Ques 01. (a) Differentiate between Parallel and Serial Communication. Give an example of each.

Ans: Difference Between Parallel and Serial Communication

1. Definition

Parallel Communication: In parallel communication, multiple bits of data are transmitted simultaneously using multiple channels or wires. Each wire carries one bit, allowing the transmission of several bits at the same time.

Serial Communication: In serial communication, data is transmitted sequentially, one bit at a time, over a single channel or wire. This reduces the number of wires required but results in slower transmission compared to parallel communication.

2. Speed

Parallel Communication: It is faster because multiple bits are sent simultaneously across different wires, allowing for higher data throughput.

Serial Communication: It is relatively slower as only one bit is transmitted at a time. However, improvements in technology have increased serial communication speeds in recent years.

3. Distance

Parallel Communication: It is suitable for short distances because over long distances, the signals on different wires can become unsynchronized, leading to errors.

Serial Communication: It works better over long distances since it avoids the problem of signal misalignment by transmitting data bit by bit.

4. Complexity

Parallel Communication: It requires more wires or channels, making the system more complex and expensive to implement, especially over longer distances.

Serial Communication: It is simpler and cheaper to implement because fewer wires are needed. This makes serial communication more common in modern applications.

5. Example

Parallel Communication: An example of parallel communication is the connection between a computer and an older printer using a parallel port (such as a Centronics port), where 8 bits of data are sent simultaneously over 8 wires.

Serial Communication: A common example of serial communication is the USB (Universal Serial Bus), which transmits data serially between devices like a computer and peripherals such as a mouse, keyboard, or external hard drive.

(b) Compare POP and IMAP.

Ans: **Comparison Between POP and IMAP**

1. **Definition**

- **POP (Post Office Protocol)**: POP is a simple protocol that allows an email client to download messages from a mail server to a local computer. Once the messages are downloaded, they are typically deleted from the server, although some settings allow keeping a copy on the server.

- **IMAP (Internet Message Access Protocol)**: IMAP allows an email client to access and manage emails directly on the mail server. Messages remain on the server, and the user can view, organize, and manage emails from multiple devices without needing to download them permanently.

2. Email Storage

- **POP**: Emails are downloaded to the user's device, and once downloaded, they are generally deleted from the server. This makes POP suitable for users who primarily access email from one device.

- **IMAP**: Emails are stored on the server and remain there, allowing users to access, manage, and sync emails across multiple devices.

Changes made (like reading, deleting, or organizing emails) are reflected on all devices.

3. Device Synchronization

- **POP**: POP does not sync between devices. Once emails are downloaded to one device, they are not accessible from another device unless configured to leave a copy on the server (which can lead to inboxes getting out of sync).

- **IMAP**: IMAP is designed for synchronization. Emails and folders are kept in sync across all devices accessing the same account, ensuring a consistent view of your inbox.

4. Server Usage

- **POP**: Since emails are typically downloaded and deleted from the server, less storage is needed on the server. However, this also means that once emails are downloaded, they cannot be accessed from other devices unless manually left on the server.

- **IMAP**: Emails remain on the server until manually deleted. This requires more server storage, but offers the advantage of access from any device, making it ideal for users who frequently switch between devices.

5. Offline Access

- **POP**: POP is ideal for offline access. Once emails are downloaded, users can read and manage them offline, as they are stored locally on the device.

- **IMAP**: IMAP generally requires an internet connection to view emails since they are stored on the server. However, most email clients allow local caching of messages for offline access.

6. Use Case

- **POP**: Best suited for users who access email from one device and do not need to sync emails between multiple devices. POP is also beneficial for users with limited server storage.

- **IMAP**: Ideal for users who access their email from multiple devices (e.g., phone, tablet, computer) and need real-time synchronization across devices. IMAP is widely used in modern email setups because of its flexibility.

7. Example

- **POP**: An example of POP usage is an old email client that downloads all messages to a personal computer and removes them from the server (e.g., older versions of Outlook with POP configuration).

- **IMAP**: An example of IMAP is using modern email services like Gmail, where emails are accessible and synced across devices (e.g., phone, laptop, webmail).

Ques 02. (a) What is Ad hoc Wireless Communication System? Explain.

Ans: An **Ad Hoc Wireless Communication System** is a decentralized type of wireless network where devices (or nodes) communicate directly with each other without relying on any pre-existing infrastructure, such as routers, access points, or centralized servers. This system is self-configuring, and each device in the network participates in forwarding data for other devices, creating a dynamic and flexible network structure.

Key Characteristics:

1. **Decentralized Network**
    - There is no central control or infrastructure, unlike traditional networks (e.g., Wi-Fi), which require routers or access points to manage communication. In an ad hoc network, nodes can communicate with each other directly or through intermediate nodes if they are not within direct range.

2. **Dynamic Topology**
    - The network's structure is constantly changing as devices (or nodes) can move, join, or leave the network at any time. This makes ad hoc networks highly flexible and adaptable, especially in environments where the network must constantly reconfigure itself.

