Otherwise, the 4 bits are discarded (error).

Sender

Encoder Let us take a closer look at the encoder. The encoder takes a dataword and augments it with $n - k$ number of 0s. It then divides the augmented dataword by the divisor, as shown in Figure 5.13.

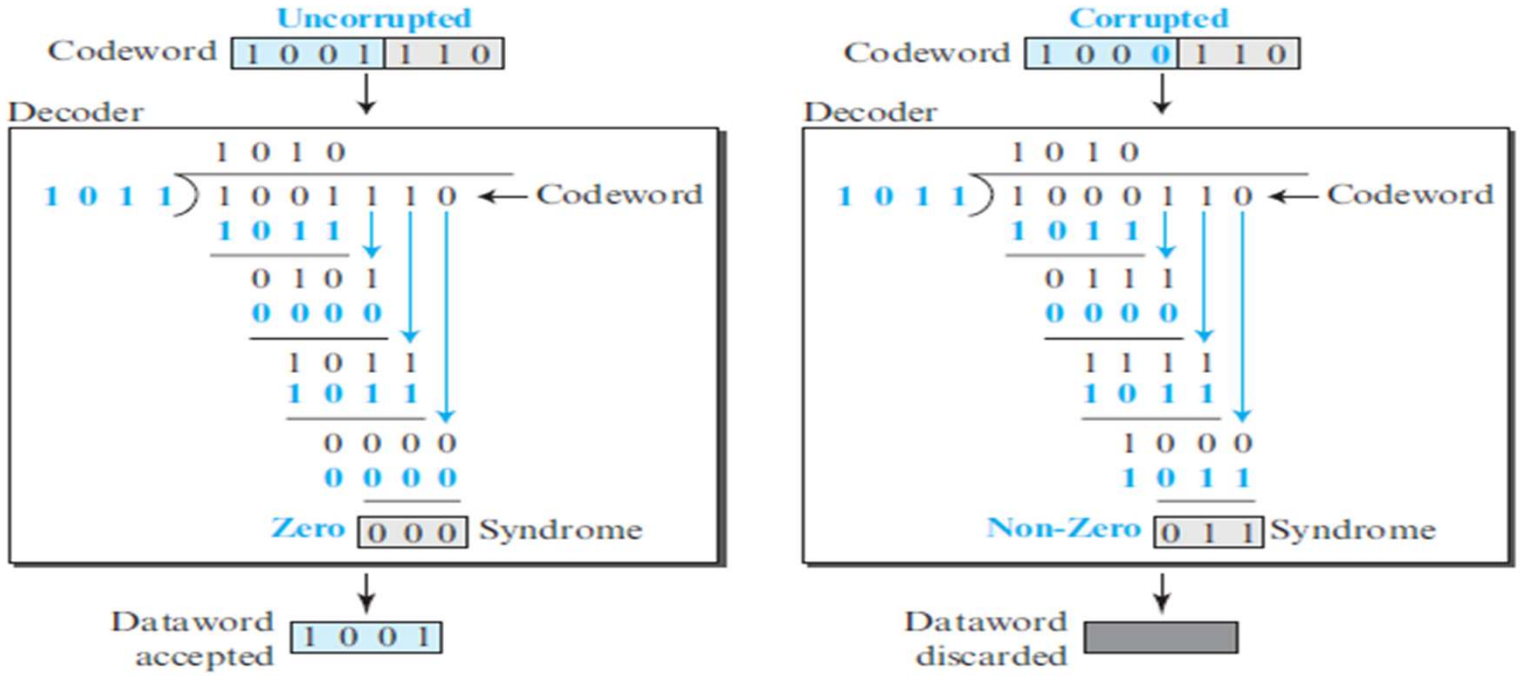
Figure 5.13 *Division in CRC encoder*



[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

Receiver

Figure 5.14 *Division in the CRC decoder for two cases*



[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

Checksum

- Checksum is an error-detecting technique that can be applied to a message of any length.
- In the Internet, the checksum technique is mostly used at the network and transport layer rather than the data-link layer

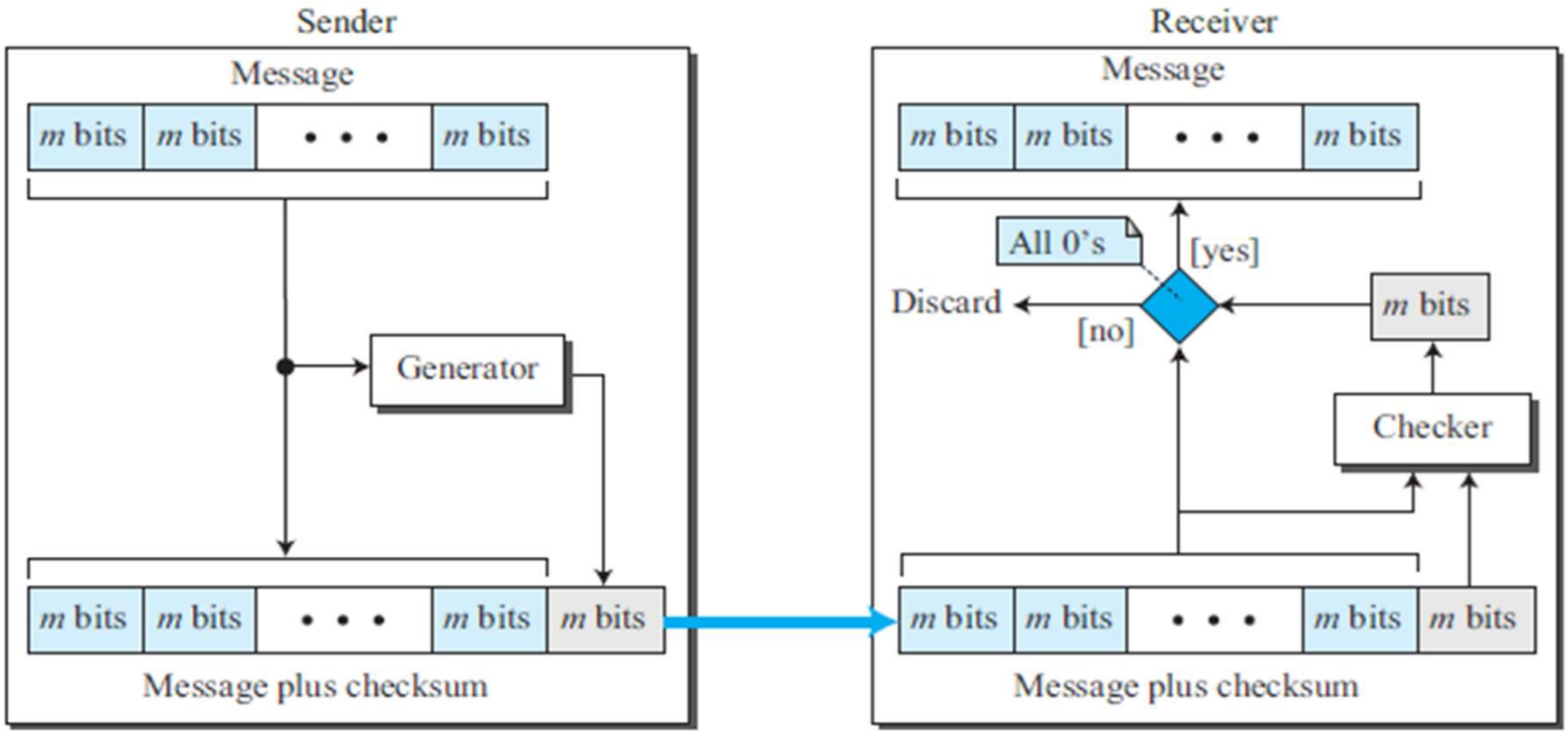
Traditional checksum

- Suppose the message is a list of five 4-bit numbers that we want to send to a destination.
- In addition to sending these numbers, we send the sum of the numbers.
- For example, if the set of numbers is (7, 11, 12, 0, 6), we send (7, 11, 12, 0, 6, 36), where 36 is the sum of the original numbers.
- The receiver adds the five numbers and compares the result with the sum.
- If the two are the same, the receiver assumes no error, accepts the five numbers, and discards the sum.
- Otherwise, there is an error somewhere and the message not accepted

Checksum

- At the source, the message is first divided into m-bit units.
- The generator then creates an extra m-bit unit called the checksum, which is sent with the message.
- At the destination, the checker creates a new checksum from the combination of the message and sent checksum.
- If the new checksum is all 0s, the message is accepted; otherwise, the message is discarded .
- Note that in the real implementation, the checksum unit is not necessarily added at the end of the message; it can be inserted in the middle of the message.

Figure 5.15 *Checksum*



[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

Checksum

Checksum = Check + sum.

Sender side – Checksum Creation.

Receiver side – Checksum Validation.

