# Computer Networks

## Flow Control Mechanism

-Sankalp Nayak

- **Flow Control** is a technique used in data communication to manage the pace of data transmission between a sender and receiver. It ensures that the sender does not overwhelm the receiver by sending data faster than it can be processed. Flow control mechanisms are crucial for maintaining data integrity and preventing buffer overflow at the receiving end.

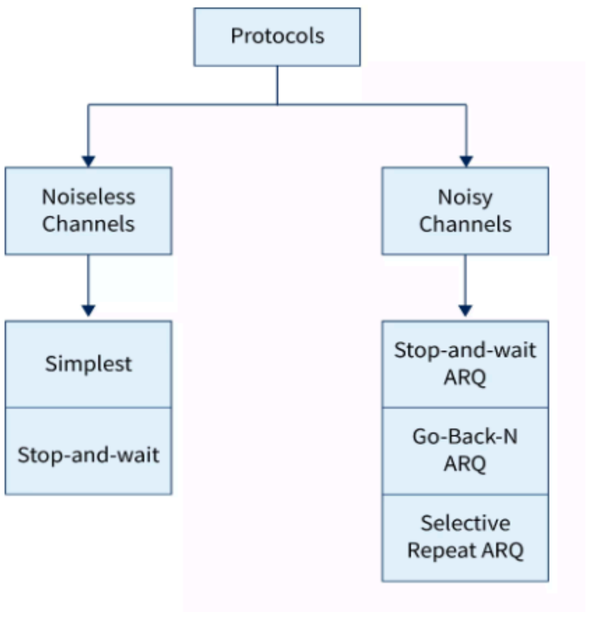
### 1. Purpose of Flow Control

- **Prevent Data Loss:** Avoids the loss of data due to buffer overflow at the receiver.
- **Ensure Efficient Data Transmission:** Ensures that the network resources are utilized effectively, preventing congestion and ensuring a smooth flow of data.
- **Manage Variable Data Rates:** Handles the differences in data processing speeds between the sender and the receiver.

### 2. Classifications of channels based on their operational characteristics

**1. Noiseless Channels**
A noiseless channel is an ideal communication medium where data is transmitted without any errors or interference. The signal reaches the receiver exactly as it was sent, with no need for error detection or correction.

**2. Noisy Channels**
A noisy channel is a more realistic communication medium where data can be affected by interference, leading to errors. Error detection and correction techniques are necessary to ensure reliable communication in noisy channels.



## 3. Stop and Wait ARQ

- Stop and Wait ARQ (Automatic Repeat reQuest) is a basic yet fundamental method used in networking to ensure the reliable delivery of data frames between a sender and a receiver. This protocol is specifically designed to handle errors that might occur during data transmission over noisy channels or unreliable links. The essence of the Stop and Wait ARQ protocol is its simplicity: the sender sends one data frame and then waits for an acknowledgment (ACK) from the receiver before sending the next frame.

### Understanding Stop and Wait.

The Stop and Wait ARQ works on a straightforward principle. The sender transmits a single data frame and then halts further transmission until it receives an acknowledgment from the receiver. The acknowledgment serves as a confirmation that the data frame has been successfully received and processed by the receiver. Only after receiving this acknowledgment does the sender proceed to send the next data frame.

This process of sending one frame, waiting for an acknowledgment, and then sending the next frame is why the protocol is named "Stop and Wait." The sender stops sending data and waits until it gets the go-ahead from the receiver to continue.

### Key Characteristics of Stop and Wait ARQ

1. **Connection-Oriented Communication:** Establishes a connection before data transmission.
2. **Error and Flow Control:** Ensures correct frame reception and prevents sender overload.
3. **Layer Usage:** Used at Data Link and Transport Layers for reliable communication.
4. **Sliding Window Protocol with Window Size 1:** Sends one frame at a time, and waits for acknowledgment.
5. **Half-Duplex Link:** Data transmission occurs in one direction at a time.
6. **Throughput:** Limited to one frame per Round-Trip Time (RTT).
7. **Efficiency Considerations:** Inefficient with high bandwidth-delay products due to waiting for acknowledgments.
8. **Sequence Numbers:** Uses two sequence numbers (0 and 1) to differentiate frames.