3. **Peer-to-Peer Communication**
   - Devices communicate in a peer-to-peer manner, meaning there is no need for a fixed infrastructure to send or receive data. Each device acts both as a client and a router, forwarding packets for others when necessary.

4. **Multi-hop Routing**
   - In many cases, devices may not be within the direct communication range of each other. In such scenarios, multi-hop routing is used, where data is passed through intermediate nodes to reach its destination.

Applications of Ad Hoc Wireless Networks:

1. **Military Communication**
   - Ad hoc networks are often used in military operations where infrastructure is either unavailable or impractical. They enable soldiers or vehicles to communicate in remote locations, providing a robust communication system in the field.

2. **Disaster Recovery and Emergency Situations**
   - In natural disasters or emergencies, where traditional communication infrastructure might be damaged or inaccessible, ad hoc networks can be quickly set up for rescue teams and emergency personnel to coordinate their efforts.

3. **Vehicular Ad Hoc Networks (VANETs)**
   - These networks are designed for vehicles to communicate with each other, enabling real-time traffic updates, safety alerts, and autonomous driving functionalities.

4. **Sensor Networks**
   - In environmental monitoring or IoT applications, ad hoc networks are used to interconnect sensor devices spread over a large area, allowing them to collect and transmit data without relying on fixed infrastructure.

Advantages:

1. **Flexibility and Mobility**
   - Ad hoc networks are highly mobile and can be set up anywhere, without the need for a fixed infrastructure, making them ideal for temporary or emergency use.

2. **Scalability**
   - These networks can easily scale, as new nodes can join the network at any time without the need for reconfiguration.

3. **Cost-Effective**
   - Since no centralized infrastructure is needed, ad hoc networks can be more cost-effective to deploy, especially in situations where building infrastructure is difficult or expensive.

Challenges:

1. **Security Issues**
   - Ad hoc networks are more vulnerable to security threats like eavesdropping, man-in-the-middle attacks, and denial of service (DoS) attacks due to the lack of centralized control.

2. **Limited Resources**
   - Devices in ad hoc networks typically have limited battery power and processing capacity, which can affect the network's performance over time.

3. **Routing Complexity**
   - The constantly changing network topology can make routing data between nodes more complex, requiring efficient routing protocols to ensure reliable communication.

(b) What is better for computer communication ---- Analog or digital? Justify your answer.

Ans: When comparing analog and digital communication for computer systems, **digital communication** is generally considered better. Here's why:

Advantages of Digital Communication

1. **Accuracy and Reliability**:
   - **Digital**: Digital communication is more accurate because it transmits data as discrete signals (binary code), which are less susceptible to noise and interference. Digital signals can be regenerated at intermediate points to maintain signal integrity,

leading to more reliable data transmission.

    - **Analog**: Analog communication transmits data as continuous signals, which are more prone to noise and distortion. Even small amounts of interference can degrade the quality of the signal, leading to potential data loss or errors.

2. **Error Detection and Correction**:
    - **Digital**: Digital systems employ various error detection and correction techniques (e.g., checksums, parity bits, error-correcting codes) to identify and correct errors that occur during transmission. This ensures data integrity and reduces the need for retransmissions.

    - **Analog**: Analog systems lack robust error detection and correction mechanisms. Errors due to noise or interference are more challenging to detect and correct, leading to potential inaccuracies in the transmitted data.

3. **Data Compression**:
    - **Digital**: Digital communication allows for efficient data compression techniques (e.g., Huffman coding, run-length encoding) that reduce the amount of data that needs to be transmitted. This improves bandwidth utilization and reduces transmission costs.

    - **Analog**: Data compression is more complex in analog systems, and efficient compression methods are not as commonly used. Analog signals are typically transmitted in their raw form.

4. **Security**:
    - **Digital**: Digital communication supports encryption and other security measures that protect data from unauthorized access and tampering. This enhances data security and privacy.
    - **Analog**: Analog communication is more challenging to secure because encryption methods are not as straightforwardly applied to analog signals, making them more vulnerable to interception and eavesdropping.

5. **Integration with Modern Technology**:
    - **Digital**: Digital communication is better suited for integration with

  modern computing and networking technologies. It is compatible with various digital devices, protocols, and standards, facilitating interoperability and scalability.

  - **Analog**: Analog systems are less compatible with modern digital technology and may require conversion (analog-to-digital or digital-to-analog) to interface with digital systems, potentially introducing additional complexity and potential points of failure.

6. **Scalability and Flexibility**:
   - **Digital**: Digital networks are more scalable and flexible. They can easily handle increasing data loads, support complex data types, and adapt to evolving technologies and standards.

   - **Analog**: Analog systems have limitations in scaling and adapting to new technologies, making them less flexible in dynamic environments.

While analog communication has been historically significant and can still be used in certain applications, **digital communication** offers clear advantages in terms of accuracy, reliability, error handling, data compression, security, and integration with modern technology. For computer communication, which relies on precise and efficient data transfer, digital communication is generally the preferred choice.