Checksum:

- Sender:
  - the message is first divided into m-bit units
  - All the m bit units are then added.
  - Then 1's complement of the sum is taken which is the checksum value.
  - The data along with checksum is then sent to the receiver.
- Receiver:
  - the received message is first divided into m-bit units.
  - All the m bit units are then added along with the checksum value.
  - Then take the 1's complement of the result.
  - If the result is zero, then receiver assumes that no error has occurred and accepts the data; otherwise, the receiver assumes that error has occurred and discards the data.

- Consider the data to be transmitted: 1001100111100010001001001000100
- Consider 8 bit checksum is used.
- Sender:
    - Divide into 8 bit units:

      10011001 11100010 00100100 10000100

    - Take the sum of these units (1000100011). The sum is 10 bits so extra 2 bits are wrapped around. Thus we get 00100101 (00100011 + 10)
    - Take 1's complement: 11011010 which is the checksum value.

- Receiver:

    - Take the sum of all units and checksum; Sum of all units + Checksum value = 00100101 + 11011010 = 11111111
    - Take 1's complement; we get all 0's. Thus the receiver assumes that no error has occurred and accepts the data.

Checksum

- The Internet checksum is based on this approach—bytes of data are treated as 16-bit integers and summed.
- The 1s complement of this sum then forms the Internet checksum that is carried in the segment header.
- The receiver checks the checksum by taking the 1s complement of the sum of the received data (including the checksum) and checking whether the result is all 1 bits.
- If any of the bits are 0, an error is indicated

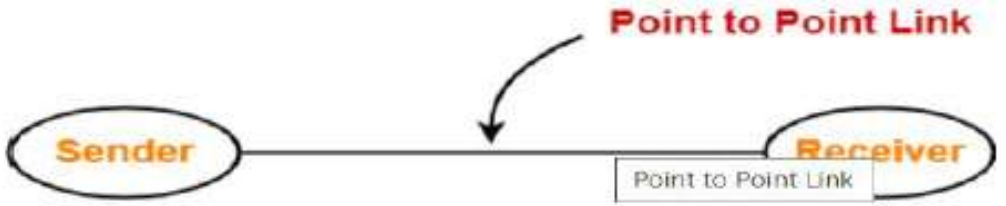
MUTLIPLE ACCESS LINKS AND PROTOCOLS

Types of network links

- Communication links enable the devices to communicate with each other.
- The types of links are:

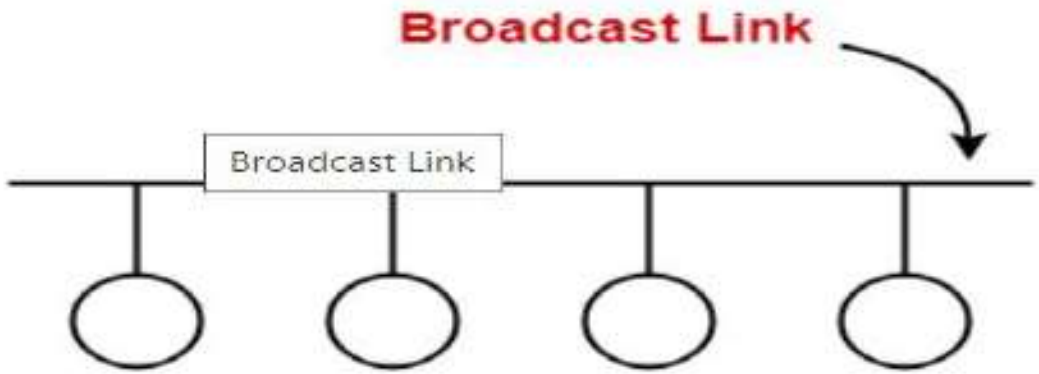
i) Point to point links:

- is a dedicated link that exists between the two stations.
- the entire capacity of the link is available only for these 2 stations.
- The link-layer protocols for point-to-point links: the point-to-point protocol (PPP) and high-level data link control (HDLC)



ii) Broadcast links:

- is a common link to which multiple stations are connected.
- The capacity of the link is shared among these stations.
- Eg: Ethernet and wireless LANs

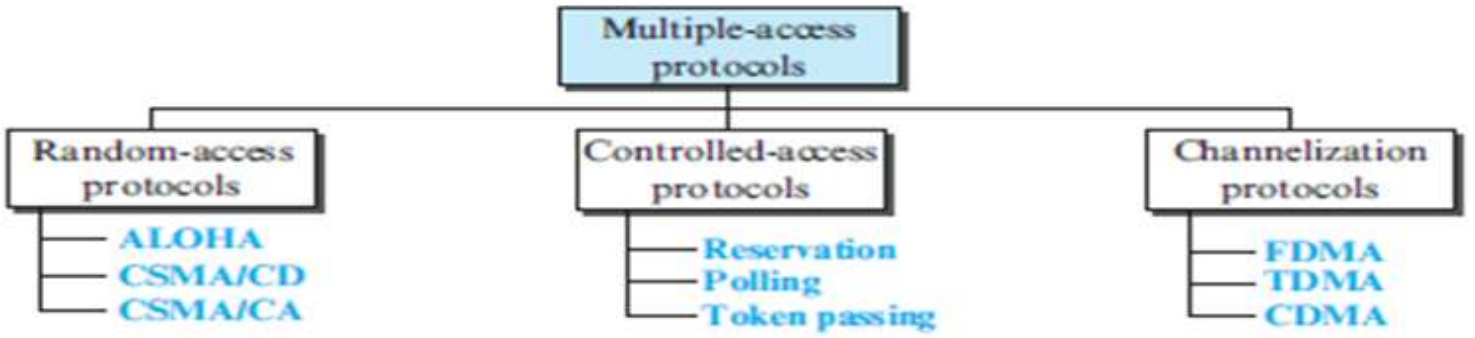


(Bus Topology)

Multiple access protocols (collision and token based)

- Data-link layer is divided into two sublayers:
  - Data link control (DLC)
  - Media access control (MAC).
- When nodes or stations are connected and use a common link, called a multipoint or broadcast link, we need a multiple-access protocol to coordinate access to the link

Figure 5.28 *Taxonomy of multiple-access protocols*



[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

Random Access Protocols

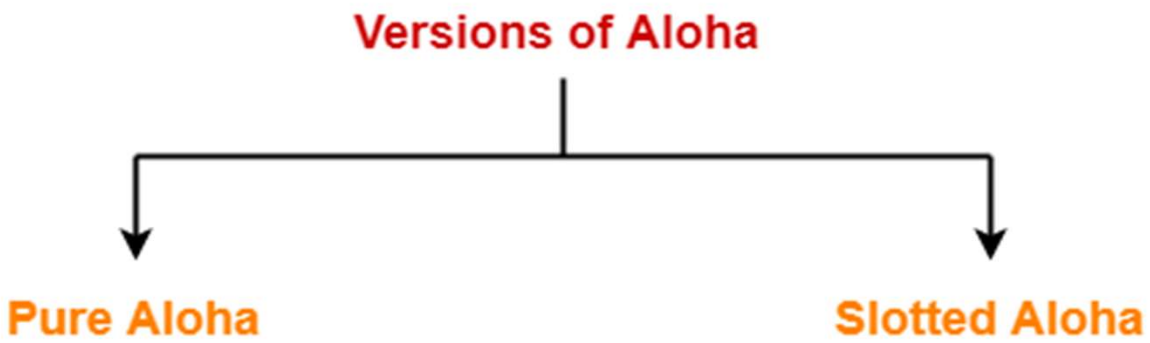
- When node has packet to send
  - transmit at full channel data rate R.
  - no *a priori* coordination among nodes
- two or more transmitting nodes → "collision",
- random access MAC protocol specifies:
  - how to detect collisions
  - how to recover from collisions (e.g., via delayed retransmissions)
- Examples of random access MAC protocols:
  - slotted ALOHA
  - ALOHA
  - CSMA, CSMA/CD, CSMA/CA

- So each station follows a procedure that answers the following questions:
  - When can the station access the medium?
  - What can the station do if the medium is busy?
  - How can the station determine the success or failure of the transmission?
  - What can the station do if there is an access conflict?

- The random-access methods are
  - ALOHA: used a very simple procedure called multiple access (MA)
  - Carrier sense multiple access (CSMA): each station has to sense the medium before transmitting
  - Carrier sense multiple access with collision detection (CSMA/CD): it tells the station what to do when a collision is detected
  - Carrier sense multiple access with collision avoidance (CSMA/CA): it tries to avoid the collision.