### Useful Terms in Stop and Wait Protocol

- **Round Trip Time (RTT)**:

RTT is the time it takes for a packet to reach the receiver and for the acknowledgment to return to the sender. It is a critical factor in determining the performance of Stop and Wait ARQ. ( $RTT = 2 * T_p$)

- **Timeout (TO)**:

The timeout period is typically set to twice the RTT (2 * RTT). If the sender does not receive an acknowledgment within this time frame, it assumes that the frame was lost or corrupted and retransmits the frame.

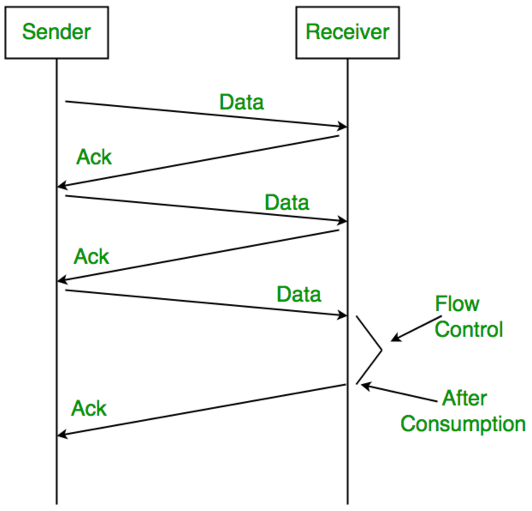
- **Time To Live (TTL)**:

TTL is a mechanism that limits the lifespan of a data packet in a network. In Stop and Wait ARQ, TTL is often set to twice the timeout period, with a maximum value of 255 seconds.

## Working of Stop and Wait ARQ

Let's walk through a typical interaction in Stop and Wait ARQ:

1. **At the Sender**:
   - The sender follows two simple rules:
     - Rule 1: Send one data packet at a time.
     - Rule 2: Send the next packet only after receiving an acknowledgment for the previous packet.
2. **At the Receiver**:
   - The receiver has two corresponding rules:
     - Rule 1: Send an acknowledgment after receiving and processing a data packet.
     - Rule 2: Ensure the acknowledgment is sent after consuming the data packet to maintain flow control.

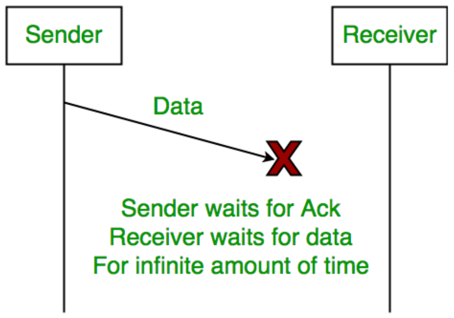


## Problems Associated with Stop and Wait

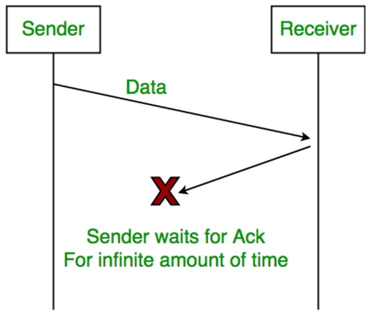
Despite its simplicity, the Stop and Wait ARQ protocol faces several challenges:

1. **Lost Data**:

If a data packet is lost during transmission, the receiver will not receive it and hence will not send an acknowledgment. The sender, waiting indefinitely for the acknowledgment, will not send any further data. This can halt communication entirely.



2. **Lost Acknowledgment**:
Even if the receiver successfully receives a data packet, the acknowledgment could be lost in the network. The sender must receive the acknowledgment to send the next data packet, leading to delays.

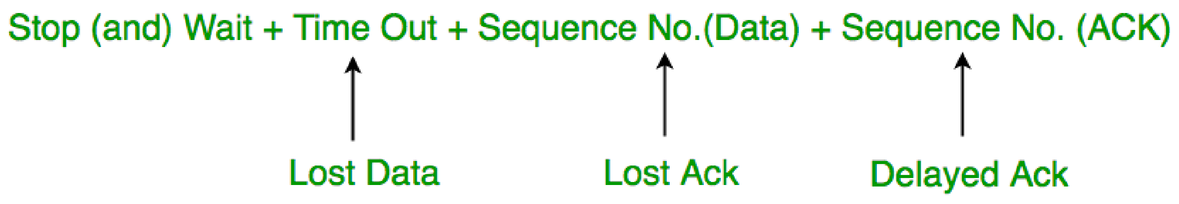


3. **Delayed Acknowledgment/Data**:
Sometimes, acknowledgments or data packets may be delayed due to network congestion. A delayed acknowledgment might arrive after the sender's timeout period has expired, causing the sender to retransmit the packet unnecessarily.