Ques 03. What is Windowing? How is flow control and reliability achieved through windowing at transport layer?

Ans : Windowing is a fundamental technique employed in the transport layer of the OSI model to manage data flow and ensure the reliable transmission of data between sender and receiver. This concept is crucial for protocols like Transmission Control Protocol (TCP), which use windowing to handle flow control and maintain the reliability of data communication across networks.

Flow Control
Flow control is a mechanism designed to regulate the rate at which data is transmitted from the sender to the receiver. Its primary purpose is to prevent the receiver's buffer from being overwhelmed by too much data at once, which could lead to data loss or corruption. Windowing achieves flow control through a technique known as the sliding window protocol.

In the sliding window protocol, the sender is permitted to transmit multiple packets before requiring an acknowledgment for the initial packets sent. The size of this window, referred to as the "window size," dictates the maximum number of packets that can be sent without receiving an acknowledgment from the receiver. By adhering to this window size, the sender ensures that it does not exceed the receiver’s capacity to handle incoming data.

The window size is dynamically adjusted based on the receiver’s buffer availability. If the receiver's buffer is approaching its capacity, it may advertise a smaller window size to slow down the sender’s rate of transmission. Conversely, as the receiver processes and acknowledges packets, it may increase the window size, allowing the sender to transmit more packets. This dynamic adjustment helps maintain an optimal flow of data and prevents buffer overflow.

Reliability

Reliability in data communication is essential to ensure that data is delivered accurately and in the correct order. Windowing plays a crucial role in achieving reliability through several mechanisms:

1. **Acknowledgments (ACKs)**: To confirm the successful receipt of packets, the receiver sends acknowledgments back to the sender. If the sender does not receive an acknowledgment within a specified timeout period, it presumes that the packet may have been lost or corrupted and will initiate a retransmission. This process ensures that all packets are eventually received by the receiver.

2. **Sequence Numbers**: Each packet is assigned a unique sequence number, which allows the receiver to reorder packets and detect any missing ones. Sequence numbers enable the receiver to reconstruct the data stream accurately, even if packets arrive out of order or some packets are lost.

3. **Retransmission**: In the event of packet loss or corruption, the receiver can request the retransmission of specific packets. This is typically managed through positive acknowledgments (ACKs) and, in some cases, negative acknowledgments (NACKs). By retransmitting lost or corrupted packets, the sender ensures that the data stream remains intact and complete.

4. **Sliding Window for Retransmissions**: The sliding window protocol also facilitates retransmissions. If a packet within the window is lost or erroneous, the protocol ensures that such packets are retransmitted and acknowledged before advancing the window. This mechanism guarantees that data is transmitted reliably and in sequence.

Windowing is an integral technique in the transport layer, providing essential functions for both flow control and reliability in data communication. Through the sliding window protocol, it manages the rate of data transmission to prevent buffer overflow and dynamically adjusts to network conditions. Additionally, by employing mechanisms such as acknowledgments, sequence numbers, and retransmissions, windowing ensures that data is delivered accurately and in the correct order. As a result, windowing contributes significantly to maintaining efficient and dependable communication channels across networks, adapting to varying conditions and receiver capabilities.

Ques 04. (a) Compare between CSMA/CD and Ethernet protocol. How does CSMA/CD resolve the problem of line connection? Explain.

Ans: In the realm of network communication, particularly within Ethernet networks, the protocol known as **CSMA/CD (Carrier Sense Multiple Access with Collision Detection)** plays a crucial role in managing how multiple devices share a communication channel. Understanding the differences between CSMA/CD and Ethernet, and how CSMA/CD addresses line connection problems, provides insight into effective network management.

CSMA/CD: Managing Access and Collisions
CSMA/CD is a protocol designed to handle data transmission in networks where multiple devices share the same communication medium. Its primary function is to minimize data collisions and ensure that data is transmitted efficiently. The protocol operates through a series of steps:

- **Carrier Sense**: Before transmitting data, a device listens to the network to check if the channel is currently in use. This helps avoid transmitting when another device is already sending data.

- **Multiple Access**: Multiple devices can access the network simultaneously. If the channel is clear, a device begins transmission.

- **Collision Detection**: While transmitting, the device continues to monitor the channel. If a collision is detected (indicating that another device has also started transmitting), the device stops and sends a jam signal to alert all devices of the collision.

- **Collision Resolution**: After a collision, devices wait for a random period before attempting to retransmit. This reduces the likelihood of repeated collisions and ensures a fair chance for all devices to access the channel.

Ethernet Protocol: Incorporating CSMA/CD
Ethernet is a broad networking technology that includes various standards for local area networks (LANs). It encompasses the use of frames for data transmission and integrates protocols like CSMA/CD to manage network traffic effectively. Ethernet protocols define how data is formatted, transmitted, and received over different types of physical media, including twisted pair cables and Fiber optics.

- **Ethernet Frames**: Data is encapsulated in Ethernet frames, which include source and destination addresses, payload, and error-checking information.