Aloha-

There are two different versions of Aloha-

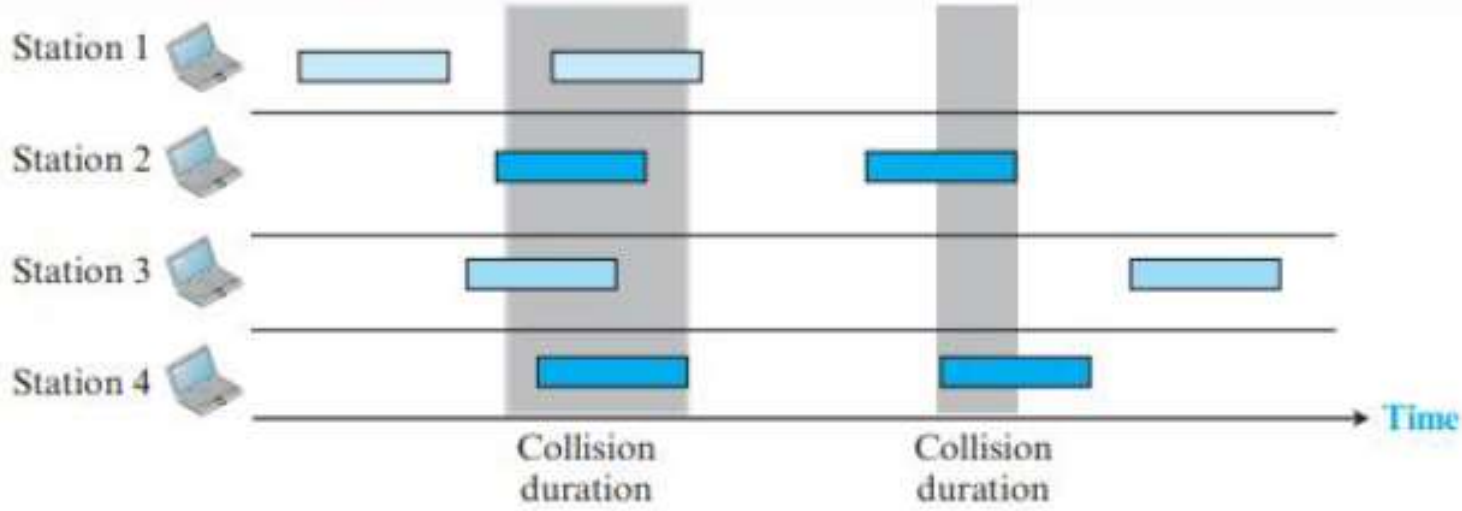


1. Pure Aloha
2. Slotted Aloha

ALOHA

- The original ALOHA protocol is called pure ALOHA.
- each station sends a frame whenever it has a frame to send (multiple access).
- But there is a chance of collision between frames from different stations.
-

Figure *Frames in a pure ALOHA network*



- The pure ALOHA protocol relies on acknowledgments from the receiver.
- If the acknowledgment does not arrive within a particular period of time, the station resends the frame.
- A collision involves two or more stations.
- Pure ALOHA dictates that when the time-out period passes, each station waits a random amount of time before resending its frame. The randomness will help avoid more collisions.
- That time is called the back-off time T_B.
- Pure ALOHA has a second method to prevent congesting the channel with retransmitted frames.
- After a maximum number of retransmission attempts Kmax, a station must give up and try later.

1. Pure Aloha-

- It allows the stations to transmit data at any time whenever they want.
- After transmitting the data packet, station waits for some time.

Then, following 2 cases are possible-

Case-01:

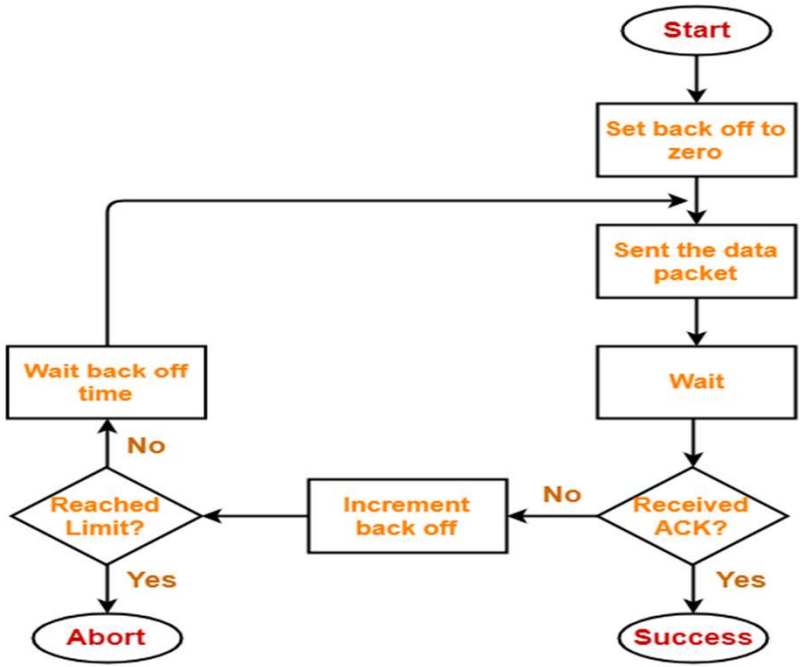
- Transmitting station receives an acknowledgement from the receiving station.
- In this case, transmitting station assumes that the transmission is successful.

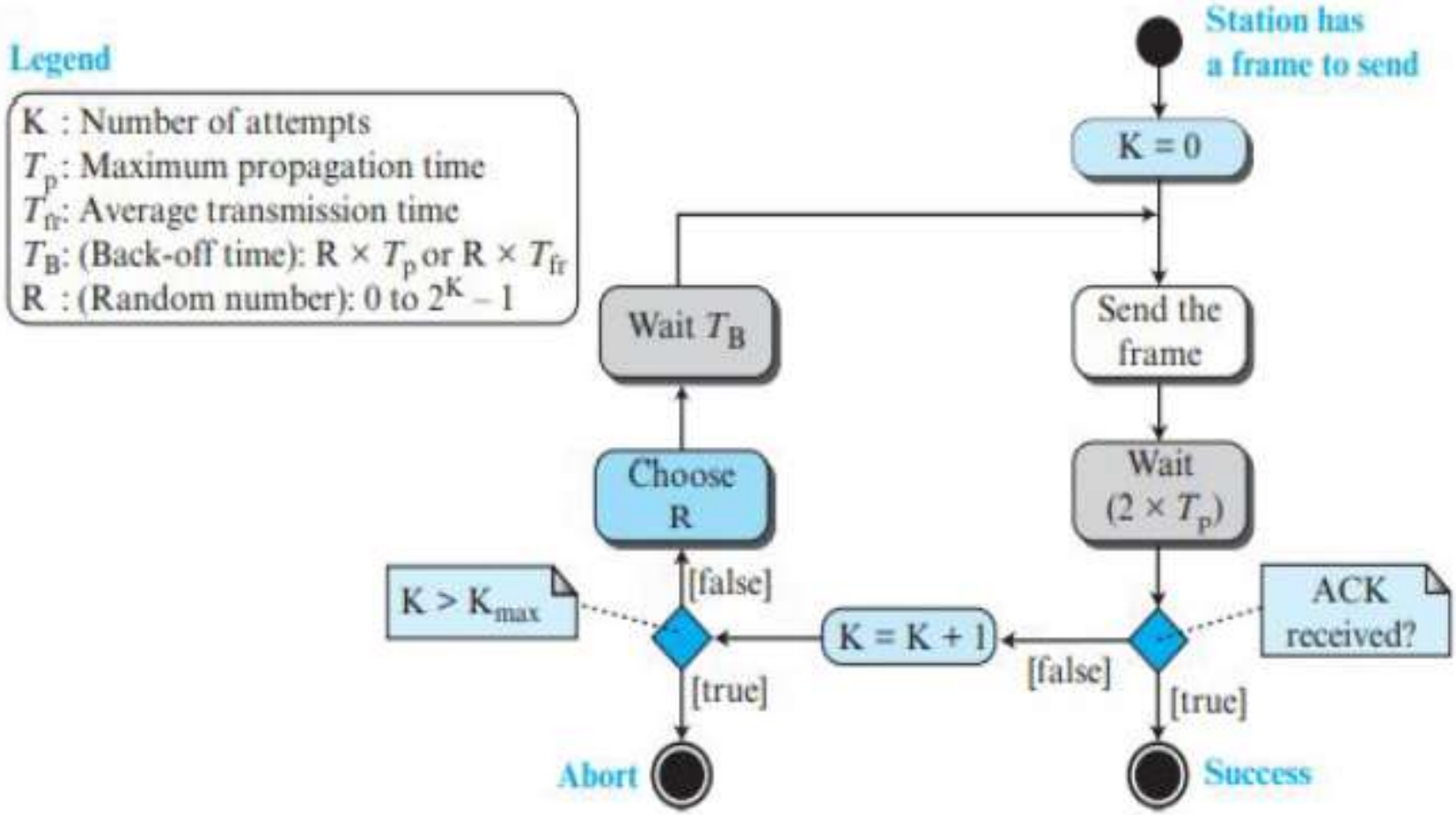
Case-02:

- Transmitting station does not receive any acknowledgement within specified time from the receiving station.
- In this case, transmitting station assumes that the transmission is unsuccessful.

Then,

- Transmitting station uses a **Back Off Strategy** and waits for some random amount of time.
- After back off time, it transmits the data packet again.
- It keeps trying until the back off limit is reached after which it aborts the transmission.





