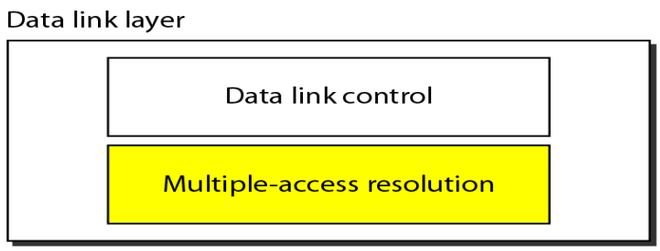
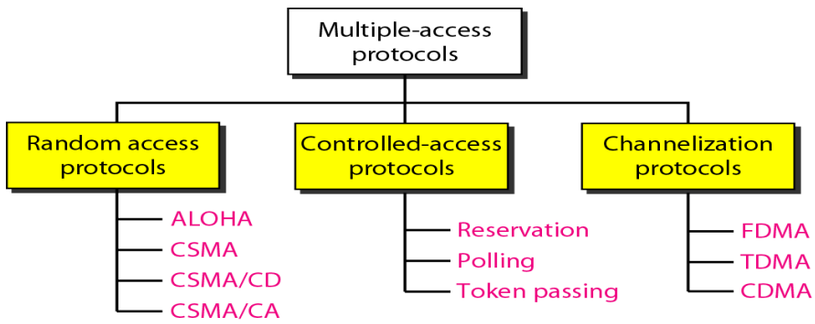
COMPUTER NETWORKS
UNIT-IV

Introduction:

- Up to now we discuss about data link control, a mechanism which provides a link with reliable communication.
- In the protocols we described, we assumed that there is an available dedicated link (or channel) between the sender and the receiver.
- This assumption mayor may not be true.
- If, indeed, we have a dedicated link, as when we connect to the Internet using PPP as the data link control protocol, then the assumption is true and we do not need anything else.
- On the other hand, if we use our cellular phone to connect to another cellular phone, the channel (the band allocated to the vendor company) is not dedicated.
- A person a few feet away from us may be using the same channel to talk to her friend.
- We can consider the data link layer as **two sublayers**.
- The upper sublayer is responsible for data link control, and the lower sublayer is responsible for resolving access to the shared media.
- If the channel is dedicated, we do not need the lower sublayer. Figure shows these two sublayers in the data link layer.
- The upper sublayer that is responsible for flow and error control is called the **logical link control (LLC)** layer; the lower sublayer that is mostly responsible for multiple access resolution is called the **media access control (MAC)** layer.
- 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.



- Many formal protocols have been devised to handle access to a shared link. We categorize them into three groups.



RANDOM ACCESS PROTOCOLS:

- In random access or contention methods, no station is superior to another station and none is assigned the control over another.
- No station permits, or does not permit, another station to send.
- At each instance, a station that has data to send uses a procedure defined by the protocol to make a decision on whether or not to send.
- This decision depends on the state of the medium (idle or busy).
- In other words, each station can transmit when it desires on the condition that it follows the predefined procedure, including the testing of the state of the medium.
- First, there is no scheduled time for a station to transmit.
- Transmission is random among the stations. That is why these methods are called ***random access***.
- Second, no rules specify which station should send next.
- Stations compete with one another to access the medium. That is why these methods are also called ***contention methods***.
- In a random access method, each station has the right to the medium without being controlled by any other station.
- However, if more than one station tries to send, there is an access conflict-collision-and the frames will be either destroyed or modified.

- To avoid access conflict or to resolve it when it happens, 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 evolved from a very interesting protocol known as **ALOHA**, which used a very simple procedure called **multiple access (MA)**.
- The method was improved with the addition of a procedure that forces the station **to sense** the medium before transmitting. This was called **carrier sense multiple access**.
- This method later evolved into two parallel methods: carrier sense multiple access with collision detection (CSMA/CD) and carrier sense multiple access with collision avoidance *(CSMA/CA). CSMA/CD* tells the station what to do when a collision is detected. CSMA/CA tries to avoid the collision.

ALOHA:

- ALOHA, the earliest random access method was developed at the University of Hawaii in early 1970.
- It was designed for a radio (wireless) LAN, but it can be used on any shared medium.
- It is obvious that there are potential collisions in this arrangement.
- The medium is shared between the stations.
- When a station sends data, another station may attempt to do so at the same time.
- The data from the two stations collide and become garbled.

PURE ALOHA:

- The original ALOHA protocol is called pure ALOHA.
- This is a simple, but elegant protocol.
- The idea is that each station sends a frame whenever it has a frame to send.
- However, since there is only one channel to share, there is the possibility of collision between frames from different stations.

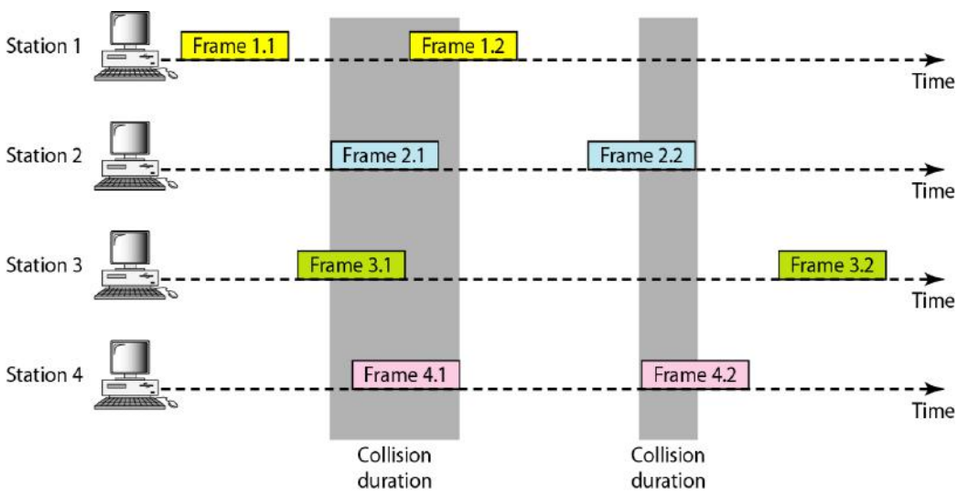


Figure: Frames in a pure ALOHA network

- In pure ALOHA that even if one bit of a frame coexists on the channel with one bit from another frame, there is a collision and both will be destroyed.
- It is obvious that we need to resend the frames that have been destroyed during transmission.
- The pure ALOHA protocol relies on acknowledgments from the receiver.
- When a station sends a frame, it expects the receiver to send an acknowledgment.
- If the acknowledgment does not arrive after a time-out period, the station assumes that the frame (or the acknowledgment) has been destroyed and resends the frame.
- A collision involves two or more stations. If all these stations try to resend their frames after the time-out, the frames will collide again.
- 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. We call this time 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 $\mathbf{K_{max}}$ a station must give up and try later.

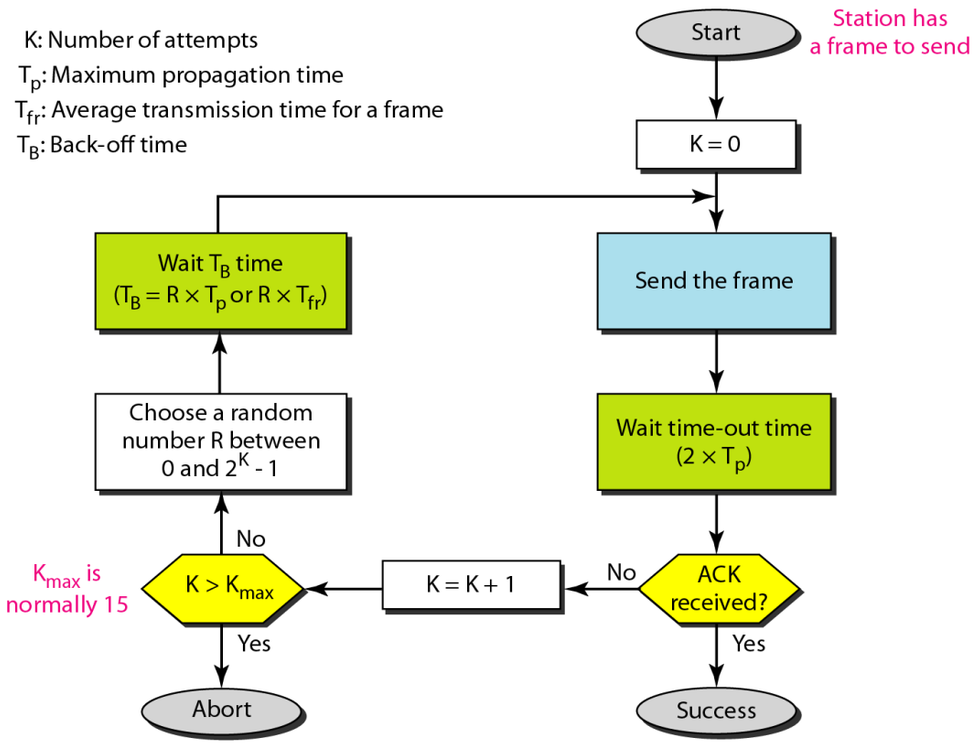


Figure: Procedure for pure ALOHA protocol

- The time-out period is equal to the maximum possible round-trip propagation delay, which is twice the amount of time required to send a frame between the two most widely separated stations (2 x $\mathbf{T_p}$).
- The back-off time T_B is a random value that normally depends on *K (the number of attempted unsuccessful transmissions).*
- The formula for T_B depends on the implementation. One common formula is the **binary exponential back-off.**
- In this method, for each retransmission, a multiplier in the range 0 to $2^K - 1$ is randomly chosen and multiplied by $\mathbf{T_p}$ (maximum propagation time) or $\mathbf{T_{fr}}$ (the average time required to send out a frame) to find TB.
- Note that in this procedure, the range of the random numbers increases after each collision.
- The value of $\mathbf{K_{max}}$ is usually chosen as 15.

Vulnerable time:

- Let us find the **length of time**, the **vulnerable time**, in which there is a possibility of collision.
- We assume that the stations send fixed-length frames with each frame taking $\mathbf{T_{fr}}$S to send. Figure shows the vulnerable time for station A.