- **CSMA/CD Integration**: In traditional Ethernet implementations, such as those using coaxial cables or shared twisted pair cables, CSMA/CD is used to manage access to the shared medium and handle collisions.

- **Full-Duplex Ethernet**: Modern Ethernet standards, such as Gigabit Ethernet, use full-duplex communication, which allows simultaneous transmission and reception of data, thereby eliminating collisions.

Resolving Line Connection Problems with CSMA/CD
CSMA/CD addresses several key issues associated with line connections in shared networks:

1. **Collision Management**: Collisions occur when multiple devices attempt to transmit data simultaneously. CSMA/CD reduces collisions by using carrier sensing to detect if the channel is clear before transmission. If a collision happens, the protocol ensures that devices detect it and retransmit data after a random delay.

2. **Fair Access**: Without a protocol to manage access, some devices could dominate the network, leading to inefficiencies. CSMA/CD promotes fair access by allowing devices to transmit only when the channel is clear and implementing random backoff periods after collisions to prevent unfair access.

3. **Channel Utilization**: Efficient use of the communication channel is critical for network performance. CSMA/CD improves channel utilization by reducing the likelihood of collisions and optimizing retransmissions, leading to better overall performance.

4. **Error Handling**: Data transmission errors can lead to corrupted or incomplete data. CSMA/CD includes mechanisms for detecting and correcting errors through retransmissions and error-checking codes in Ethernet frames, ensuring data integrity.

CSMA/CD is an essential protocol within Ethernet networks that manages how multiple devices share a communication medium, addressing issues such as collisions, fair access, and efficient channel utilization. By employing mechanisms for collision detection and resolution, CSMA/CD ensures reliable and efficient data transmission. As Ethernet technology has evolved, modern implementations often use full-duplex communication to further enhance network performance and eliminate collisions. Understanding the role of CSMA/CD in Ethernet helps in appreciating the complexities of network management and the evolution of networking technologies.

(b) Differentiate between circuit switching and virtual circuit. Also explain the effect of router failure in virtual circuits.

Ans: **Circuit Switching** and **Virtual Circuits** are two fundamental techniques used in network communication to manage how data is transmitted between devices. Although they serve similar purposes, they operate in different ways and are suited to different types of network environments. Understanding their differences helps in selecting the appropriate method for a given network design.

Circuit Switching
Circuit switching is a communication method where a dedicated communication path or circuit is established between two devices for the duration of their conversation. This technique is commonly used in traditional telephone networks.

- **Dedicated Path**: In circuit switching, a dedicated communication path is established for the entire duration of the call or session. This path remains reserved for the exclusive use of the two communicating devices, ensuring consistent and uninterrupted communication.

- **Connection Establishment**: Before data can be transmitted, a connection setup phase is required. The network establishes a specific route between the sender and receiver, reserving the necessary resources for the session.

- **Resource Allocation**: Resources along the established path, such as bandwidth and switching equipment, are reserved and used exclusively for the duration of the session. This guarantees a fixed level of performance and quality.

- **Usage Example**: Circuit switching is used in traditional telephone networks and some private leased-line services, where a consistent and dedicated connection is essential.

Virtual Circuits
Virtual circuits are a method used in packet-switched networks, where a logical path is established between two devices for the duration of their communication session. Unlike circuit switching, virtual circuits do not require a dedicated physical path.

- **Logical Path**: In virtual circuits, a logical path is established within a network, but there is no dedicated physical circuit. Data is divided into packets, each of which follows the logical path defined for the session.

- **Connection Establishment**: Before data transmission begins, a virtual circuit setup phase occurs. The network determines the path that packets will follow and sets up the necessary routing information.

- **Resource Sharing**: Resources are not reserved exclusively for the session. Instead, multiple virtual circuits can share the same physical resources. This allows for more efficient use of network capacity and flexibility in routing.

- **Usage Example**: Virtual circuits are commonly used in modern packet-switched networks such as Frame Relay and Asynchronous Transfer Mode (ATM), where flexible and efficient data transmission is crucial.

Effect of Router Failure in Virtual Circuits
In virtual circuit networks, router failures can significantly impact data transmission. Here's how:

- **Impact on Logical Paths**: Virtual circuits rely on logical paths established through a network of routers. If a router fails, the logical path associated with a virtual circuit can be disrupted. This can lead to delays, packet loss, or the need to reroute the data through alternative paths.

- **Path Recalculation**: When a router fails, the network must recalculate the path for the affected virtual circuits. This involves finding new routes that avoid the failed router, which can introduce additional latency and temporarily affect network performance.

- **Resilience Mechanisms**: Modern virtual circuit networks often include resilience and redundancy mechanisms, such as backup paths and dynamic rerouting, to mitigate the effects of router failures. However, these mechanisms may not always be immediate or perfect, leading to potential temporary disruptions in communication.

Ques 05. Given data frame is 1101011011 and generator polynomial $G(x)=x^4+x+1$. Derive the transmitted frame using CRC method. Write all steps involve in this process.