[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

Pure ALOHA

- The basic idea of an ALOHA system is simple: let users transmit whenever they have data to be sent.
- There will be collisions, and the colliding frames will be destroyed.
- If the frame was destroyed,
  - Sender just waits a random amount of time
  - Sends the frame again.
- Whenever two frames try to occupy the channel at the same time, there will be a collision and both will be garbled.
- Systems in which multiple users share a common channel in a way that can lead to conflicts are known as **contention systems**.

Pure ALOHA - collision

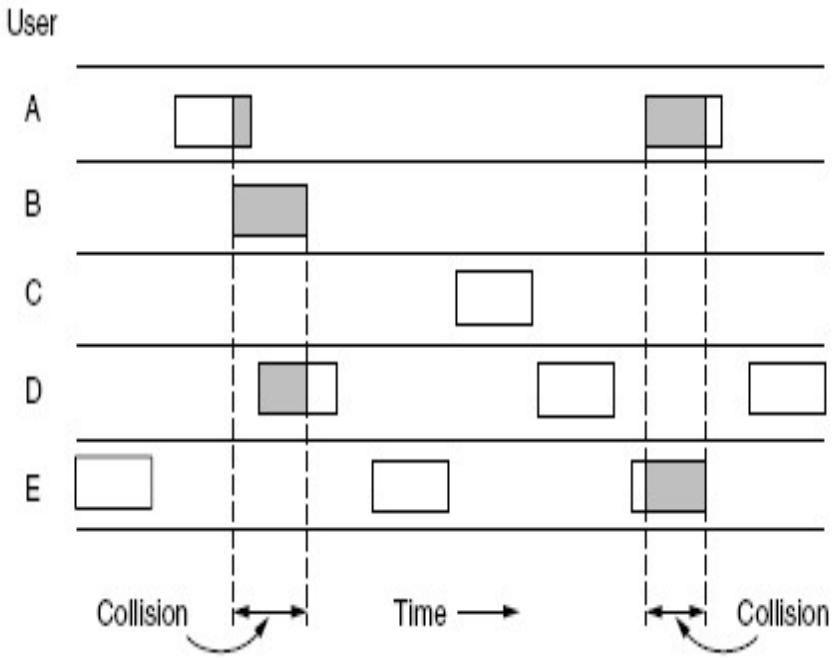


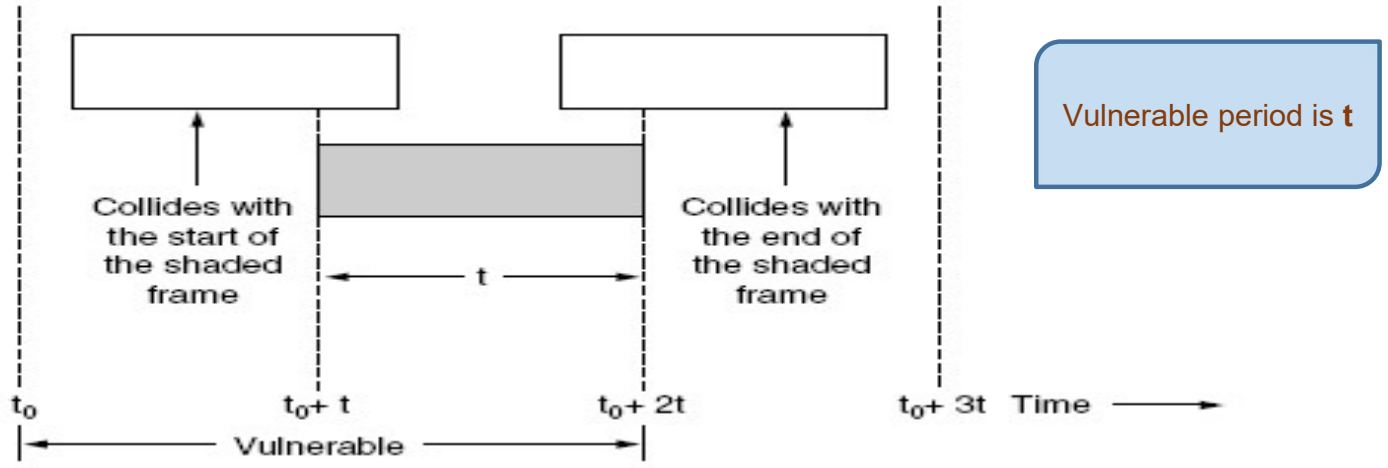
Figure 4-1. In pure ALOHA, frames are transmitted at completely arbitrary times.

- Whenever two frames try to occupy the channel at the same time, there will be a collision and both will be garbled.
- If the first bit of a new frame overlaps with just the last bit of a frame almost finished, both frames will be totally destroyed and both will have to be retransmitted later.
- The checksum cannot (and should not) distinguish between a total loss and a near miss. Bad is bad.

[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

Pure ALOHA

- A frame will not suffer a collision if no other frames are sent within one frame time of its start.



- *t* - the time required to send one frame.
- If any other user has generated a frame between time t_0 and $t_0 + t$, the end of that frame will collide with the beginning of the shaded one.
- Any frame started between t_0+t and t_0+2t will bump into the end of the shaded frame.
- Since in pure ALOHA a station does not listen to the channel before transmitting, it has no way of knowing that another frame was already underway.

Pure ALOHA

- Throughput

$$S = Ge^{-2G}$$

where G is the Mean no. of frames per frame time

- The relation between the offered traffic and the throughput is shown as

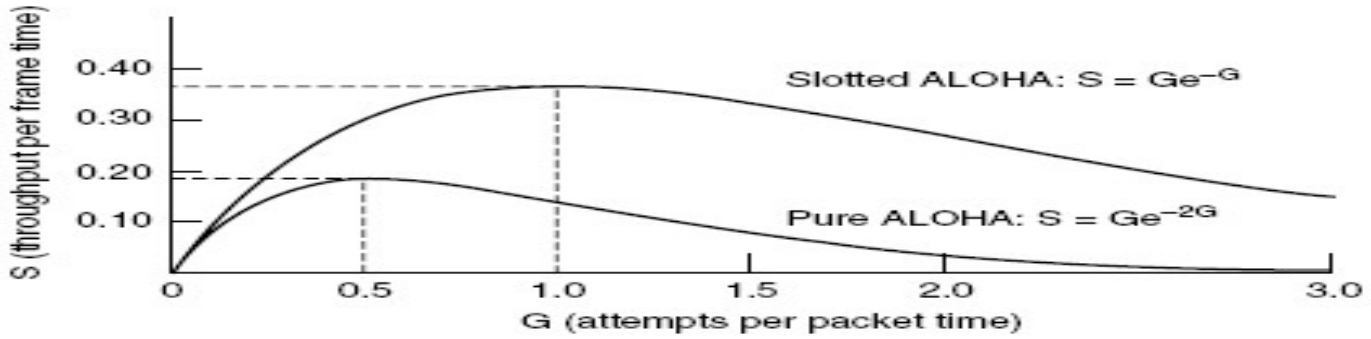


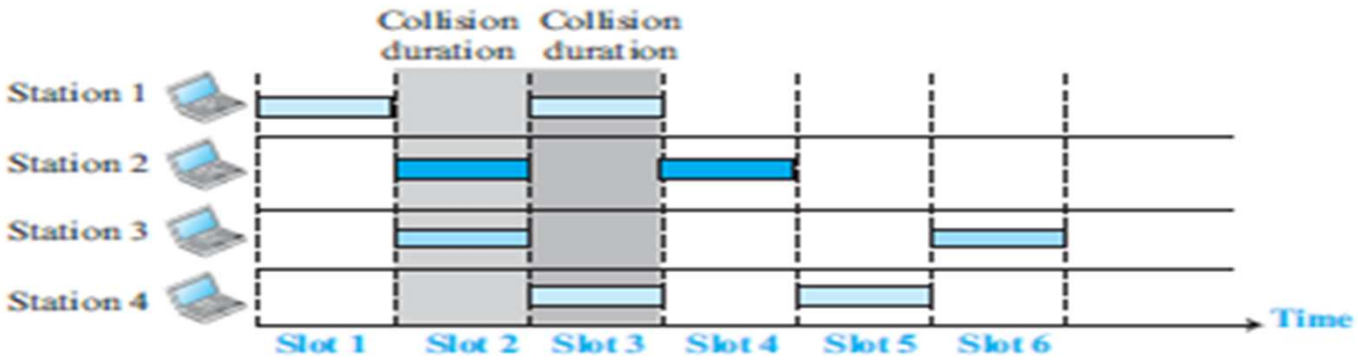
Figure 4-3. Throughput versus offered traffic for ALOHA systems.

- Maximum throughput occurs at *G = 0.5, with S = 1/2e, which is* about 0.184. (e=2.71)
- So the channel utilization is 18%.