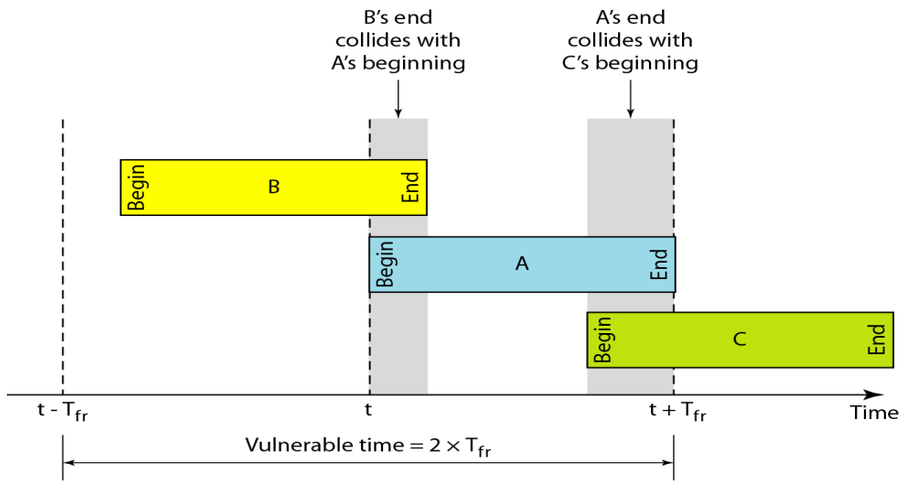


Figure : Vulnerable time for pure ALOHA protocol

- The vulnerable time, during which a collision may occur in pure ALOHA, is 2 times the frame transmission time.

Pure ALOHA vulnerable time = 2 x $\mathbf{T_{fr}}$

Throughput:

- Let us call **G** the **average number of frames** generated by the system during one frame transmission time.
- Then it can be proved that the average number of successful transmissions for pure ALOHA is $\mathbf{S = G \times e^{-2G}}$.
- The maximum throughput S_{max} is 0.184, for G = 1.
- In other words, if one-half a frame is generated during one 2 frame transmission time (in other words, one frame during two frame transmission times), then 18.4 percent of these frames reach their destination successfully.

- This is an expected result because the vulnerable time is 2 times the frame transmission time.
- Therefore, if a station generates only one frame in this vulnerable time (and no other stations generate a frame during this time), the frame will reach its destination successfully.

SLOTTED ALOHA:

- Pure ALOHA has a vulnerable time of 2 x T_{fr} .
- This is so because there is no rule that defines when the station can send.
- A station may send soon after another station has started or soon before another station has finished.
- Slotted ALOHA was invented to improve the efficiency of pure ALOHA.
- In slotted ALOHA we divide the time into slots of T_{fr} s and force the station to send only at the beginning of the time slot.

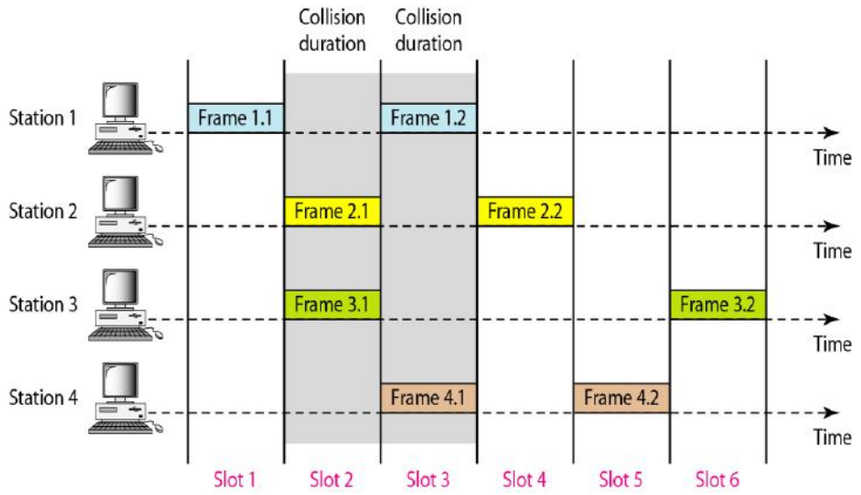
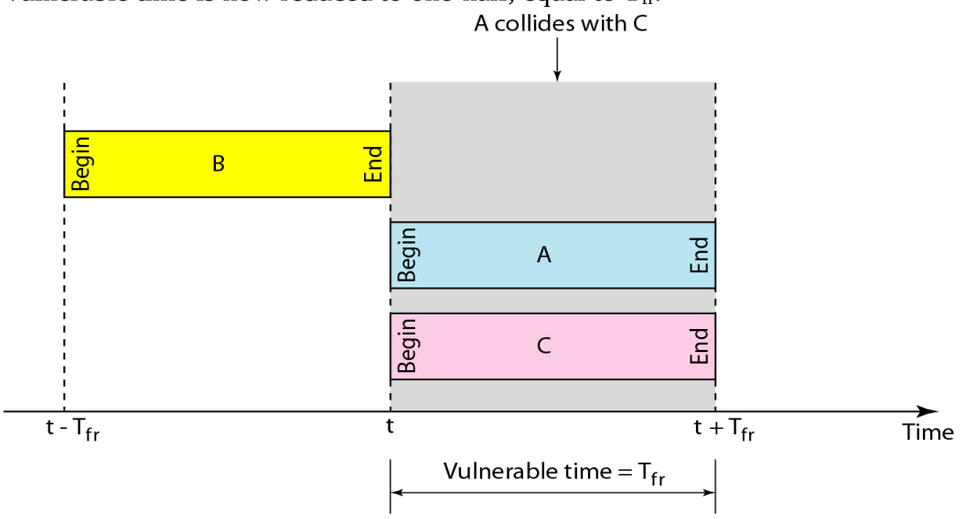


Figure: Frames in a slotted ALOHA network

- Because a station is allowed to send only at the beginning of the synchronized time slot, if a station misses this moment, it must wait until the beginning of the next time slot.
- This means that the station which started at the beginning of this slot has already finished sending its frame.
- Of course, there is still the possibility of collision if two stations try to send at the beginning of the same time slot.
- However, the vulnerable time is now reduced to one-half, equal to T_{fr}.



- The vulnerable time for slotted ALOHA is one-half that of pure ALOHA.

Slotted ALOHA vulnerable time = T_{fr}

Throughput:

- It can be proved that the average number of successful transmissions for slotted ALOHA is **$S = G \times e^{-G}$.**
- The maximum throughput S_{max} is 0.368, when G = 1

CARRIER SENSE MULTIPLE ACCESS:

- To minimize the chance of collision and, therefore, increase the performance, the CSMA method was developed.
- The chance of collision can be reduced if a station senses the medium before trying to use it.
- Carrier sense multiple access (CSMA) requires that each station first listen to the medium (or check the state of the medium) before sending.

- In other words, CSMA is based on the principle "**sense before transmit**" or "**listen before talk**".
- CSMA can reduce the possibility of collision, but it cannot eliminate it.

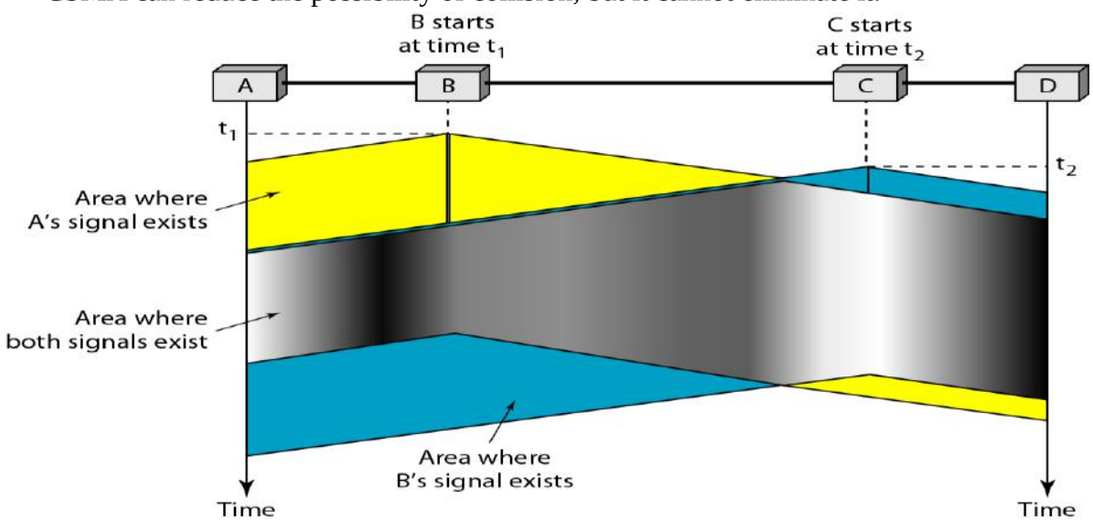


Figure: Space/time model of the collision in CSMA

- Stations are connected to a shared channel (usually a dedicated medium).
- The possibility of collision still exists because of propagation delay; when a station sends a frame, it still takes time (although very short) for the first bit to reach every station and for every station to sense it.
- In other words, a station may sense the medium and find it idle, only because the first bit sent by another station has not yet been received.
- At time t1 station B senses the medium and finds it idle, so it sends a frame.
- At time t2 (t2> t1) station C senses the medium and finds it idle because, at this time, the first bits from station B have not reached station C. Station C also sends a frame. The two signals collide and both frames are destroyed.

Vulnerable Time:

- **The vulnerable time for CSMA is the propagation time Tp .**
- This is the time needed for a signal to propagate from one end of the medium to the other.
- When a station sends a frame, and any other station tries to send a frame during this time, a collision will result.
- But if the first bit of the frame reaches the end of the medium, every station will already have heard the bit and will refrain from sending.
- Following Figure shows the worst case. The leftmost station A sends a frame at time t1 which reaches the rightmost station D at time t1 + Tp . The gray area shows the vulnerable area in time and space.