Ans:

To derive the transmitted frame using the Cyclic Redundancy Check (CRC) method, follow these detailed steps:

1. Represent the Generator Polynomial and Data Frame

- **Data Frame**: 1101011011
- **Generator Polynomial**: $G(x) = x^4 + x + 1$

First, convert the generator polynomial into its binary form. For $G(x) = x^4 + x + 1$, the binary representation is:

- **Binary Representation of Generator Polynomial**: 10011

2. Append Zeros to the Data Frame

Append n zeros to the end of the data frame, where n is the degree of the generator polynomial. In this case, the degree of $G(x)$ is 4. Therefore, append 4 zeros to the data frame:

- **Modified Data Frame**: 1101011011 + 0000 = 11010110110000

3. Perform Binary Division (Modulo-2 Division)

Perform binary division using the generator polynomial 10011 on the modified data frame 11010110110000. The division is carried out as follows:

1. **Align the Generator Polynomial** with the leftmost bits of the data frame that match its length (5 bits):
   - **Dividend**: 11010
   - **Divisor**: 10011
2. **Perform XOR Operation** between the divisor and the current dividend bits:
   - Dividend: 11010
   - Divisor: 10011
   - **Result after XOR**: 01001
3. **Bring Down the Next Bit** from the data frame to the result of the XOR operation:
   - New Dividend: 01001
   - Bring down the next bit: 10010

4. **Align the Divisor** with the new dividend:
   - New Dividend: 10010
   - Divisor: 10011
5. **Perform XOR Operation**:
   - Dividend: 10010
   - Divisor: 10011
   - **Result after XOR**: 00001
6. **Continue Bringing Down Bits** and performing XOR operations until all bits of the modified data frame have been processed.
7. **Final Remainder**: After completing the division process, the remainder obtained is used for error detection.

4. Append the Remainder to the Original Data Frame

Suppose the final remainder obtained from the division process is 1101 (assuming this is the remainder from the division process). Append this remainder to the original data frame:

- **Original Data Frame**: 1101011011
- **Remainder**: 1101 (derived from the division process)
- **Transmitted Frame**: $1101011011 + 1101 = 11010110111101$

Conclusion

To derive the transmitted frame using the CRC method, one must first convert the generator polynomial to binary, append zeros to the data frame, perform binary division using the generator polynomial, and then append the remainder to the original data frame. This transmitted frame, including the CRC code, helps in detecting errors during transmission and ensuring data integrity.

Ques 06. Differentiate between public key cryptography and private key cryptography. Assume two prime numbers p and q are 13 and 17 respectively. Calculate private key and public key using RSA algorithm.

Ans: In cryptography, there are two main types of encryption systems: **Public Key Cryptography** and **Private Key Cryptography**. Both are essential for secure communication, but they differ in how they encrypt and decrypt data.

Public Key Cryptography

- **Definition**: Public key cryptography, also known as asymmetric cryptography, uses two different keys: a **public key** and a **private key**. The public key is shared openly, while the private key is kept secret.

- **Key Pair**: In this system, the public key is used to encrypt the data, and the private key is used to decrypt it. The two keys are mathematically related but cannot be derived from each other easily.

- **Usage**: Public key cryptography is commonly used for secure communications over the internet, such as in HTTPS, digital signatures, and encryption of emails. RSA is a widely used public key encryption system.

- **Example**: RSA, Diffie-Hellman, and Elliptic Curve Cryptography (ECC).

Private Key Cryptography

- **Definition**: Private key cryptography, also known as symmetric cryptography, uses a single key for both encryption and decryption of data.

- **Single Key**: The same key is used by both the sender and the receiver to encrypt and decrypt data. The key must be kept secret by both parties, as the security relies entirely on this shared key.

- **Usage**: Private key cryptography is faster than public key cryptography and is commonly used for encrypting large volumes of data. It is used in protocols like AES (Advanced Encryption Standard) and DES (Data Encryption Standard).

- **Example**: AES, DES, and Blowfish.

Now, let's calculate the **public key** and **private key** using the RSA algorithm, assuming the two prime numbers p=13 and q=17.

1. **Choose Two Prime Numbers**:

   Given $p = 13$ and $q = 17$.

2. **Calculate n**:

   n is the product of p and q.

$$n = p \times q = 13 \times 17 = 221$$

3. **Calculate Euler's Totient Function $\phi(n)$**:

   Euler's totient function $\phi(n)$ is calculated as:

$$\phi(n) = (p - 1) \times (q - 1) = (13 - 1) \times (17 - 1) = 12 \times 16 = 192$$

4. **Choose Public Exponent e**:

   The public exponent e is a number such that $1 < e < \phi(n)$ and $\gcd(e, \phi(n)) = 1$. Let's choose $e = 5$, as it satisfies this condition:

$$\gcd(5, 192) = 1$$

   Therefore, $e = 5$ is valid.