2. Slotted Aloha-

- Slotted Aloha divides the time of shared channel into discrete intervals called as **time slots**.
- Any station can transmit its data in any time slot.
- The only condition is that station must start its transmission from the beginning of the time slot.
- If the beginning of the slot is missed, then station has to wait until the beginning of the next time slot.
- A collision may occur if two or more stations try to transmit data at the beginning of the same time slot.

Figure 5.32 *Frames in a slotted ALOHA network*



[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

Slotted ALOHA

- Divide time into discrete intervals called **slots,** each interval corresponding to one frame.

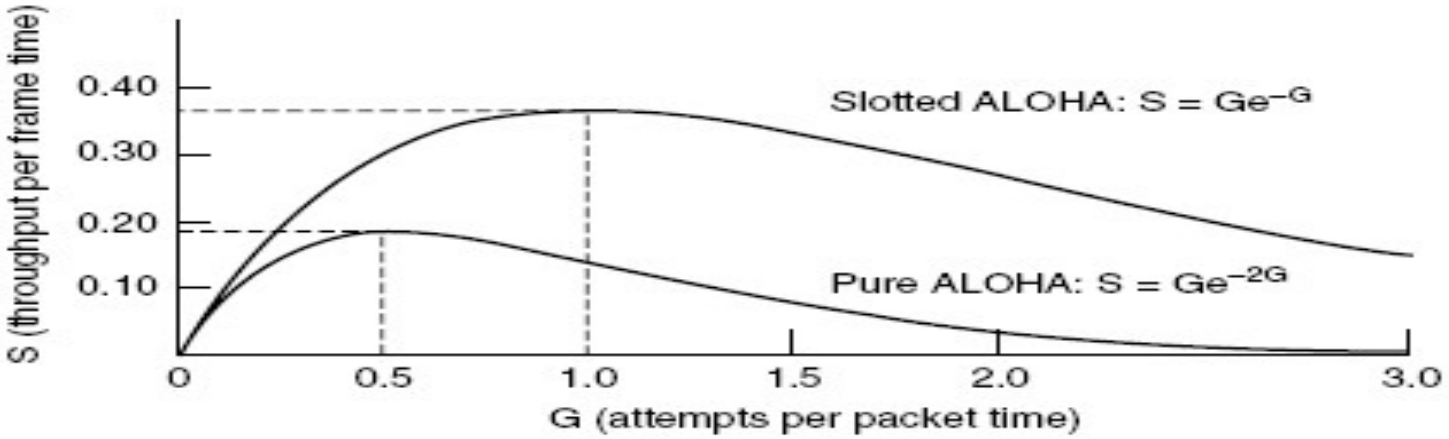


Figure 4-3. Throughput versus offered traffic for ALOHA systems.

- Throughput:

$$S = Ge^{-G}$$

where G is the Mean no. of frames per frame time

- Maximum throughput occurs at *G = 1, with S = 1/e, which is* about 0.368.
- So the channel utilization is 36%.

Difference Between Pure Aloha And Slotted Aloha-

Pure Aloha	Slotted Aloha
Any station can transmit the data at any time.	Any station can transmit the data at the beginning of any time slot.
The time is continuous and not globally synchronized.	The time is discrete and globally synchronized.
Vulnerable time in which collision may occur $= 2 \times T_t$	Vulnerable time in which collision may occur $= T_t$
Probability of successful transmission of data packet $= G \times e^{-2G}$	Probability of successful transmission of data packet $= G \times e^{-G}$
Maximum efficiency = 18.4% (Occurs at G = 1/2)	Maximum efficiency = 36.8% (Occurs at G = 1)
The main advantage of pure aloha is its simplicity in implementation.	The main advantage of slotted aloha is that it reduces the number of collisions to half and doubles the efficiency of pure aloha.

Problem-

A group of N stations share 100 Kbps slotted ALOHA channel. Each station output a 500 bits frame on an average of 5000 ms even if previous one has not been sent. What is the required value of N?

Solution-

Throughput Of One Station-

Throughput of each station

= Number of bits sent per second

= 500 bits / 5000 ms

= 500 bits / (5000 x 10^{-3} sec)

= 100 bits/sec

Throughput Of Slotted Aloha-

Throughput of slotted aloha

= Efficiency x Bandwidth

= 0.368 x 100 Kbps

= 36.8 Kbps

Total Number Of Stations-

Throughput of slotted aloha = Total number of stations x Throughput of each station

Substituting the values, we get-

36.8 Kbps = N x 100 bits/sec

∴ N = 368

HOME WORK [SOLUTION:PAGE:434-435

- A slotted ALOHA network transmits 200-bit frames using a shared channel with a 200-kbps bandwidth. Find the throughput if the system (all stations together) produces
- a. 1000 frames per second.
- b. 500 frames per second.
- c. 250 frames per second.

Carrier Sense Multiple Access (CSMA):

- To minimize the chance of collision and, therefore, increase the performance, the CSMA method was developed.
- Carrier sense multiple access (CSMA) requires that each station first listen to the medium (or check the state of the medium) before sending.

Principle of CSMA: **"sense before transmit" or "listen before talk."**

- Carrier busy: some transmission is going on
- Carrier idle: currently no transmission at all
- If the station finds the carrier free, it starts transmitting its data packet otherwise not.

CSMA

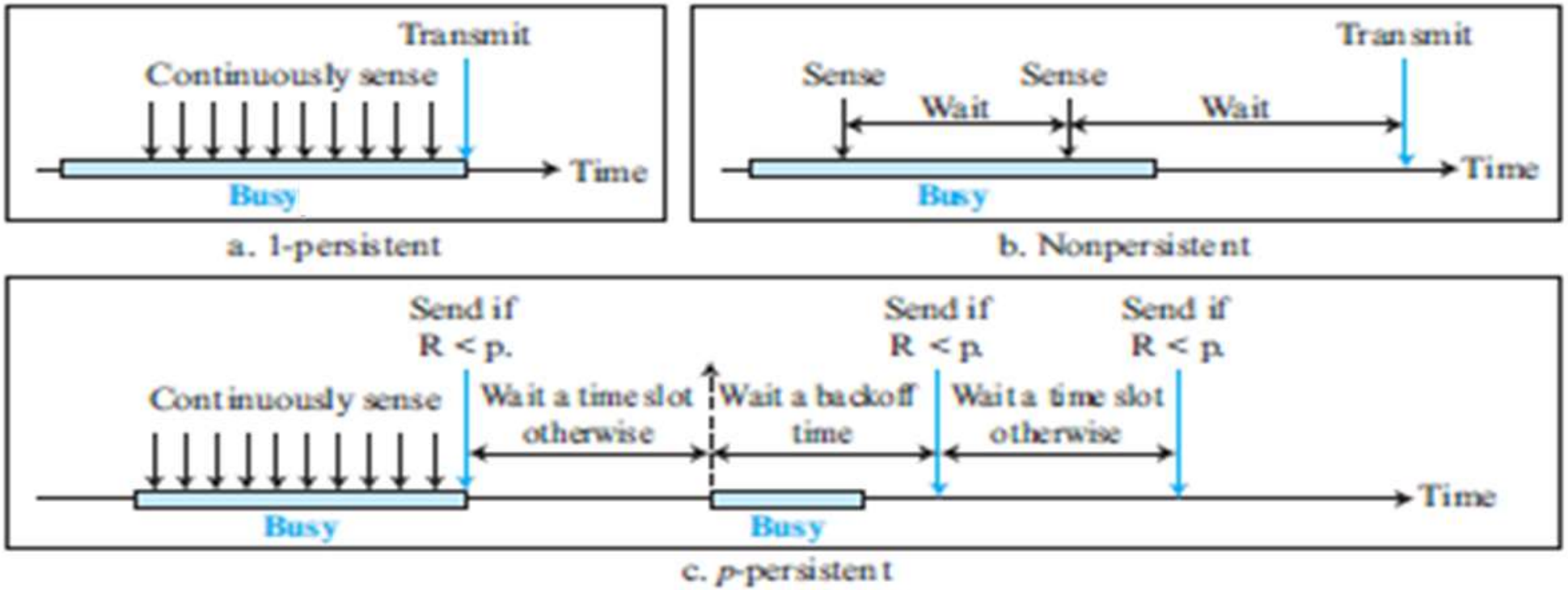
- Carrier sense multiple access (CSMA) requires that each station first listen to the medium (or check the state of the medium) before sending.
- The station is required to first sense the medium (for idle or busy) before transmitting data.
- If it is idle then it sends data.
- Otherwise it waits till the channel becomes idle.
- However there is still chance of collision in CSMA due to propagation delay.
  - For example, if station A wants to send data, it will first sense the medium. If it finds the channel idle, it will start sending data. However, by the time the first bit of data is transmitted (delayed due to propagation delay) from station A, if station B requests to send data and senses the medium it will also find it idle and will also send data. This will result in collision of data from station A and B.