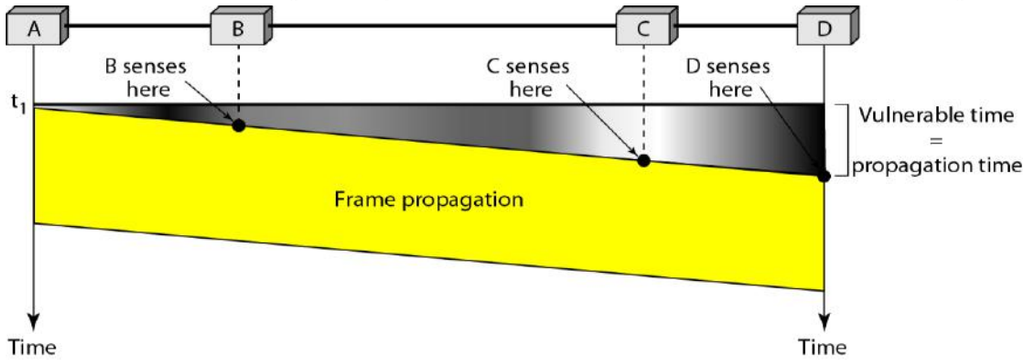
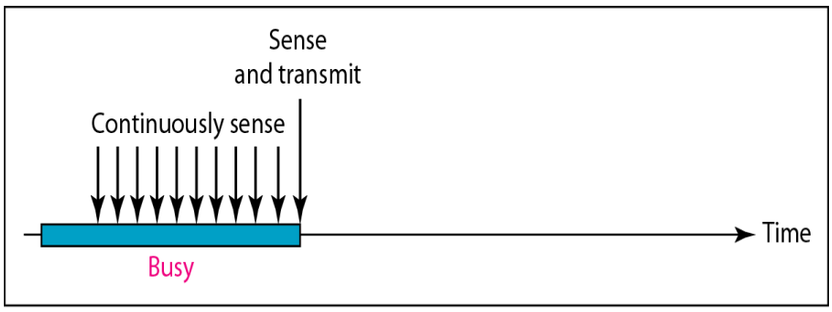


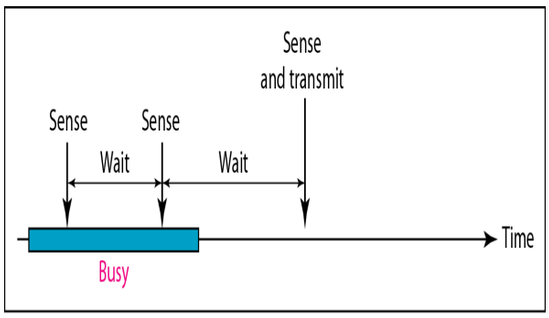
Figure: Vulnerable time in CSMA

PERSISTENCE METHODS:

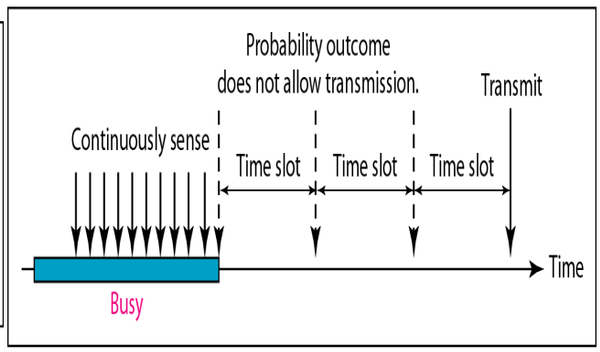
- What should a station do if the channel is busy?
- What should a station do if the channel is idle?
- Three methods have been devised to answer these questions: the **1-persistent method, the nonpersistent method, and the p-persistent method**.
- Following Figure shows the behavior of three persistence methods when a station finds a channel busy.



a. 1-persistent



b. Nonpersistent



c. p-persistent

1-persistent:

- The 1-persistent method is simple and straightforward.
- In this method, after the station finds the line idle, it sends its frame immediately (with probability 1).
- This method has the highest chance of collision because two or more stations may find the line idle and send their frames immediately.
- Ethernet uses this method.

Nonpersistent:

- In the nonpersistent method, a station that has a frame to send senses the line.
- If the line is idle, it sends immediately.
- If the line is not idle, it waits a random amount of time and then senses the line again.
- The nonpersistent approach reduces the chance of collision because it is unlikely that two or more stations will wait the same amount of time and retry to send simultaneously.
- However, this method reduces the efficiency of the network because the medium remains idle when there may be stations with frames to send.

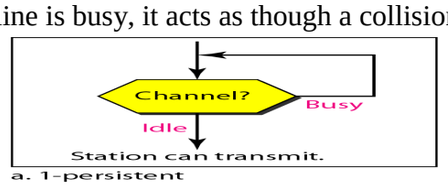
P-Persistent:

- The p-persistent method is used if the channel has time slots with a slot duration equal to or greater than the maximum propagation time.
- The p-persistent approach combines the advantages of the other two strategies.
- It reduces the chance of collision and improves efficiency.
- In this method, after the station finds the line idle it follows these steps:

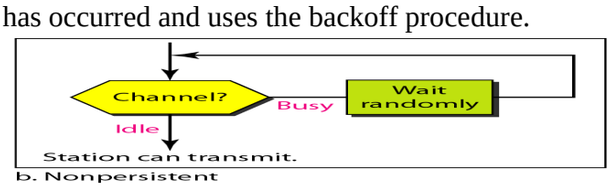
1. With probability p, the station sends its frame.
2. With probability q = 1 - p, the station waits for the beginning of the next time slot and checks the line again.

a. If the line is idle, it goes to step 1.

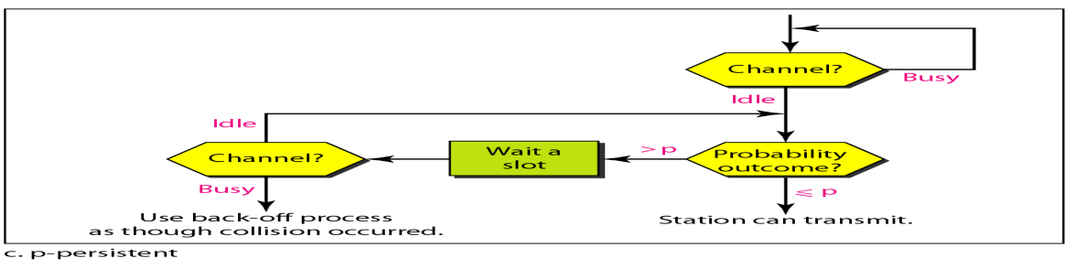
b. If the line is busy, it acts as though a collision has occurred and uses the backoff procedure.



a. 1-persistent



b. Nonpersistent



c. p-persistent

Figure: Flow diagram for three persistence methods

CARRIER SENSE MULTIPLE ACCESS WITH COLLISION DETECTION (CSMA/CD):

- The CSMA method does not specify the procedure following a collision.
- Carrier sense multiple access with collision detection (CSMA/CD) augments the algorithm to handle the collision.
- In this method, a station monitors the medium after it sends a frame to see if the transmission was successful. If so, the station is finished.
- If, however, there is a collision, the frame is sent again.
- To better understand CSMA/CD, let us look at the first bits transmitted by the two stations involved in the collision.
- Although each station continues to send bits in the frame until it detects the collision, we show what happens as the first bits collide. In Following Figure, stations A and C are involved in the collision.

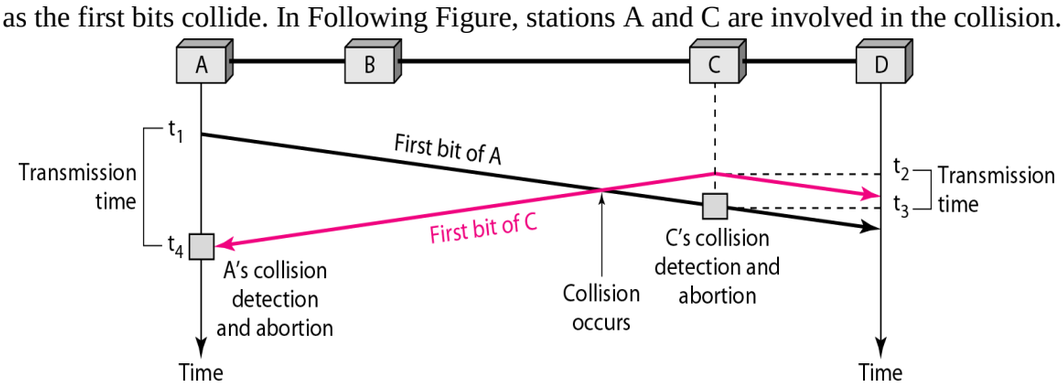


Figure: Collision of the first bit in CSMA/CD

- At time t1, station A has executed its persistence procedure and starts sending the bits of its frame.
- At time t2, station C has not yet sensed the first bit sent by A.
- Station C executes its persistence procedure and starts sending the bits in its frame, which propagate both to the left and to the right.
- The collision occurs sometime after time t2.
- Station C detects a collision at time t3 when it receives the first bit of A's frame.
- Station C immediately aborts transmission.
- Station A detects collision at time t4.
- When it receives the first bit of C's frame; it also immediately aborts transmission.
- Looking at the figure, we see that A transmits for the duration t4 - tl;
- C transmits for the duration t3 - t2.
- At time t4, the transmission of A's frame, though incomplete, is aborted; at time t3, the transmission of C's frame, though incomplete, is aborted.

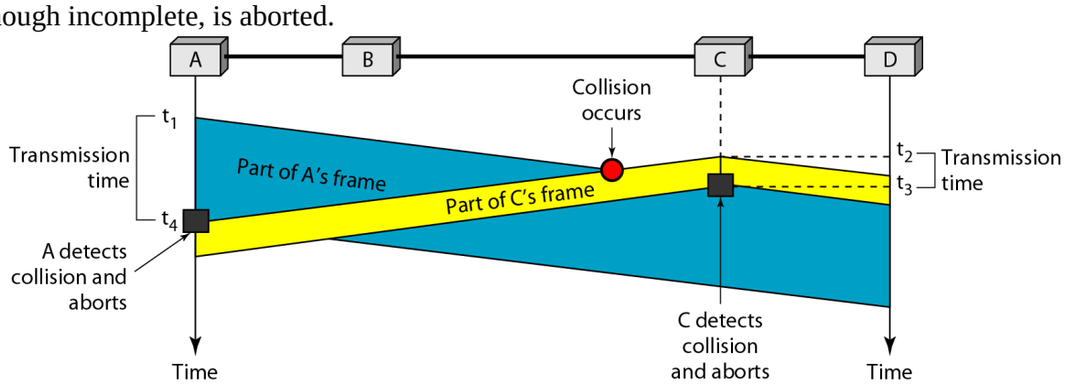


Figure: Collision and abortion in CSMA/CD

Minimum Frame Size:

- For CSMA/CD to work, we need a restriction on the frame size.
- Before sending the last bit of the frame, the sending station must detect a collision, if any, and abort the transmission.
- This is so because the station, once the entire frame is sent, does not keep a copy of the frame and does not monitor the line for collision detection.
- Therefore, **the frame transmission time T_{fr} must be at least two times the maximum propagation time T_p.**
- If the two stations involved in a collision are the maximum distance apart, the signal from the first takes time T_p to reach the second and the effect of the collision takes another time T_p to reach the first.

- So the requirement is that the first station must still be transmitting after **2Tp**.

Procedure:

- It is similar to the one for the ALOHA protocol, but there are differences.
- The **first difference** is the addition of the persistence process.
- We need to sense the channel before we start sending the frame by using one of the persistence processes.
- The **second difference** is the frame transmission.
- In ALOHA, we first transmit the entire frame and then wait for an acknowledgment.
- In CSMA/CD, transmission and collision detection is a continuous process.
- We do not send the entire frame and then look for a collision.
- The station transmits and receives continuously and simultaneously.
- We use a loop to show that transmission is a continuous process.
- We constantly monitor in order to detect one of two conditions: either transmission is finished or a collision is detected.
- Either event stops transmission.
- When we come out of the loop, if a collision has not been detected, it means that transmission is complete; the entire frame is transmitted.
- Otherwise, a collision has occurred.
- The **third difference** is the sending of a short jamming signal that enforces the collision in case other stations have not yet sensed the collision.