5. **Calculate Private Key d**:

   The private key d is the modular inverse of e modulo $\phi(n)$, meaning:

$$d \times e \equiv 1 \pmod{\phi(n)}$$

   We need to solve for d such that:

$$d \times 5 \equiv 1 \pmod{192}$$

   Using the Extended Euclidean Algorithm, we find that:

$$d = 77$$

6. **Public and Private Keys**:

   - The **public key** is the pair $(e, n) = (5, 221)$.
   - The **private key** is $d = 77$.

Ques 07. (a) Differentiate between pure ALOHA and slotted ALOHA. Give formulas for their throughputs.

Ans: ALOHA is a simple network communication protocol that allows devices to send data over a shared channel without any prior coordination. It was initially developed for wireless communication but has also been applied to wired networks. There are two types of ALOHA: **Pure ALOHA** and **Slotted ALOHA**. Both protocols aim to manage access to the shared channel and resolve collisions, but they differ in their approach and efficiency.

1. Pure ALOHA

- **Definition**: In Pure ALOHA, a device can send data at any time, without waiting for a specific time slot. If two or more devices send data simultaneously, a collision occurs, and the data is lost. When a collision is detected, the device waits for a random amount of time before retransmitting.

- **Transmission**: Devices can transmit data as soon as they have a packet to send. Since there is no coordination, collisions are more frequent.

- **Efficiency**: Pure ALOHA is less efficient because it does not organize transmission times. Collisions can happen at any time, leading to higher retransmissions.

- **Collision Window**: The vulnerable time window for collisions in Pure ALOHA is **2 times the packet time** (i.e., the time required to send one packet). This is because a packet can collide with another packet sent within one packet time before or after the current packet's transmission.

2. Slotted ALOHA

- **Definition**: In Slotted ALOHA, time is divided into equal-sized slots, and devices are allowed to send data only at the beginning of a time slot. If two devices attempt to send data in the same time slot, a collision occurs. Similar to Pure ALOHA, when a collision happens, devices wait for a random amount of time before retransmitting.

- **Transmission**: Devices must wait for the start of the next time slot before transmitting. This reduces the chances of collisions because packets can only collide if they are sent in the same slot.

- **Efficiency**: Slotted ALOHA is more efficient than Pure ALOHA because it reduces the collision window. The possibility of a collision is limited to the duration of one time slot, leading to fewer retransmissions.
- **Collision Window**: In Slotted ALOHA, the vulnerable time window for collisions is only **1 packet time** (i.e., the time required to send one packet), as packets can only collide if they start in the same time slot.

1. Throughput for Pure ALOHA

The throughput S for Pure ALOHA is given by the formula:

$$S = G \times e^{-2G}$$

Where:

- S = throughput (the average number of successful transmissions per frame time)
- G = average number of frames generated by the system in one frame time
- e = Euler's number (approximately 2.718)

The maximum throughput for Pure ALOHA occurs when $G = 0.5$, and it is approximately **18.4%** or **0.184**.

2. Throughput for Slotted ALOHA

The throughput S for Slotted ALOHA is given by the formula:

$$S = G \times e^{-G}$$

Where:

- S = throughput
- G = average number of frames generated in one time slot
- e = Euler's number

The maximum throughput for Slotted ALOHA occurs when $G = 1$, and it is approximately **36.8%** or **0.368**.

(b) Explain the importance of Sliding Window Protocol. Also, list the types of Sliding Window techniques.

Ans: The **Sliding Window Protocol** is crucial in computer networks for managing data transmission between two devices in a reliable and efficient way. It is used at the **Transport Layer** to ensure the successful and orderly delivery of packets. The protocol controls how much data can be sent before needing acknowledgment from the receiving side, helping in **flow control** and **error recovery**.

Key Importance of Sliding Window Protocol:

1. **Flow Control**: The Sliding Window Protocol ensures that the sender does not overwhelm the receiver by sending too much data at once. Each side maintains a window size that represents the maximum number of unacknowledged frames that can be sent. This prevents the receiver's buffer from becoming full, ensuring smooth communication.

2. **Efficient Use of Network Resources**: The protocol allows the sender to transmit multiple packets before receiving an acknowledgment, which optimizes the bandwidth usage. Instead of waiting for an acknowledgment after sending each packet, the sender can continue sending within the window size. This leads to higher throughput, especially in high-latency networks.

3. **Error Control and Reliability**: By using sequence numbers and acknowledgments, the Sliding Window Protocol helps detect and retransmit lost or corrupted packets. If the sender does not receive an acknowledgment for a packet, it retransmits it after a timeout period. This ensures data integrity and reliability in communication.

4. **Congestion Control**: The Sliding Window Protocol dynamically adjusts the window size based on network conditions. This helps manage network congestion. If the network is congested, the window size is reduced, and if the network is clear, it can be increased, helping to maintain efficient transmission without overloading the network.

5. **Ordered Delivery**: The protocol ensures that data is received in the correct order, even if packets are received out of order. Sequence numbers help the receiver reorder the data correctly before passing it to the application layer, ensuring reliable and ordered delivery.