CSMA access modes-

- **1-persistent:** The node senses the channel, if idle it sends the data, otherwise it continuously keeps on checking the medium for being idle and transmits unconditionally (with 1 probability) as soon as the channel gets idle.
- **Non-Persistent:** The node senses the channel, if idle it sends the data, otherwise it checks the medium after a random amount of time (not continuously) and transmits when found idle.
- **P-persistent:** The node senses the medium, if idle it sends the data with p probability. If the data is not transmitted ((1-p) probability) then it waits for some time and checks the medium again, now if it is found idle then it send with p probability. This repeat continues until the frame is sent. It is used in Wifi and packet radio systems.
- **O-persistent:** Superiority of nodes is decided beforehand and transmission occurs in that order. If the medium is idle, node waits for its time slot to send data.

Persistence Methods

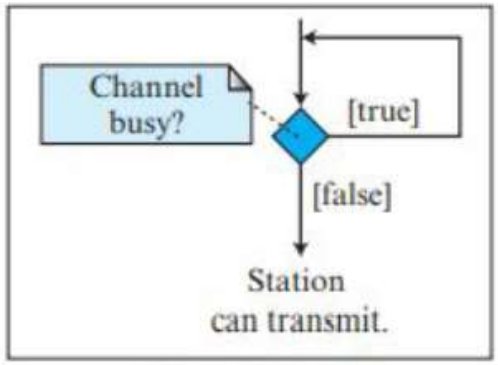
Figure 5.36 *Behavior of three persistence methods*



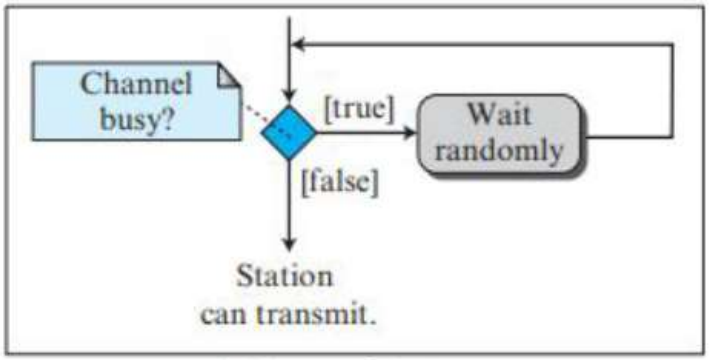
[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

Persistence Methods

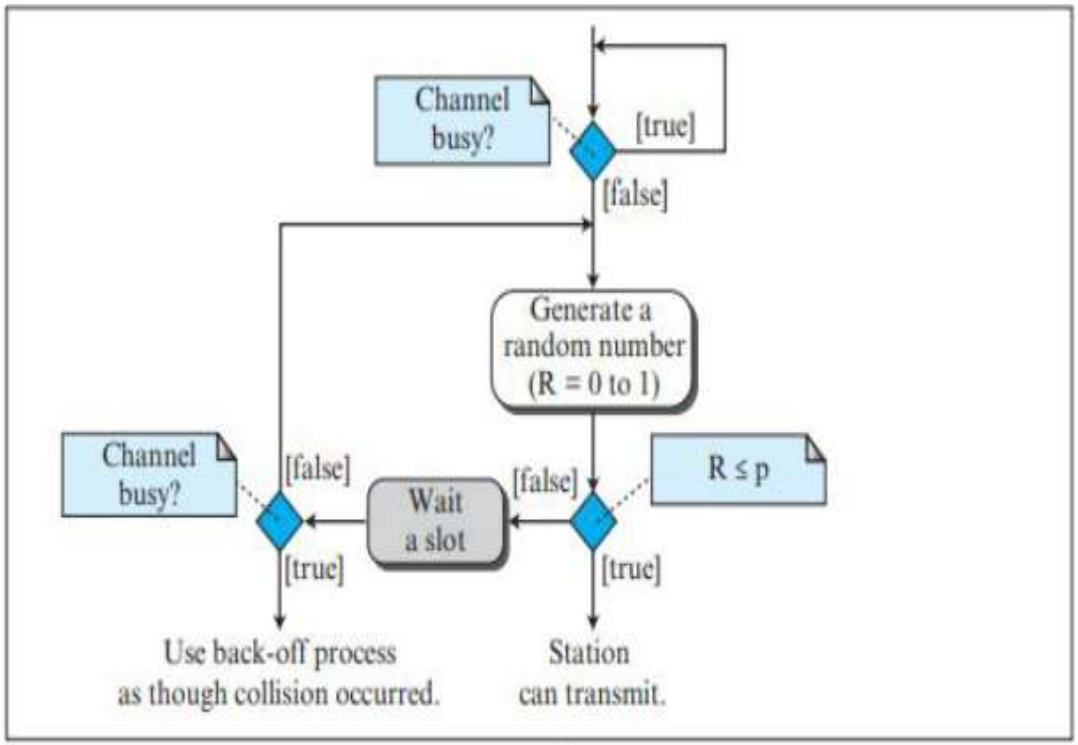
Flow diagram for three persistence methods



a. 1-persistent



b. Nonpersistent



c. *p*-persistent

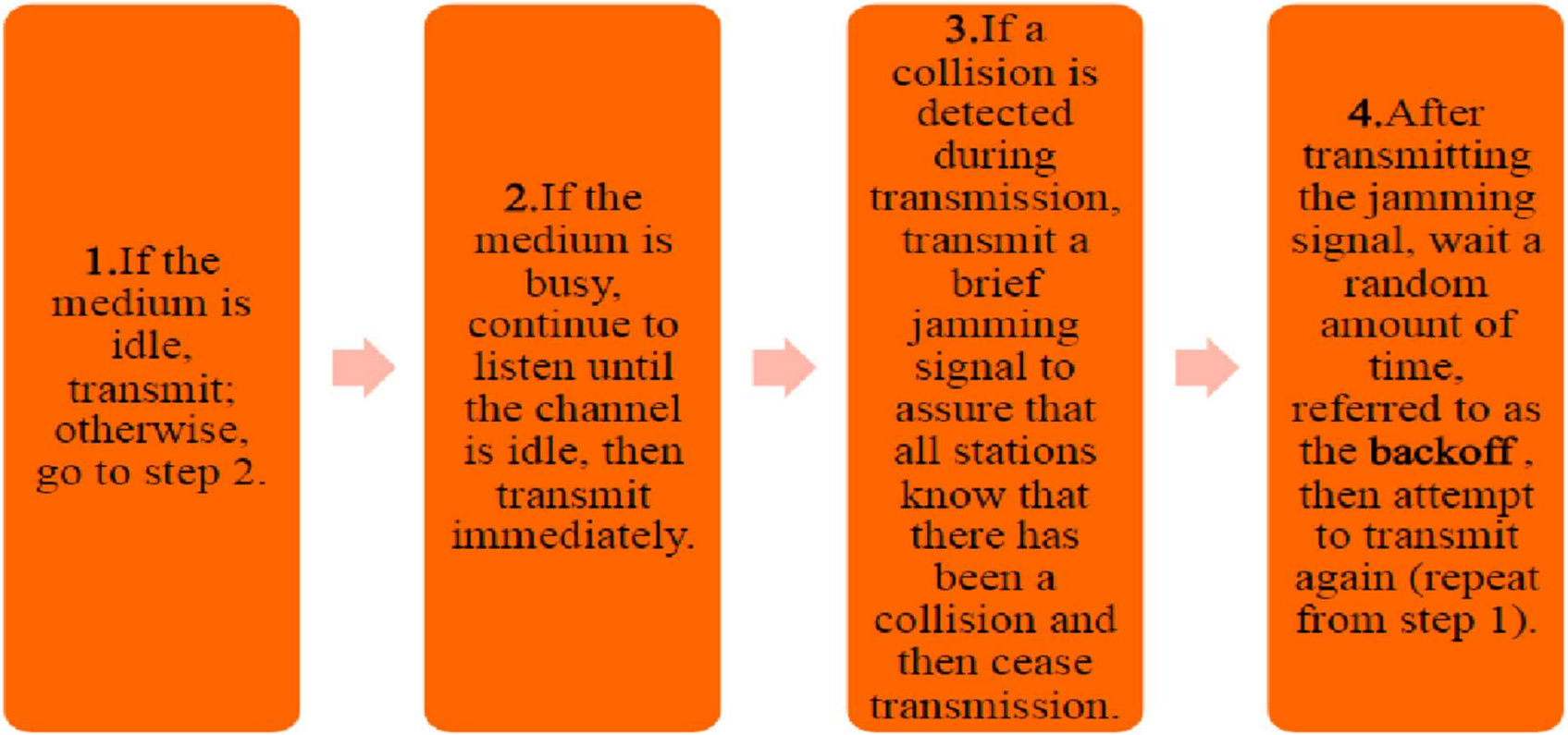
[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

Contention Protocols

- **CSMA** (Carrier Sense Multiple Access)
  - Improvement: Start transmission only if no transmission is ongoing
- **CSMA/CD** (CSMA with Collision Detection)
  - Improvement: Stop ongoing transmission if a collision is detected
- **CSMA/CA** (CSMA with Collision Avoidance)
  - Improvement: Wait a random time and try again when carrier is quiet. If still quiet, then transmit
- **CSMA/CA with ACK**
- **CSMA/CA with RTS/CTS**

RTS (request to send)
CTS (clear to send)

Description of CSMA/CD



CSMA/CD (CSMA with Collision Detection)

- In CSMA, if 2 terminals begin sending packet at the same time, each will transmit its complete packet (although collision is taking place).
- Wasting medium for an entire packet time.