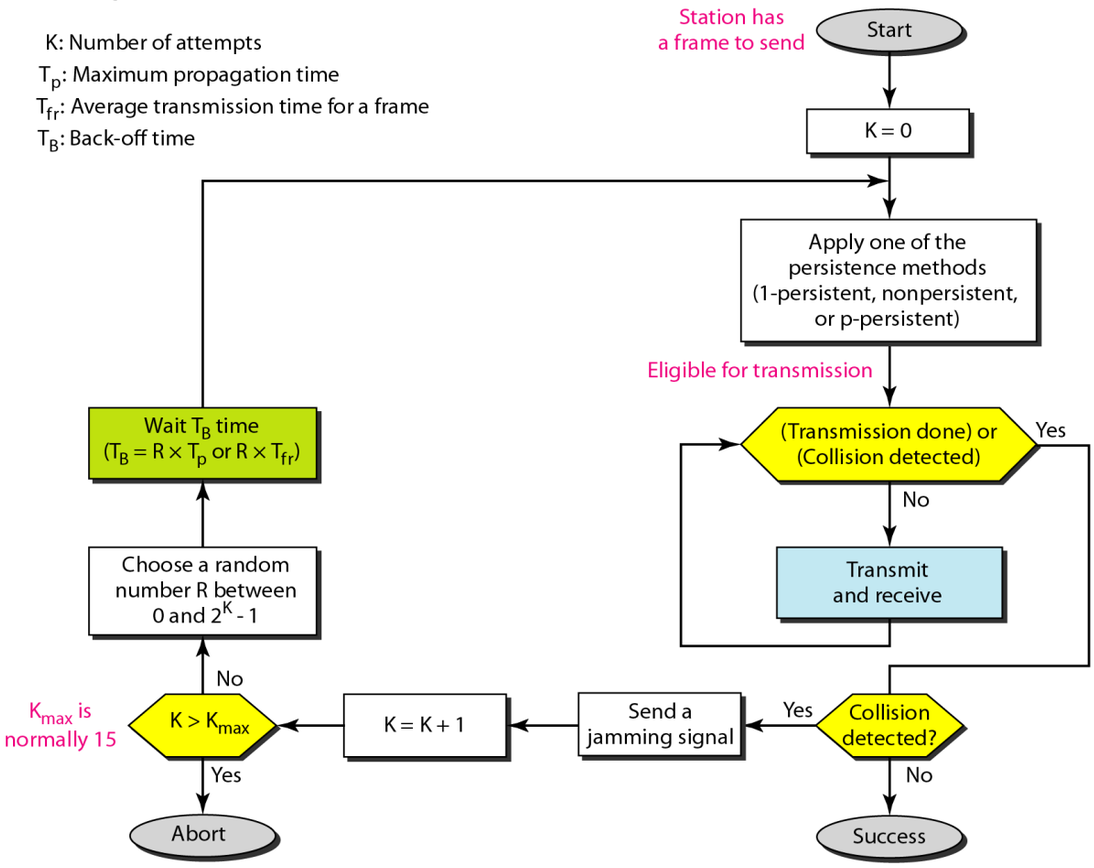


Figure: Flow diagram for the CSMA/CD

Energy Level:

- We can say that the level of energy in a channel can have three values: **zero, normal, and abnormal**.
- At the zero level, the channel is idle.
- At the normal level, a station has successfully captured the channel and is sending its frame.
- At the abnormal level, there is a collision and the level of the energy is twice the normal level.
- A station that has a frame to send or is sending a frame needs to monitor the energy level to determine if the channel is idle, busy, or in collision mode.

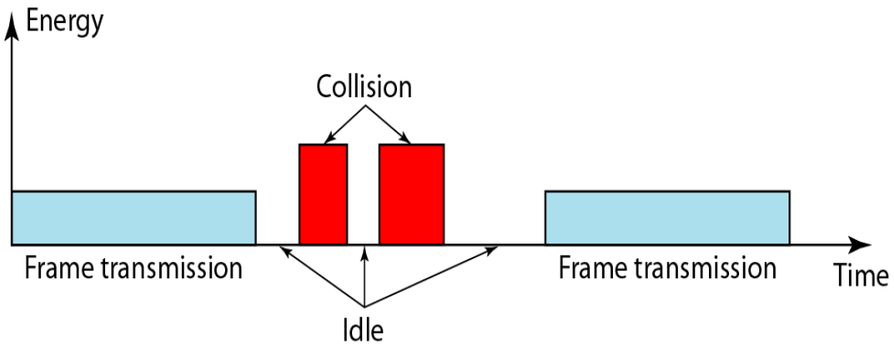


Figure: Energy level during transmission, idleness, or collision

Throughput:

- The throughput of CSMA/CD is greater than that of pure or slotted ALOHA.
- The maximum throughput occurs at a different value of G and is based on the persistence method and the value of p in the p-persistent approach.
- For 1-persistent method the maximum throughput is around 50 percent when G =1.
- For nonpersistent method, the maximum throughput can go up to 90 percent when G is between 3 and 8.

CARRIER SENSE MULTIPLE ACCESS WITH COLLISION AVOIDANCE (CSMA/CA):

- The basic idea behind CSMA/CD is that a station needs to be able to receive while transmitting to detect a collision.
- When there is **no collision**, the station receives **one signal: its own signal**.
- When there is a **collision**, the station receives two signals: its own signal and the signal transmitted by a second station.
- To distinguish between these two cases, the received signals in these two cases must be significantly different.
- In other words, the signal from the second station needs to add a significant amount of energy to the one created by the first station.
- In a **wired network**, the received signal has almost the same energy as the sent signal because either the length of the cable is short or there are repeaters that amplify the energy between the sender and the receiver.
- This means that in a collision, the detected energy almost doubles.
- However, **in a wireless network**, much of the sent energy is lost in transmission.
- The received signal has very little energy. Therefore, a collision may add only 5 to 10 percent additional energy. This is not useful for effective collision detection.
- We need to avoid collisions on wireless networks because they cannot be detected.
- Carrier sense multiple access with collision avoidance (CSMA/CA) was invented for this network.
- Collisions are avoided through the use of CSMA/CA's **three strategies: the inter frame space, the contention window, and acknowledgments.**

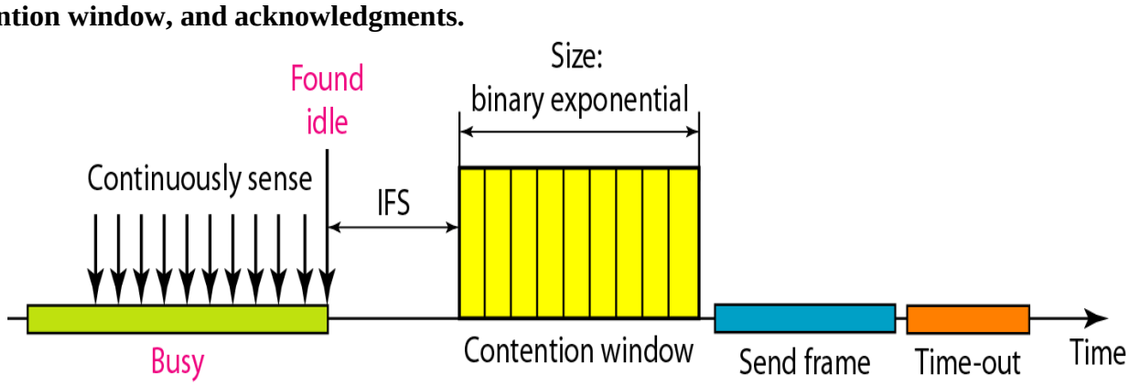


Figure: Timing in CSMA/CA

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.

- If after the IFS time the channel is still idle, the station can send, but it still needs to wait a time equal to the contention time.
- The IFS variable can also be used to prioritize stations or frame types.
- For example, a station that is assigned 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 back-off 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.

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.

Procedure:

- Note that the channel needs to be sensed before and after the IFS.
- The channel also needs to be sensed during the contention time.
- For each time slot of the contention window, the channel is sensed.
- If it is found idle, the timer continues; if the channel is found busy, the timer is stopped and continues after the timer becomes idle again.

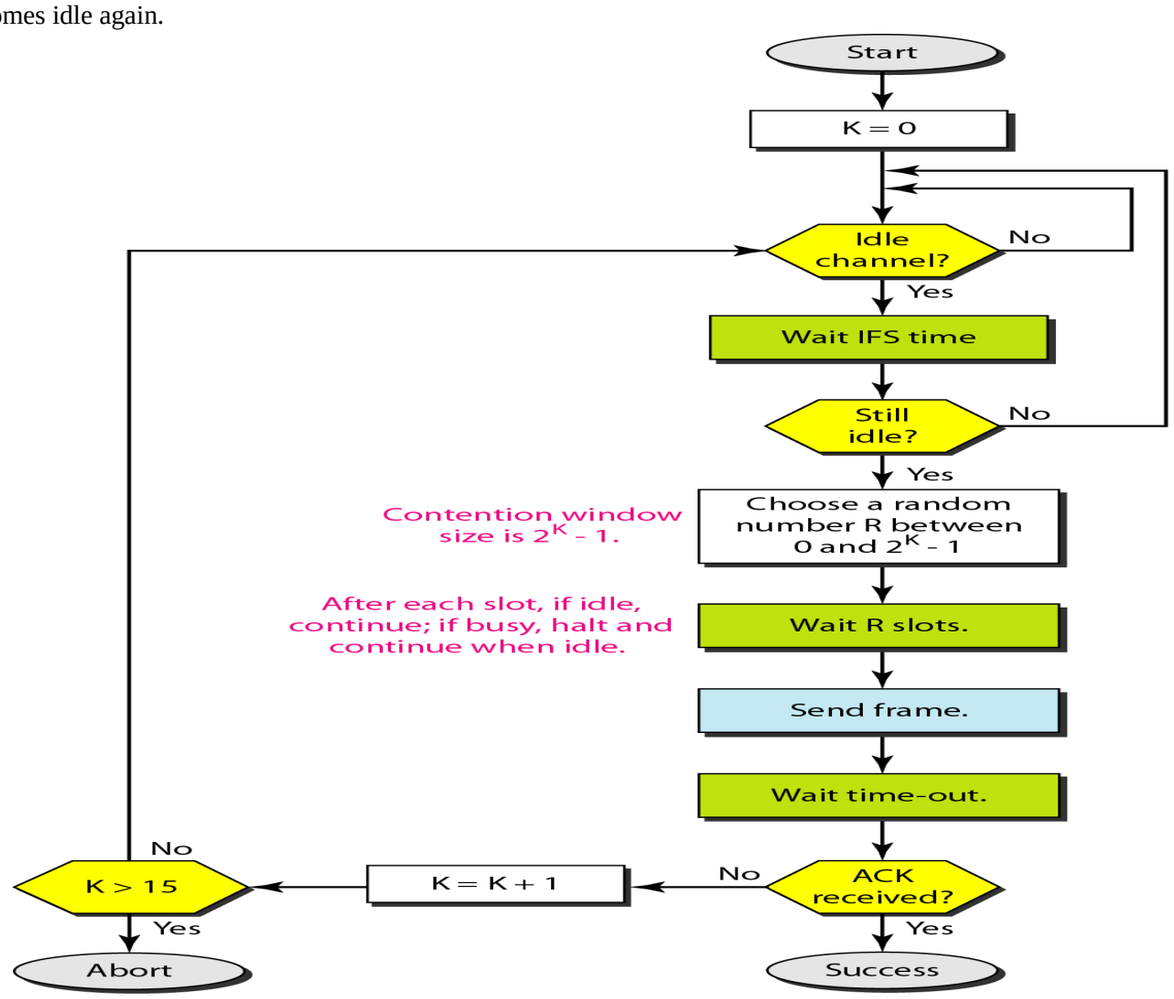


Figure: Flow diagram for CSMA/CA

CONTROLLED ACCESS PROTOCOLS:

- In controlled access, the stations consult one another to find which station has the right to send.
- A station cannot send unless it has been authorized by other stations.
- There are three popular controlled-access methods.
    1. Reservation

2. Polling
3. Token Passing

RESERVATION:

- In the reservation method, a station needs to make a reservation before sending data.
- Time is divided into intervals.
- In each interval, a reservation frame precedes the data frames sent in that interval.
- If there are N stations in the system, there are exactly N reservation minislots in the reservation frame.
- Each minislot belongs to a station. When a station needs to send a data frame, it makes a reservation in its own minislot.
- The stations that have made reservations can send their data frames after the reservation frame.
- Figure shows a situation with five stations and a five-minislot reservation frame.
- In the first interval, only stations 1, 3, and 4 have made reservations.
- In the second interval, only station 1 has made a reservation.

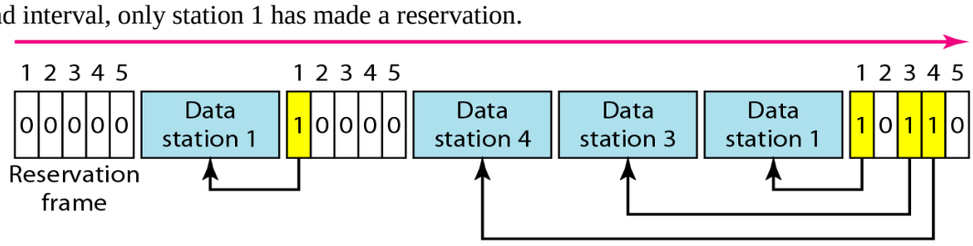


Figure: Reservation access method

POLLING:

- Polling works with topologies in which one device is designated as a primary station and the other devices are secondary stations.
- All data exchanges must be made through the primary device even when the ultimate destination is a secondary device.
- The primary device controls the link; the secondary devices follow its instructions.
- It is up to the primary device to determine which device is allowed to use the channel at a given time.
- The primary device, therefore, is always the initiator of a session.
- If the primary wants to **receive data**, it asks the secondaries if they have anything to send; this is called **poll function**.
- If the primary wants to **send data**, it tells the secondary to get ready to receive; this is called **select function**.

Select:

- The select function is used whenever the primary device has something to send.
- If it has something to send, the primary device sends it.
- What it does not know, however, is whether the target device is prepared to receive.
- So the primary must alert the secondary to the upcoming transmission and wait for an acknowledgment of the secondary's ready status.
- Before sending data, the primary creates and transmits a select (SEL) frame, one field of which includes the address of the intended secondary.

Poll:

- The poll function is used by the primary device to solicit transmissions from the secondary devices.
- When the primary is ready to receive data, it must ask (poll) each device in turn if it has anything to send.
- When the first secondary is approached, it responds either with a **NAK** frame if it has **nothing to send** or with data (in the form of a data frame) if it does.
- If the response is negative (a NAK frame), then the primary polls the next secondary in the same manner until it finds one with data to send.
- When the response is positive (a data frame), the primary reads the frame and returns an acknowledgment (ACK frame), verifying its receipt.

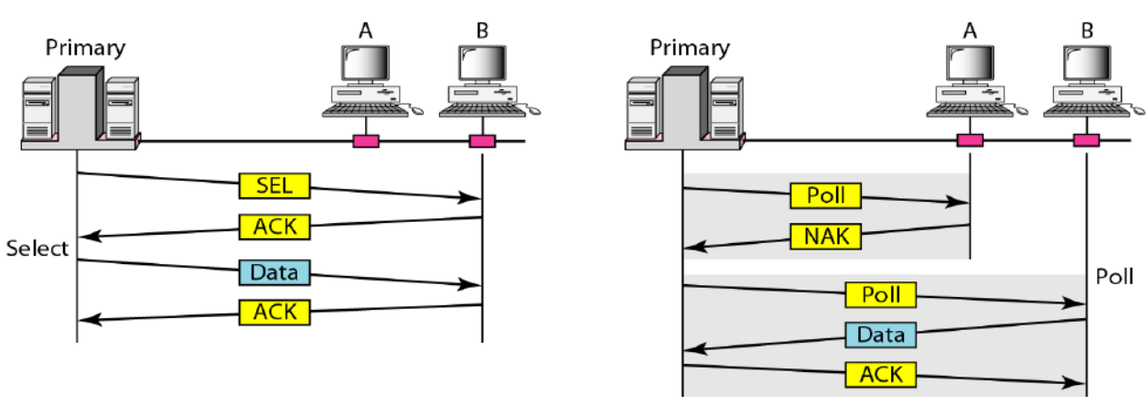


Figure: Select and poll functions in polling access method

TOKEN PASSING:

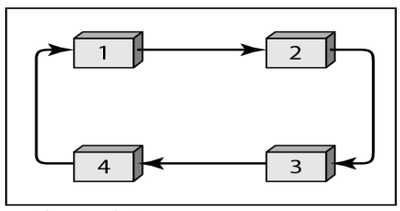
- In the token-passing method, the stations in a network are organized in a **logical ring**.
- In other words, for each station, there is a ***predecessor*** *and a* ***successor***.
- *The predecessor* is the station which is logically before the station in the ring;
- The *successor* is the station which is after the station in the ring.
- The current station is the one that is accessing the channel now.
- The right to this access has been passed from the predecessor to the current station.
- The right will be passed to the successor when the current station has no more data to send.
- But how is the right to access the channel passed from one station to another?
- In this method, a special packet called a **token** circulates through the ring.
- The possession of the token gives the station the right to access the channel and send its data.
- When a station has some data to send, it waits until it receives the token from its predecessor.
- It then holds the token and sends its data.
- When the station has no more data to send, it releases the token, passing it to the next logical station in the ring.
- The station cannot send data until it receives the token again in the next round.
- In this process, when a station receives the token and has no data to send, it just passes the data to the next station.
- **Token management** is needed for this access method.
- Stations must be limited in the time they can have possession of the token.
- The token must be monitored to ensure it has not been lost or destroyed.
- For example, if a station that is holding the token fails, the token will disappear from the network.
- Another function of token management is to assign priorities to the stations and to the types of data being transmitted.
- And finally, token management is needed to make low-priority stations release the token to high priority stations.

Logical Ring:

- In a token-passing network, stations do not have to be physically connected in a ring; the ring can be a logical one.

Physical ring topology:

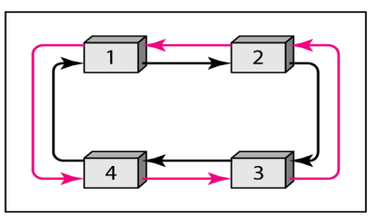
- In the physical ring topology, when a station sends the token to its successor, the token cannot be seen by other stations; the successor is the next one in line.
- This means that the token does not have to have the address of the next successor.
- The problem with this topology is that if one of the links-the medium between two adjacent stations fails, the whole system fails.



a. Physical ring

Dual ring topology:

- The dual ring topology uses a second (auxiliary) ring which operates in the reverse direction compared with the main ring.
- The second ring is for emergencies only (such as a spare tire for a car).
- If one of the links in the main ring fails, the system automatically combines the two rings to form a temporary ring.
- After the failed link is restored, the auxiliary ring becomes idle again.
- Note that for this topology to work, each station needs to have two transmitter ports and two receiver ports.
- The high-speed Token Ring networks called FDDI (Fiber Distributed Data Interface) and CDDI (Copper Distributed Data Interface) use this topology.

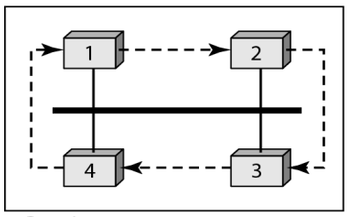


b. Dual ring

Bus ring topology:

- In the bus ring topology, also called a token bus, the stations are connected to a single cable called a bus.
- They, however, make a logical ring, because each station knows the address of its successor (and also predecessor for token management purposes).
- When a station has finished sending its data, it releases the token and inserts the address of its successor in the token.

- Only the station with the address matching the destination address of the token gets the token to access the shared media.
- The Token Bus LAN, standardized by IEEE, uses this topology.



c. Bus ring

Star ring topology:

- In a star ring topology, the physical topology is a star.
- There is a hub, however, that acts as the connector.
- The wiring inside the hub makes the ring; the stations are connected to this ring through the two wire connections.
- This topology makes the network less prone to failure because if a link goes down, it will be bypassed by the hub and the rest of the stations can operate.
- Also adding and removing stations from the ring is easier.
- This topology is still used in the Token Ring LAN designed by IBM.

CHANNELIZATION:

- Channelization is a multiple-access method in which the available bandwidth of a link is shared in time, frequency, or through code, between different stations.
- Three most commonly used multiple access methods are-
    1. Frequency Division Multiple Access (FDMA)
    2. Time Division Multiple Access (TDMA)
    3. Code Division Multiple Access (CDMA)

FREQUENCY-DIVISION MULTIPLE ACCESS (FDMA):

- In frequency-division multiple access (FDMA), the available bandwidth is divided into frequency bands.
- Each station is allocated a band to send its data. In other words, each band is reserved for a specific station, and it belongs to the station all the time.
- Each station also uses a bandpass filter to confine the transmitter frequencies.
- To prevent station interferences, the allocated bands are separated from one another by small *guard bands.*
- FDMA specifies a predetermined frequency band for the entire period of communication.
- This means that stream data (a continuous flow of data that may not be packetized) can easily be used with FDMA.