The Sliding Window Protocol comes in different variations based on how it handles acknowledgments and retransmissions. The three main types of sliding window techniques are:

1. Stop-and-Wait ARQ

- In Stop-and-Wait ARQ (Automatic Repeat Request), the sender transmits one frame at a time and waits for an acknowledgment (ACK) before sending the next one. If the sender does not receive an acknowledgment after a certain time (timeout), it retransmits the same frame.

- **Advantages**: Simple to implement and ensures reliable transmission.

- **Disadvantages**: Inefficient use of bandwidth, especially in high-latency networks, because the sender must wait for an acknowledgment after every frame before sending the next one.

2. Go-Back-N ARQ

- In Go-Back-N ARQ, the sender can transmit multiple frames (up to the window size) before needing an acknowledgment. However, if a frame is lost or corrupted, the sender goes back and retransmits that frame along with all subsequent frames, even if they were received correctly.

- **Advantages**: Better bandwidth utilization than Stop-and-Wait, as multiple frames are sent before waiting for an acknowledgment.

- **Disadvantages**: If a single frame is lost, the sender must retransmit all frames starting from that point, which can be inefficient.

3. Selective Repeat ARQ

- In Selective Repeat ARQ, the sender also transmits multiple frames, but if a frame is lost or corrupted, only that specific frame is retransmitted. The receiver stores the correct frames and reorders them if necessary, allowing for more efficient error recovery.

- **Advantages**: More efficient than Go-Back-N ARQ, as only the erroneous frames are retransmitted, reducing unnecessary retransmissions.

- **Disadvantages**: More complex to implement because both the sender and receiver must maintain buffers to store frames for correct reordering.

Ques 08. (a) Write step-by-step working of Link State Routing. Also, compare it with Distance Vector Routing.

Ans: Link State Routing is a dynamic routing protocol used in large networks to maintain an up-to-date view of the network topology. The protocol allows routers to exchange information about the network and make decisions on the best path to forward data. The most commonly used link-state routing algorithm is the **Open Shortest Path First (OSPF)**.

Step-by-Step Process of Link State Routing:

1. **Discover Neighbors**: Each router in the network discovers its directly connected neighbors by sending a special packet known as a **HELLO packet**. This packet is broadcasted to identify other routers on the same network.

2. **Measure Link Cost**: After discovering its neighbors, each router measures the cost to reach its neighbors. This cost can be based on various factors such as link speed, delay, bandwidth, or any other metric. The cost between directly connected routers forms the **link state**.

3. **Construct Link State Packets (LSPs)**: Each router creates a **Link State Packet (LSP)** that contains information about its neighbors and the cost of reaching them. This packet includes:
    - The router's identity.
    - A list of its neighbors.
    - The cost to each neighbor.
    - A sequence number (to ensure the LSP is fresh and not outdated).
    - Time-to-live (TTL) value (to prevent LSP from circulating indefinitely).

4. **Flooding LSPs**: The router sends its LSP to all neighboring routers using a process called **flooding**. Each neighbor receiving the LSP forwards it to its own neighbors, except the one it received the LSP from. This process continues until all routers in the network have received the LSP.

5. **Build the Link State Database (LSDB)**: Each router collects LSPs

from all other routers in the network and stores them in a **Link State Database (LSDB)**. This database contains a complete map of the network's topology, allowing each router to know the structure of the entire network.

6. **Run Dijkstra's Algorithm**: Once the router has the complete network topology in its LSDB, it uses **Dijkstra's Shortest Path First (SPF) algorithm** to calculate the shortest path from itself to every other router in the network. The algorithm considers the costs of all possible paths and determines the most efficient route.

7. **Update Routing Table**: Based on the results of Dijkstra's algorithm, the router updates its **routing table**. The routing table now contains the best path (i.e., the shortest path) to reach every other router in the network.

8. **Handle Link Changes**: Whenever there is a change in the network, such as a link going down or coming up, the affected router generates a new LSP with updated information and floods it to all other routers. Each router then recalculates its shortest path using the updated information.

Link State Routing and Distance Vector Routing are two primary dynamic routing algorithms used in computer networks. They differ in how they operate and the type of information they share with other routers.

1. Information Sharing

- **Link State Routing**:
  - Routers share complete information about their local topology with all routers in the network. Each router has a global view of the network.

  - Each router independently calculates the shortest path to other routers using Dijkstra's algorithm.

- **Distance Vector Routing**:
  - Routers share information only with their direct neighbors, providing a vector (distance and direction) to each destination.

  - Routers do not have a complete map of the network. They rely on the information provided by neighbors and use the Bellman-

Ford algorithm to determine the shortest path.

2. Convergence Time

- **Link State Routing**:
  - **Faster convergence** because routers maintain a complete view of the network. Changes in the network (such as link failures) are quickly communicated, and routers recalculate routes rapidly.
  -
- **Distance Vector Routing**:
  - **Slower convergence**, especially in large networks. Updates take time to propagate through the network, leading to possible routing loops and the **count-to-infinity problem** in certain cases.