- CSMA/CD

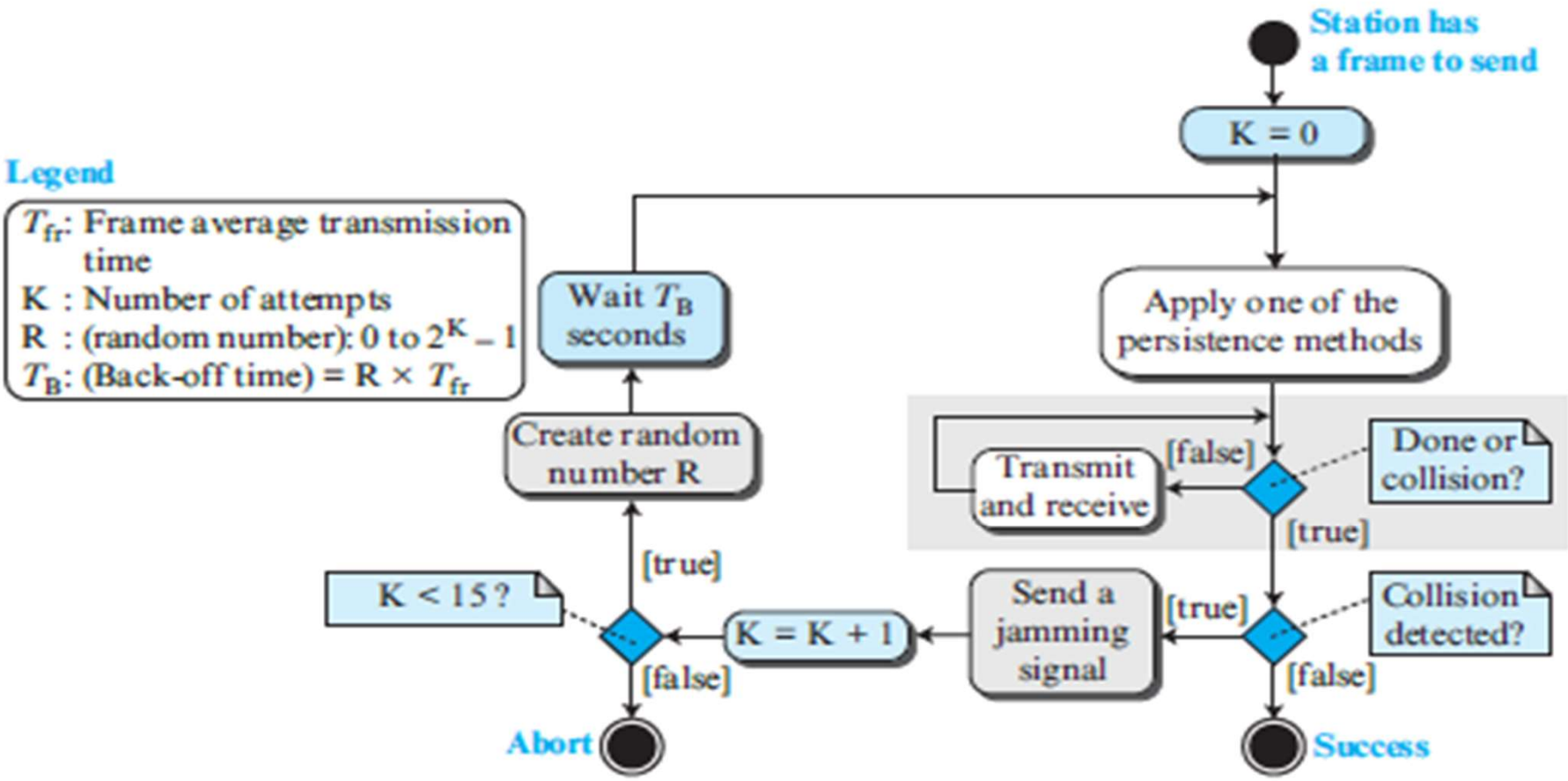
  Step 1: If the medium is idle, transmit

  Step 2: If the medium is busy, continue to listen until the channel is idle then transmit

  Step 3: If a collision is detected during transmission, cease transmitting

  Step 4: Wait a random amount of time and repeats the same algorithm

Figure 5.40 *Flow diagram for the CSMA/CD*

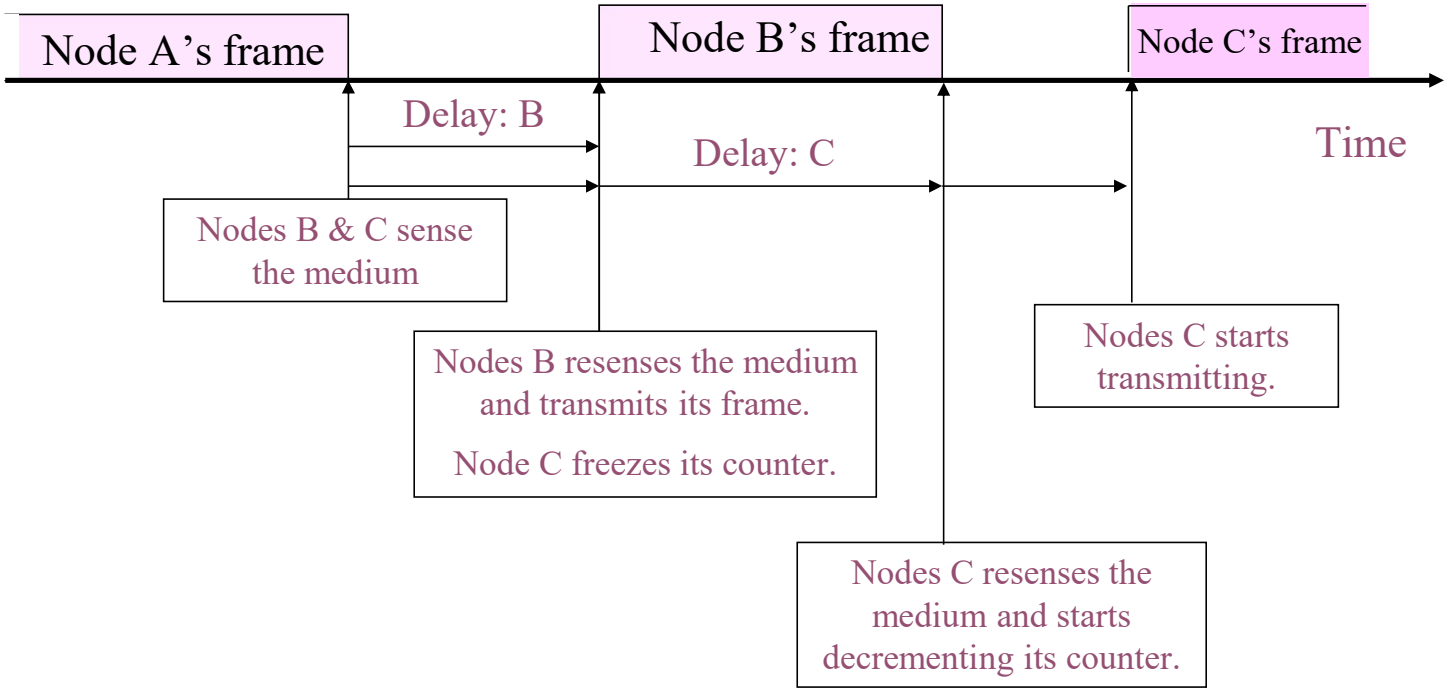


[Behrouz A Forouzan, Firouz Mosharraf, "Computer Networks: A top down Approach", McGraw Hill Education]