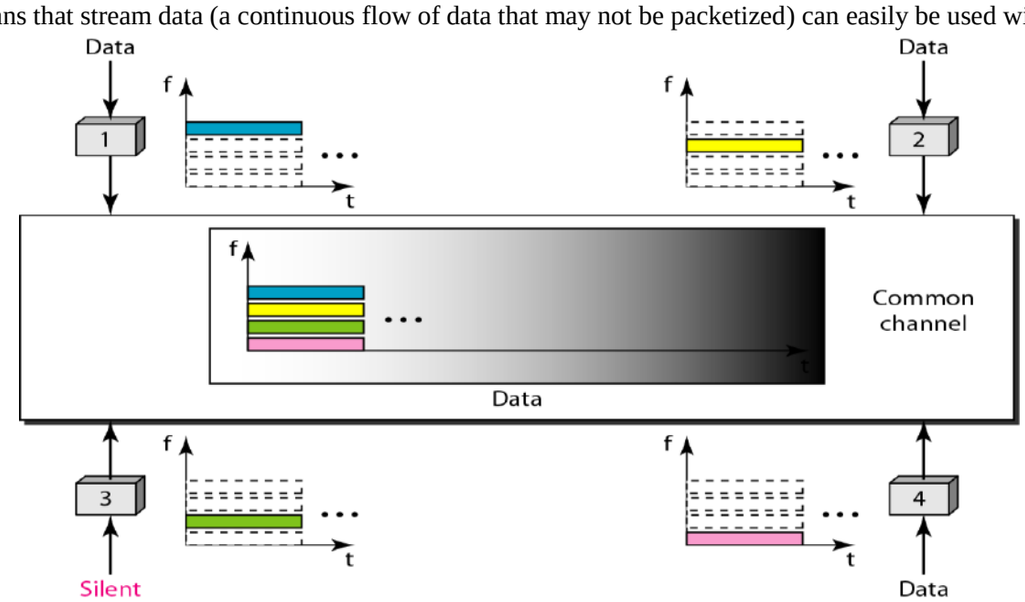


Figure: Frequency-division multiple access (FDMA)

TIME-DIVISION MULTIPLE ACCESS (TDMA):

- In time-division multiple access (TDMA), the stations share the bandwidth of the channel in time.
- Each station is allocated a time slot during which it can send data.
- Each station transmits its data in is assigned time slot.
- The main problem with TDMA lies in achieving synchronization between the different stations.
- Each station needs to know the beginning of its slot and the location of its slot.
- This may be difficult because of propagation delays introduced in the system if the stations are spread over a large area.

- To compensate for the delays, we can insert **guard times**.
- Synchronization is normally accomplished by having some synchronization bits at the beginning of each slot.

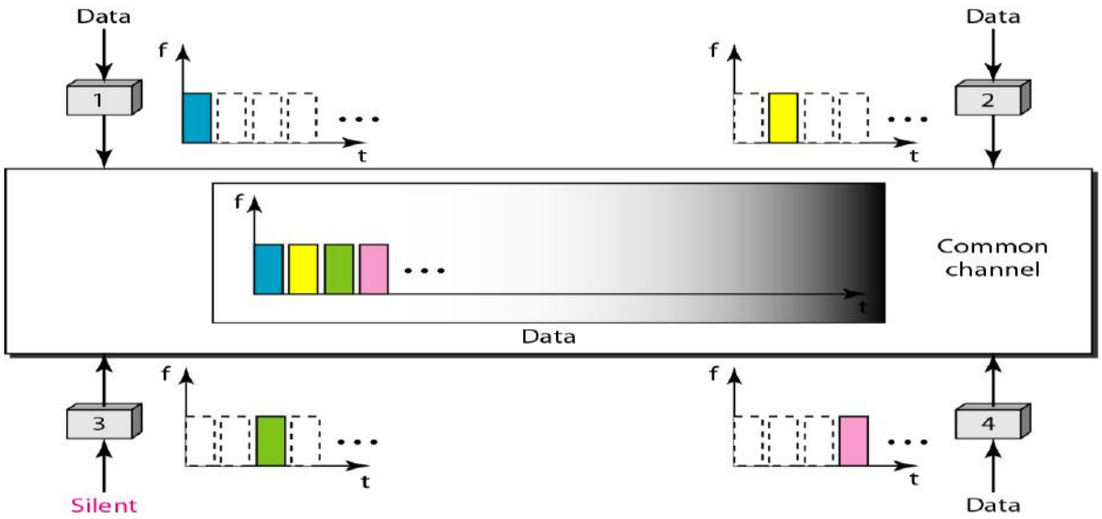


Figure: Time-division multiple access (TDMA)

CODE-DIVISION MULTIPLE ACCESS (CDMA):

- In CDMA each station transmitter may transmit whenever it requires and can use **entire bandwidth** i.e. there are no restrictions on time and bandwidth.
- CDMA is also called as **Spread Spectrum Multiple Access** because transmission can spread throughout the bandwidth.
- Each station is assigned a unique binary code; this code is called as **chip code**.
- Each station and transmission is identified by its chip code.
- The receiver uses chip code to recover the signal from desired station.

Applications of CDMA:

- CDMA is used for wireless system with fixed base station and many mobile stations at varying distance from it.
- CDMA is used in satellite systems so that many signals can use a transponder making it more efficient.
- CDMA is used in digital cellular telephone services because it permits more users to occupy a given band.
- Wideband CDMA is used for digital cell phone systems to accommodate voice transmission along with high speed data, FAX and internet communication.
- CDMA is ideally suited for military application because of immunity to noise.

Comparison of Virtual-Circuit and Datagram Subnets

Issue	Datagram subnet	Virtual-circuit subnet
Circuit setup	Not needed	Required
Addressing	Each packet contains the full source and destination address	Each packet contains a short VC number
State information	Routers do not hold state information about connections	Each VC requires router table space per connection
Routing	Each packet is routed independently	Route chosen when VC is set up; all packets follow it
Effect of router failures	None, except for packets lost during the crash	All VCs that passed through the failed router are terminated
Quality of service	Difficult	Easy if enough resources can be allocated in advance for each VC
Congestion control	Difficult	Easy if enough resources can be allocated in advance for each VC

Routing Algorithms

The routing algorithm is that part of the network layer software responsible for deciding which output line an incoming packet should be transmitted on. If the subnet uses datagrams internally, this decision must be made a new for every arriving data packet. If the subnet uses virtual circuits internally, routing decisions are made only when a new virtual circuit is being set up. Main properties of a routing algorithm are:

Correctness: it avoids data errors

Simplicity: Easy to understand and implement.
Robustness: It should be able to cope with changes in the topology and traffic.
Stability : It reaches equilibrium and stays there.
Fairness : Minimizing mean packet delay.
Optimality : . Maximizing total network throughput.

The Optimality Principle

Optimality principle states that if router J is on the optimal path from router I to router K, then the optimal path from J to K also falls along the same route.

2.1 Shortest Path Routing

The idea is to build a graph of the subnet, with each node of the graph representing a router and each arc of the graph representing a communication line. To choose a route between a given pair of routers, the algorithm just finds the shortest path between them on the graph.

Each node is labeled (in parentheses) with its distance from the source node along the best known path. Initially, no paths are known, so all nodes are labeled with infinity. As the algorithm proceeds and paths are found, the labels may change, reflecting better paths. A label may be either tentative or permanent. Initially, all abels are tentative. When it is discovered that a label represents the shortest possible path from the source to that node, it is made permanent and never changed thereafter.

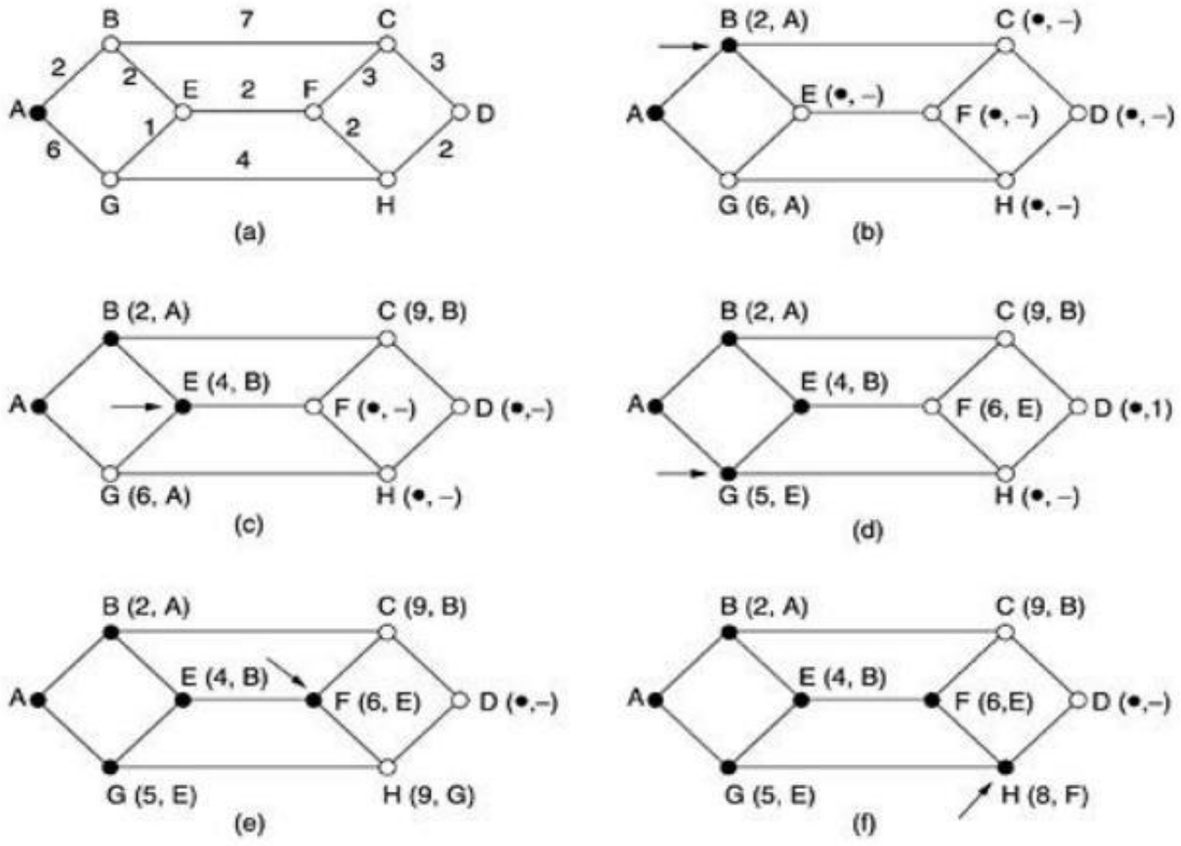


Fig.7 The first five steps used in computing the shortest path from A to D. The arrows indicate the working node.

To illustrate how the labeling algorithm works, look at the weighted, undirected graph of Fig.7(a), where the weights represent, for example, distance. We want to find the shortest path from A to D. Mark node A as permanent, indicated by a filled-in circle. Then examine, in turn, each of the nodes adjacent to A (the working node), relabeling each one with the distance to A.