3. Scalability

- **Link State Routing**:
  - **More scalable** for large networks due to its efficient handling of complex topologies. However, it requires more memory and CPU resources because each router stores the entire network topology and runs Dijkstra's algorithm.

- **Distance Vector Routing**:
  - **Less scalable**, especially in large networks, because the distance vector protocol can suffer from slow convergence and routing loops.

4. Complexity

- **Link State Routing**:
  - **More complex** to implement and maintain. It requires routers to store the complete network topology and run advanced algorithms like Dijkstra's.

- **Distance Vector Routing**:
  - **Simpler** to implement. Routers only need to know about their neighbors and the distance to different destinations, making it easier to configure in smaller networks.

5. Resource Requirements

- **Link State Routing**:
  - **Higher resource usage** in terms of CPU, memory, and bandwidth due to the need to store and compute the entire

network's topology.

- **Distance Vector Routing**:
    - **Lower resource usage**, as routers only need to store information about their direct neighbors and the distance to different networks.

(b) Explain leaky bucket algorithm for congestion control. Also lists its advantages and disadvantages.

Ans: The **Leaky Bucket Algorithm** is a method used to control data transmission and prevent network congestion. It regulates the flow of data by controlling the rate at which packets are sent into the network, ensuring a consistent output even if incoming data arrives at irregular intervals. The algorithm is widely used in networking for **traffic shaping**, **rate limiting**, and **congestion control**.

The leaky bucket algorithm can be visualized as a bucket with a small hole at the bottom. Water (representing data packets) enters the bucket at variable rates, but it leaves (i.e., is transmitted into the network) at a fixed rate through the hole. If water (data) enters the bucket faster than it can exit, the bucket overflows, representing data loss due to congestion.

Steps Involved in the Leaky Bucket Algorithm:

1. **Data Arrival**: Data packets arrive at the source node at any rate. These packets are placed in the buffer (the "bucket"). The data may arrive in bursts or irregular patterns, but the algorithm smooths the flow of packets.

2. **Data Transmission**: The data is transmitted from the buffer at a fixed, constant rate, regardless of how fast or slow the incoming data is. This ensures that the flow of data into the network is smooth and controlled.

3. **Overflow and Data Loss**: If the incoming data rate exceeds the transmission rate and the buffer becomes full, any additional packets are discarded (overflow), causing data loss.

4. **Empty Bucket Scenario**: If there is no incoming data, the bucket (buffer) becomes empty, and no packets are sent until new data arrives.

In essence, the leaky bucket algorithm transforms a bursty data flow into a steady stream, which helps prevent sudden network congestion and ensures that the network operates smoothly.

Advantages of the Leaky Bucket Algorithm

1. **Smooth Data Flow**: The primary advantage of the leaky bucket algorithm is its ability to smooth out bursts of data, providing a steady and predictable rate of data transmission. This helps avoid congestion caused by sudden surges of data packets.

2. **Prevents Buffer Overflow**: The algorithm prevents buffer overflow by limiting the amount of data that can be sent into the network. By discarding excess packets when the buffer is full, the algorithm ensures that the network is not overwhelmed by too much data at once.

3. **Simple and Easy to Implement**: The leaky bucket algorithm is relatively simple to understand and implement. It requires a fixed-size buffer and a mechanism for discarding packets when the buffer is full, making it a straightforward solution for traffic shaping.

4. **Fairness**: Since the leaky bucket algorithm enforces a constant transmission rate, it ensures fairness by preventing any one sender from flooding the network with excessive data. All senders are treated equally, with each being limited to the fixed output rate.

5. **Congestion Control**: By controlling the rate of data flow, the leaky bucket algorithm helps prevent congestion in the network, reducing the chances of packet loss, delays, and performance degradation.

Disadvantages of the Leaky Bucket Algorithm

1. **Data Loss**: If the buffer becomes full and new packets continue to arrive, the leaky bucket algorithm discards the excess packets. This can lead to **data loss**, especially when the incoming data rate is higher than the output rate for extended periods.

2. **Not Suitable for Variable Output Requirements**: The fixed transmission rate of the leaky bucket algorithm is not ideal for situations where varying output rates are needed. Networks that require flexibility in transmission rates may find the algorithm limiting.

3. **Inefficient Use of Bandwidth**: In some scenarios, the fixed-rate output can lead to inefficient use of available bandwidth. For example, if the network can handle higher traffic for a short time, the leaky bucket's constant rate may not fully utilize the available capacity.

4. **Delay in Transmission**: Since the algorithm smooths out bursts of data, packets may experience delays before being transmitted. This can be problematic in real-time applications like video streaming or voice-over-IP (VoIP), where low latency is critical.

5. **No Adaptability to Changing Network Conditions**: The leaky bucket algorithm does not dynamically adapt to network congestion or other conditions. Even if the network can temporarily handle a higher data rate, the algorithm continues to transmit at the fixed rate, which might result in under-utilizing available resources.