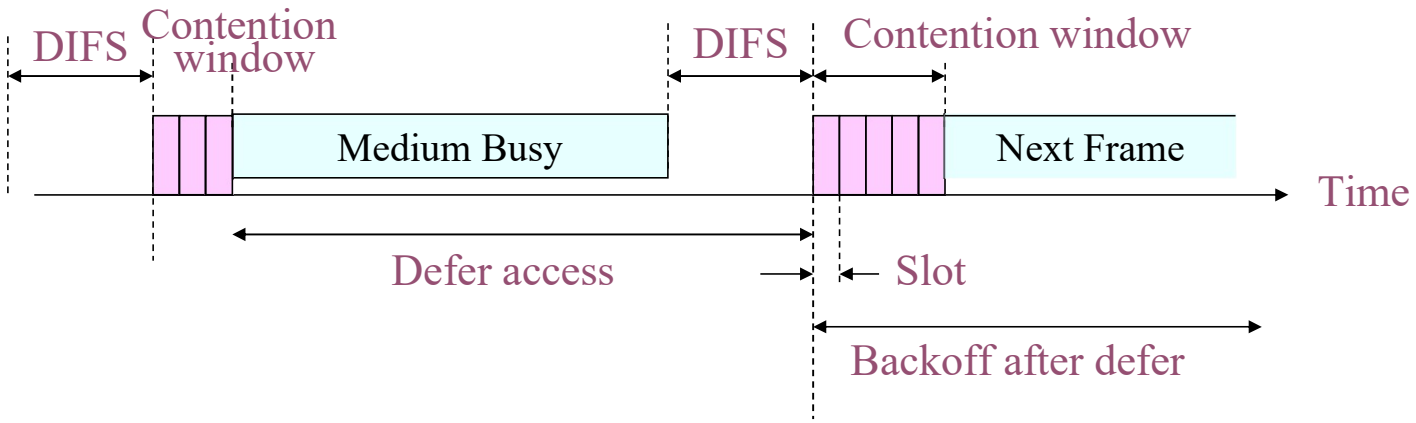
CSMA/CA (CSMA with collision Avoidance)

- All terminals listen to the same medium as CSMA/CD.
- Terminal ready to transmit senses the medium.
- If medium is busy it waits until the end of current transmission.
- It again waits for an additional predetermined time period DIFS (Distributed inter frame Space).
- Then picks up a random number of slots (the initial value of backoff counter) within a contention window to wait before transmitting its frame.
- If there are transmissions by other terminals during this time period (backoff time), the terminal freezes its counter.
- It resumes count down after other terminals finish transmission + DIFS. The terminal can start its transmission when the counter reaches to zero.

CSMA/CA (Cont'd)



CSMA/CA Explained

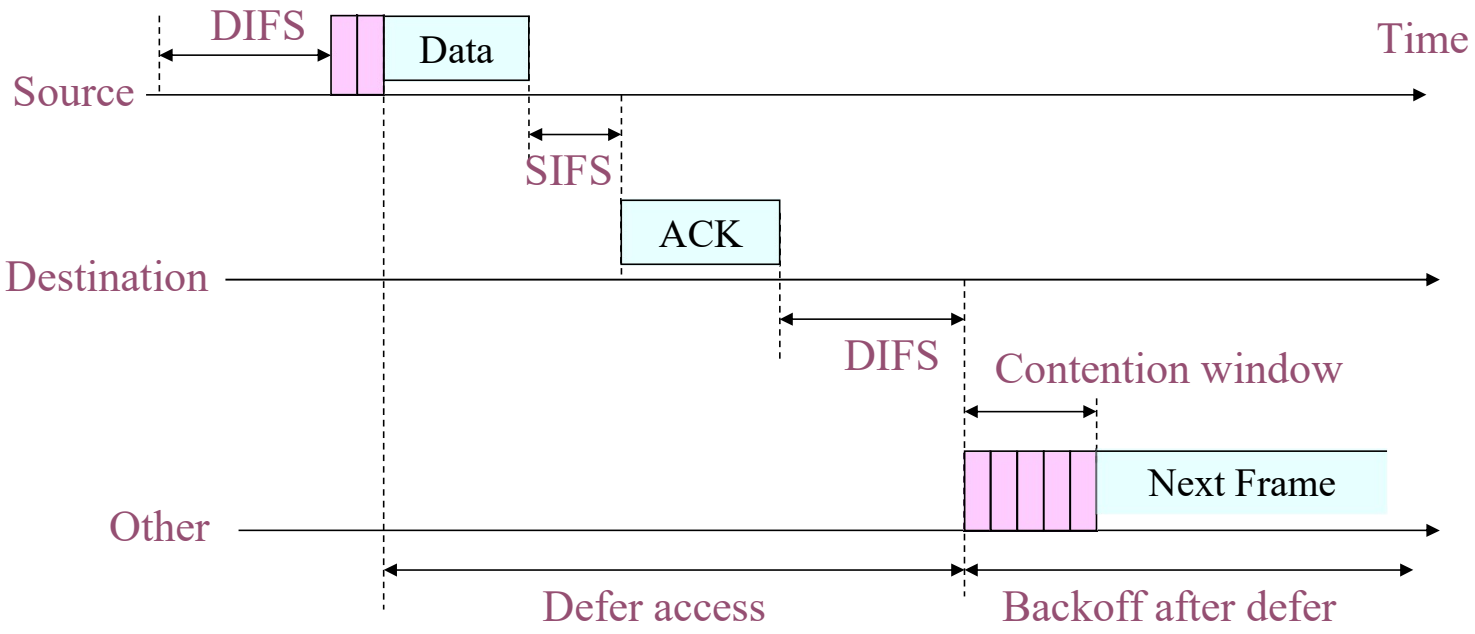


DIFS – Distributed Inter Frame Spacing

CSMA/CA with ACK

- Immediate Acknowledgements from receiver upon reception of data frame without any need for sensing the medium.
- ACK frame transmitted after time interval SIFS (*Short Inter-Frame Space*) (*SIFS < DIFS*)
- Receiver transmits ACK without sensing the medium.
- If ACK is lost, retransmission done.

CSMA/CA/ACK

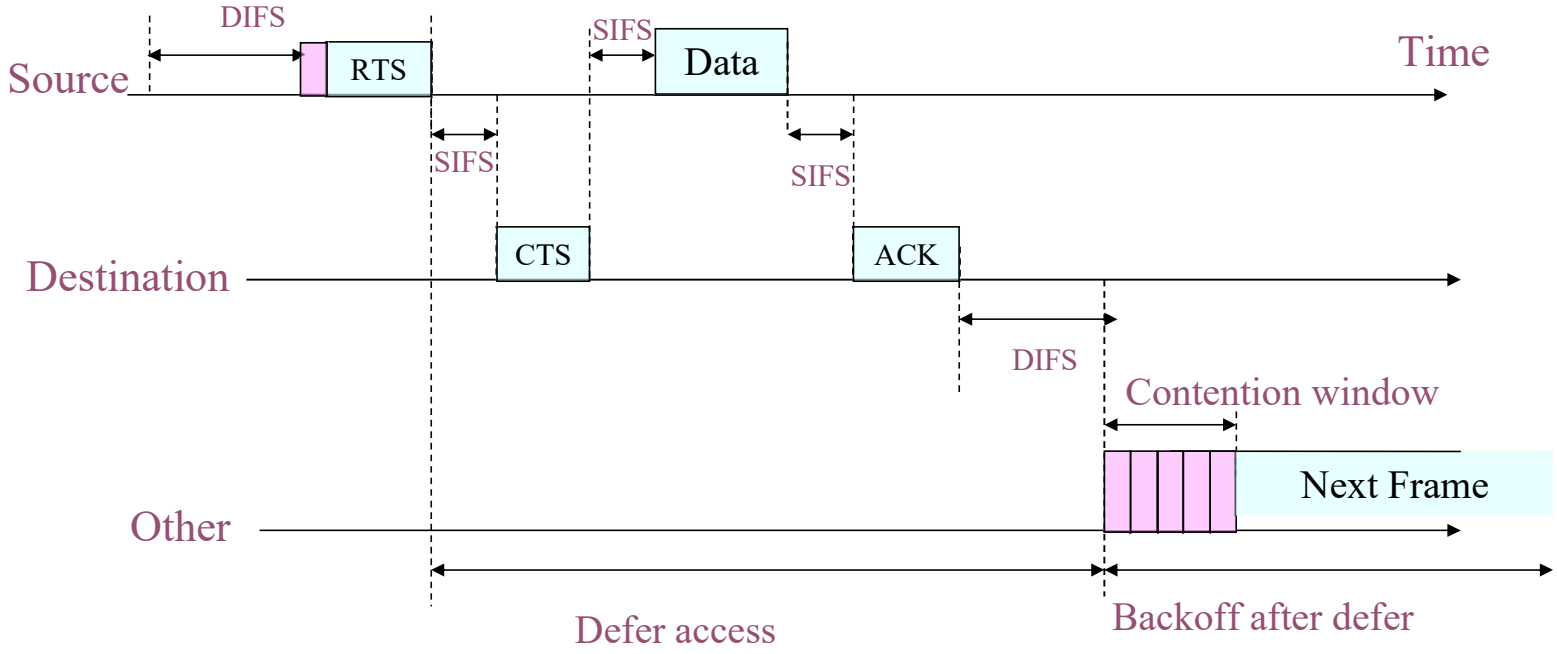


SIFS – Short Inter Frame Spacing

CSMA/CA with RTS/CTS

- Transmitter sends an RTS (request to send) after medium has been idle for time interval more than DIFS.
- Receiver responds with CTS (clear to send) after medium has been idle for SIFS.
- Then Data is exchanged.
- RTS/CTS is used for reserving channel for data transmission so that the collision can only occur in control message.

CSMA/CA with RTS/CTS (Cont'd)



RTS/CTS