Whenever a node is relabeled, label it with the node from which the probe was made so that one can reconstruct the final path later. Having examined each of the nodes adjacent to A, examine all the tentatively labeled nodes in the whole graph and make the one with the smallest label permanent, as shown in Fig.7(b). This becomes the new working node.

Now start at B and examine all nodes adjacent to it. If the sum of the label on B and the distance from B to the node being considered is less than the label on that node, is the shorter path, so the node is relabeled. After all the nodes adjacent to the working node have been inspected and the tentative labels changed if possible, the entire graph is searched for the tentatively-labeled node with the smallest value. This node is made permanent and becomes the working node for the next round. Fig.7 shows the first five steps of the algorithm.

2.2 Flooding

Flooding is a static routing algorithm, in which every incoming packet is sent out on every outgoing line except the one it arrived on. Flooding obviously generates vast numbers of duplicate packets. The following measures are taken to damp the process.

One such measure is to have a hop counter contained in the header of each packet, which is decremented at each hop, with the packet being discarded when the counter reaches zero. Ideally, the hop counter should be initialized to the length of the path from source to destination.

An alternative technique for damming the flood is to keep track of which packets have been flooded, to avoid sending them out a second time. Achieve this goal is to have the source router put a sequence number in each packet it receives from its hosts. Each router then needs a list per source router telling which sequence numbers originating at that source have already been seen. If an incoming packet is on the list, it is not flooded.

A variation of flooding that is slightly more practical is selective flooding. In this algorithm the routers do not send every incoming packet out on every line, only on those lines that are going approximately in the right direction.

Flooding is used in the following applicatons.

Military applications : where robustness of flooding is highly desirable.

Distributed database applications : to update all the databases concurrently.

Wireless networks : all messages transmitted by a station can be received by all other stations within its radio range,.

Flooding always chooses the shortest path because it chooses every possible path in parallel.

2.3 Distance Vector Routing

The distance vector routing algorithm is sometimes called Bellman-Ford routing algorithm and the Ford-Fulkerson algorithm; It was the original ARPANET routing algorithm and was also used in the Internet under the name RIP.

In distance vector routing, each router maintains a routing table indexed by, and containing one entry for, each router in the subnet. This entry contains two parts: the preferred outgoing line to use for that destination and an estimate of the time or distance to that destination. The metric used might be number of hops, time delay in milliseconds, total number of packets queued along the path, or something similar.

The router is assumed to know the "distance" to each of its neighbors. If the metric is delay, the router can measure it directly with special ECHO packets that the receiver just timestamps and sends back as fast as it can.

As an example, assume that delay is used as a metric and that the router knows the delay to each of its neighbors. Once every T msec each router sends to each neighbor a list of its estimated delays to each destination. It also receives a similar list from each neighbor. Imagine that one of these tables has just come in from neighbor X, with Xi being X's estimate of how long it takes to get to router i. If the router knows that the delay to X is m msec, it also knows that it can reach router i via X in Xi + m msec. By performing this calculation for each neighbor, a router can find out which estimate seems the best and use that estimate and the corresponding line in its new routing table. Note that the old routing table is not used in the calculation.

This updating process is illustrated in Fig.9 Part (a) shows a subnet. The first four columns of part (b) show the delay vectors received from the neighbors of router J. A claims to have a 12-msec delay to B, a 25-msec delay to C, a 40-msec delay to D, etc. Suppose that J has measured or estimated its delay to its neighbors, A, I, H, and K as 8, 10, 12, and 6 msec,

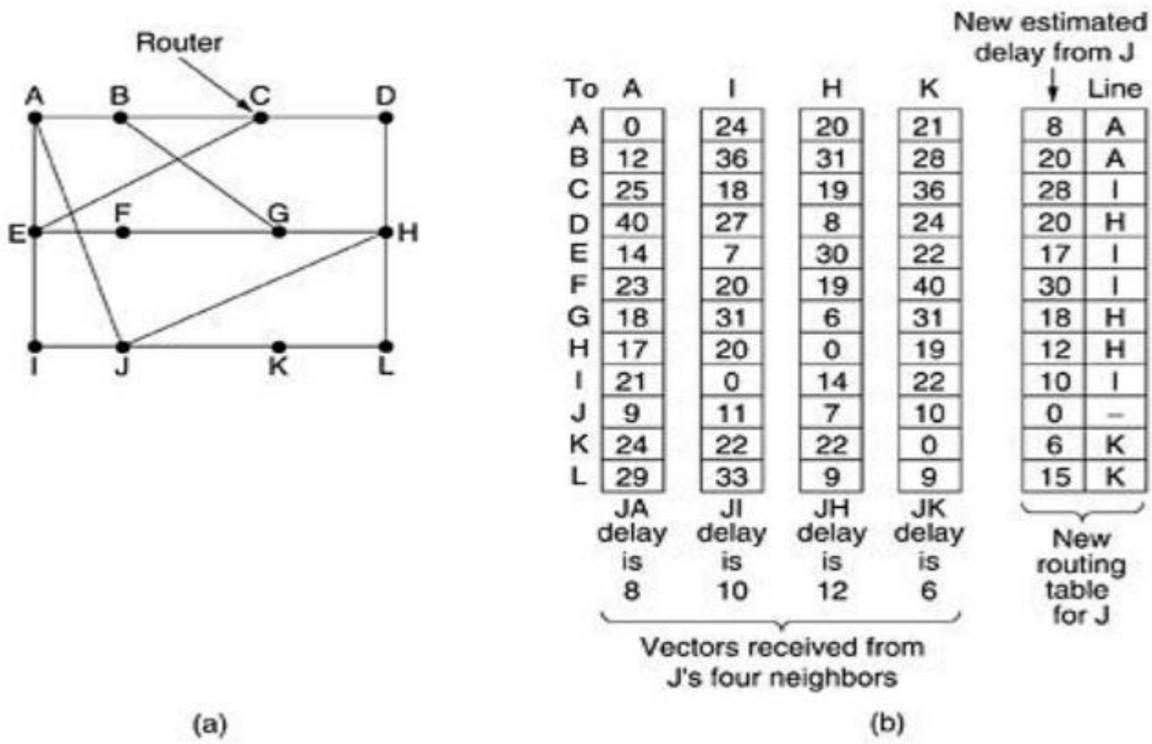


Fig.9 (a) A subnet. (b) Input from A, I, H, K, and the new routing table for J.

Consider how J computes its new route to router G. It knows that it can get to A in 8 msec, an A claims to be able to get to G in 18 msec, so J knows it can count on a delay of 26 msec to G it forwards packets bound for G to A. Similarly, it computes the delay to G via I, H, and K as 4 (31 + 10), 18 (6 + 12), and 37 (31 + 6) msec, respectively. The best of these values is 18, so it makes an entry in its routing table that the delay to G is 18 msec and that the route to use is via H. The same calculation is performed for all the other destinations, with the new routing table shown in the last column of the figure.

count-to-infinity problem

Distance vector routing works in theory but has a serious drawback in practice. To see how fast good news propagates, consider the five-node (linear) subnet of Fig.10, where the delay metric is the number of hops. Suppose A is down initially and all the other routers know this. In other words, they have all recorded the delay to A as infinity.

When A comes up, the other routers learn about it via the vector exchanges. At the time of the first exchange, B learns that its left neighbor has zero delay to A. B now makes an entry in its routing table that A is one hop away to the left. All the other routers still think that A is down. At this point, the routing table entries for A are as shown in the second row of Fig.10 (a). On the next exchange, C learns that B has a path of length 1 to A, so it updates its routing table to indicate a path of length 2, but D and E do not hear the good news until later. Clearly, the good news is spreading at the rate of one hop per exchange. In a subnet whose longest path is of length N hops, within N exchanges everyone will know about newly-revived lines and routers.

Now consider the situation of Fig.10 (b), in which all the lines and routers are initially up. Routers B, C, D, and E have distances to A of 1, 2, 3, and 4, respectively. Suddenly A goes down.

At the first packet exchange, B does not hear anything from A. Fortunately, C says: Do not worry; I have a path to A of length 2. Little does B know that C's path runs through B itself. For all B knows, C might have ten lines all with separate paths to A of length 2. As a result, B thinks it can reach A via C, with a path length of 3. D and E do not update their entries for A on the first exchange.

On the second exchange, C notices that each of its neighbors claims to have a path to A of length 3. It picks one of the them at random and makes its new distance to A 4, as shown in the third row of Fig. 10(b). Subsequent exchanges produce the history shown in the rest of Fig. 10(b).

If the metric is time delay, there is no well-defined upper bound, so a high value is needed to prevent a path with a long delay from being treated as down. This problem is known as the count-to-infinity problem. There have been a few attempts to solve it (such as split horizon with poisoned reverse in RFC 1058), but none of these work well in general. The core of the problem is that when X tells Y that it has a path somewhere, Y has no way of knowing whether it itself is on the path.

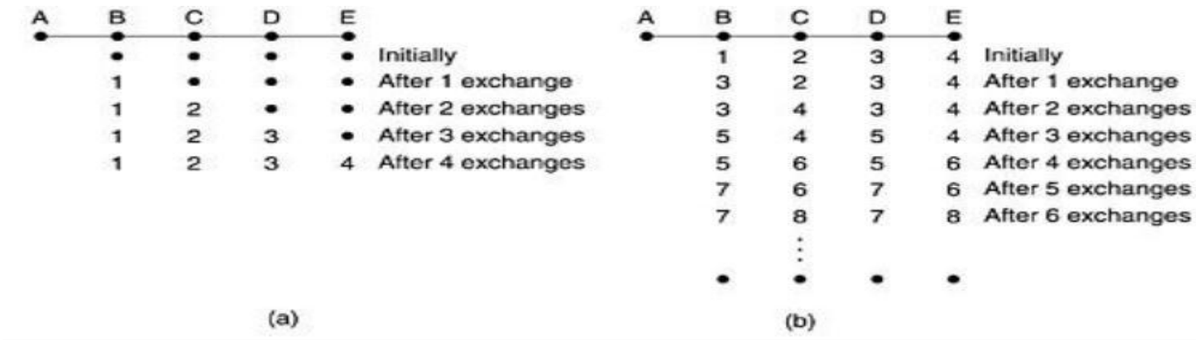


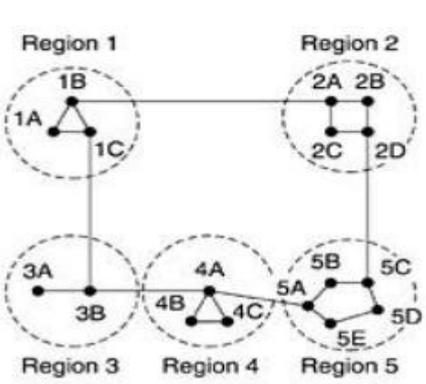
Fig.10: The count-to-infinity problem.

Hierarchical routing.

As networks grow in size, the router routing tables grow proportionally. Not only is router memory consumed by ever-increasing tables, but more CPU time is needed to scan them and more bandwidth is needed to send status reports about them. so the routing will have to be done hierarchically, as it is in the telephone network.

When hierarchical routing is used, the routers are divided into regions, with each router knowing all the details about how to route packets to destinations within its own region, but knowing nothing about the internal structure of other regions. When different networks are interconnected, it is natural to regard each one as a separate region in order to free the routers in one network from having to know the topological structure of the other ones.

For huge networks, a two-level hierarchy may be insufficient; it may be necessary to group the regions into clusters, the clusters into zones, the zones into groups, and so on. Fig.11 gives a quantitative example of routing in a two-level hierarchy with five regions. The full routing table for router 1A has 17 entries, as shown in Fig.11 (b). When routing is done hierarchically, as in Fig.11 (c), there are entries for all the local routers as before, but all other regions have been condensed into a single router, so all traffic for region 2 goes via the 1B -2A line, but the rest of the remote traffic goes via the 1C -3B line. Hierarchical routing has reduced the table from 17 to 7 entries. As the ratio of the number of regions to the number of routers per region grows, the savings in table space increase.



Full table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2A	1B	2
2B	1B	3
2C	1B	3
2D	1B	4
3A	1C	3
3B	1C	2
4A	1C	3
4B	1C	4
4C	1C	4
5A	1C	4
5B	1C	5
5C	1B	5
5D	1C	6
5E	1C	5

(b)

Hierarchical table for 1A

Dest.	Line	Hops
1A	–	–
1B	1B	1
1C	1C	1
2	1B	2
3	1C	2
4	1C	3
5	1C	4

(c)

Fig.11 Hierarchical routing.

For example, consider a subnet with 720 routers. If there is no hierarchy, each router needs 720 routing table entries.

If the subnet is partitioned into 24 regions of 30 routers each, each router needs 30 local entries plus 23 remote entries for a total of 53 entries. If a three-level hierarchy is chosen, with eight clusters, each containing 9 regions of 10 routers, each router needs 10 entries for local routers, 8 entries for routing to other regions within its own cluster, and 7 entries for distant clusters, for a total of 25 entries. Kamoun and Kleinrock (1979) discovered that the optimal number of levels for an N router subnet is ln N, requiring a total of e ln N entries per router. They have also shown that the increase in effective mean path length caused by hierarchical routing is sufficiently small that it is usually acceptable.

Broadcast Routing

Sending a packet to all destinations simultaneously is called broadcasting. In some applications, hosts need to send messages to many or all other hosts. For example, a service distributing weather reports, stock market updates, or live radio programs might work best by broadcasting to all machines and letting those that are interested read the data. Sending a packet to all destinations simultaneously is called broadcasting; various methods have been proposed for doing it.

One broadcasting method that requires no special features from the subnet is for the source to simply send a distinct packet to each destination. Not only is the method wasteful of bandwidth, but it also requires the source to have a complete list of all destinations.

Flooding is another obvious candidate. Although flooding is ill-suited for ordinary point -to- point communication, for broadcasting it might rate serious consideration, especially if none of the methods described below are applicable. The problem with flooding as a broadcast technique is the same problem it has as a point-to-point routing algorithm: it generates too many packets and consumes too much bandwidth.

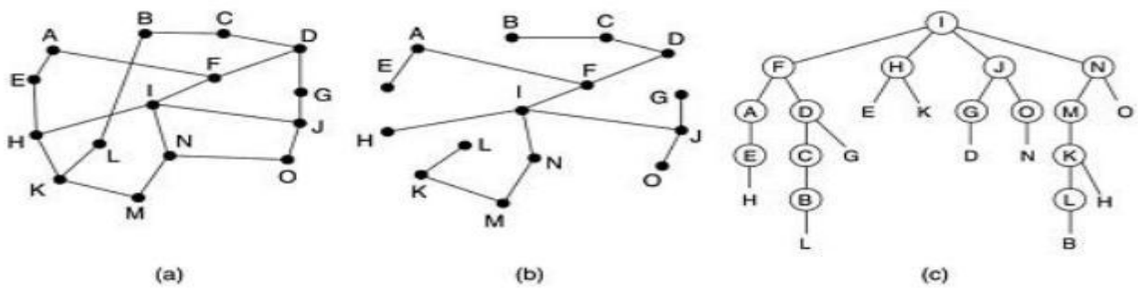
A third algorithm is multi destination routing. If this method is used, each packet contains either a list of destinations or a bit map indicating the desired destinations. When a packet arrives at a router, the router checks all the destinations to determine the set of output lines that will be needed. (An output line is needed if it is the best route to at least one of the destinations.) The router generates a new copy of the packet for each output line to be used and includes in each packet only those destinations that are to use the line. In effect, the destination set is partitioned among the output lines. After a sufficient number of hops, each packet will carry only one destination and can be treated as a normal packet. Multidestination routing is like separately addressed packets, except that when several packets must follow the same route, one of them pays full fare and the rest ride free.

A fourth broadcast algorithm makes explicit use of the sink tree for the router initiating the broadcast—or any other convenient spanning tree for that matter. A spanning tree is a subset of the subnet that includes all the routers but contains no loops. If each router knows which of its lines belong to the spanning tree, it can copy an incoming broadcast packet onto all the spanning tree lines except the one it arrived on. This method makes excellent use of bandwidth, generating the absolute minimum number of packets necessary to do the job. The only problem is that each router must have knowledge of some spanning tree for the method to be applicable. Sometimes this information is available (e.g., with link state routing) but sometimes it is not (e.g., with distance vector routing).

reverse path forwarding

The last broadcast algorithm is an attempt to approximate the behavior of the previous one, even when the routers do not know anything at all about spanning trees. The idea, called reverse path forwarding, is remarkably simple once it has been pointed out. When a broadcast packet arrives at a router, the router checks to see if the packet arrived on the line that is normally used for sending packets to the source of the broadcast.

If so, there is an excellent chance that the broadcast packet itself followed the best route from the router and is therefore the first copy to arrive at the router. This being the case, the router forwards copies of it onto all lines except the one it arrived on. If, however, the broadcast packet arrived on a line other than the preferred one for reaching the source, the packet is discarded as a likely duplicate.



Reverse path forwarding. (a) A subnet.(b) A sink tree.(c) The tree built by reverse path forwarding.

An example of reverse path forwarding is shown in Fig.12. Part (a) shows a subnet, part (b) shows a sink tree for router I of that subnet, and part (c) shows how the reverse path algorithm works. On the first hop, I sends packets to F, H, J, and N, as indicated by the second row of the tree. Each of these packets arrives on the preferred path to I (assuming that the preferred path falls along the sink tree) and is so indicated by a circle around the letter. On the
second hop, eight packets are generated, two by each of the routers that received a packet on the first hop.

As it turns out, all eight of these arrive at previously unvisited routers, and five of these arrive along the preferred line. Of the six packets generated on the third hop, only three arrive on the preferred path (at C, E, and K); the others are duplicates. After five hops and 24 packets, the broadcasting terminates, compared with four hops and 14 packets had the sink tree been followed exactly.

The principal advantage of reverse path forwarding is that it is both reasonably efficient and easy to implement. It does not require routers to know about spanning trees, nor does it have the overhead of a destination list or bit map in each broadcast packet as does multi destination addressing. Nor does it require any special mechanism to stop the process, as flooding does (either a hop counter in each packet and a priori knowledge of the subnet diameter, or a list of packets already seen per source).

Multicast routing algorithm

Sending a message to multiple receivers with a single send operation is called multicasting. Some applications require that widely-separated processes work together in groups, for example, a group of processes implementing a distributed database system. In these situations, it is frequently necessary for one process to send a message to all the other members of the group. If the group is small, it can just send each other member a point-to-point message.

Multicasting requires group management. Some way is needed to create and destroy groups, and to allow processes to join and leave groups. It is important that routers know which of their hosts belong to which groups. Either hosts must inform their routers about changes in group membership, or routers must query their hosts periodically.

To do multicast routing, each router computes a spanning tree covering all other routers. For example, in Fig. 14(a) there are two groups, 1 and 2. Some routers are attached to hosts that belong to one or both of these groups, as indicated in the figure. A spanning tree for the leftmost router is shown in Fig. 14(b).

When a process sends a multicast packet to a group, the first router examines its spanning tree and prunes it, removing all lines that do not lead to hosts that are members of the group. In our example, Fig. 14(c) shows the pruned spanning tree for group 1. Similarly, Fig. 14(d) shows the pruned spanning tree for group 2. Multicast packets are forwarded only along the appropriate spanning tree.

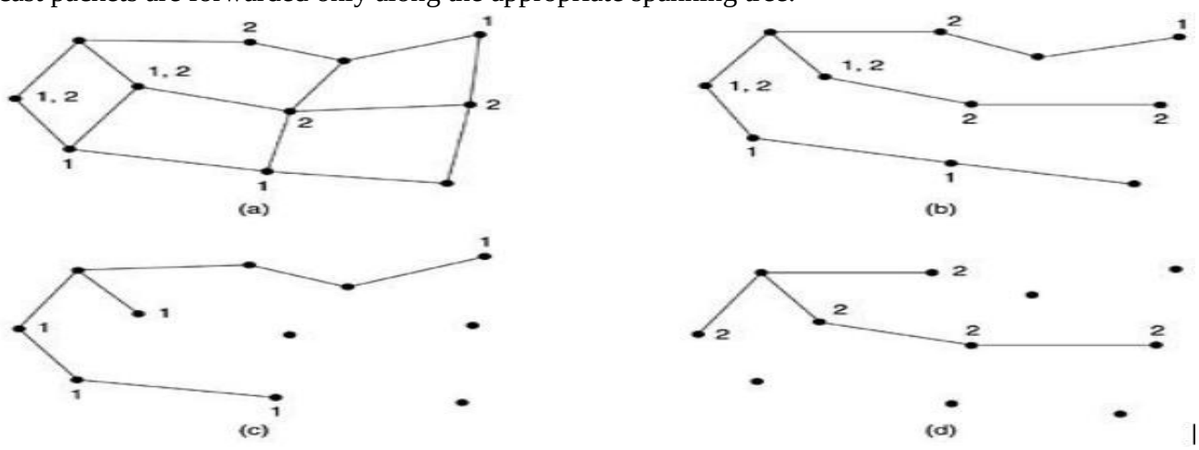


Fig.14 (a) A network. (b) A spanning tree for the leftmost router. (c) A multicast tree for group 1. (d) A multicast tree for group 2.

Various ways of pruning the spanning tree are possible. The simplest one can be used if link state routing is used and each router is aware of the complete topology, including which hosts belong to which groups. Then the spanning tree can be pruned, starting at the end of each path, working toward the root, and removing all routers that do not belong to the group in question.

One potential disadvantage of this algorithm is that it scales poorly to large networks. Suppose that a network has n groups, each with an average of m members. For each group, m pruned spanning trees must be stored, for a total of mn trees