## <u>Stop and Wait ARQ (Automatic Repeat Request)</u>

The Stop and Wait ARQ protocol addresses the above issues by introducing mechanisms for error control and flow control. Here's how it works:

1. **Timeout**:
If the sender does not receive an acknowledgment within a predefined timeout period, it assumes the frame was lost or corrupted and retransmits it.
2. **Sequence Numbers for Data**:
Each data frame is assigned a sequence number (0 or 1), which helps the receiver identify whether the received frame is a new one or a retransmission.
3. **Sequence Numbers for Acknowledgment**:
The receiver uses sequence numbers in the acknowledgment to indicate the next expected data frame. This ensures that the sender knows whether the previous transmission was successful.


Stop (and) Wait + Time Out + Sequence No.(Data) + Sequence No. (ACK)
Lost Data
Lost Ack
Delayed Ack

**Working of Stop and Wait ARQ**

**1. Data Transmission:**

- **Sender (A):** Sends a data frame or packet with sequence number 0.
- **Receiver (B):** Receives the data frame with sequence number 0.

**2. Acknowledgment:**

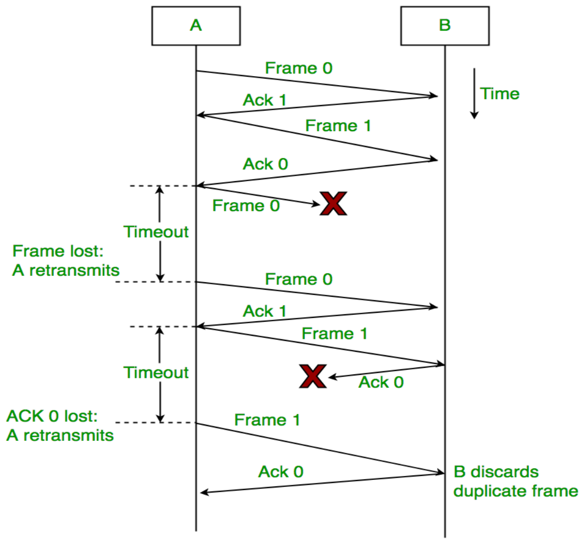
- After successfully receiving the data frame, Receiver B sends an acknowledgment (ACK) with sequence number 1. This sequence number indicates that the receiver expects the next data frame to have the sequence number 1.
- The acknowledgment numbers always announce the sequence number of the next frame expected by the receiver. For example, if frame 0 has arrived safe and sound, the receiver sends an ACK frame with acknowledgment 1. If frame 1 has arrived safe and sound, the receiver sends an ACK frame with acknowledgment 0 (meaning frame 0 is expected).

**3. Sequence Number Usage:**

- The Stop and Wait ARQ protocol uses a one-bit sequence number. This means that both the sender and receiver have buffers capable of holding only one data frame or packet at a time. The sequence number alternates between 0 and 1, allowing the sender to keep track of the frames it has sent and the receiver to know which frame is expected next.

**4. Working Example:**

- **Sender A** sends a frame with sequence number 0.
- **Receiver B** sends an ACK with sequence number 1 upon receiving the frame with sequence number 0.
- **Sender A** then sends the next frame with sequence number 1 after receiving the ACK.



**Efficiency:** Efficiency in the Stop and Wait ARQ protocol measures how well the channel is utilized for sending data. It reflects the proportion of time the channel is actively used for transmitting data versus waiting for acknowledgments.

Calculation: The total time to send one data packet and receive its acknowledgment can be calculated as:

$$\text{Total Time} = T_t(\text{data}) + 2 \times T_p$$

where:

- $T_t(\text{data})$ is the transmission delay for the data packet.
- $T_p$ is the propagation delay.

In cases where other delays like queuing and processing are negligible, the formula simplifies to:

$$\text{Total Time} = T_t(\text{data}) + 2 \times T_p$$

**Efficiency Formula:**

$$\eta = \frac{\text{Useful Time}}{\text{Total Cycle Time}}$$

Given that the useful time is the time taken to transmit the data packet:

$$\eta = \frac{T_t(\text{data})}{T_t(\text{data})+2\times T_p}$$

Alternatively, using the ratio $a = \frac{T_p}{T_t(\text{data})}$:

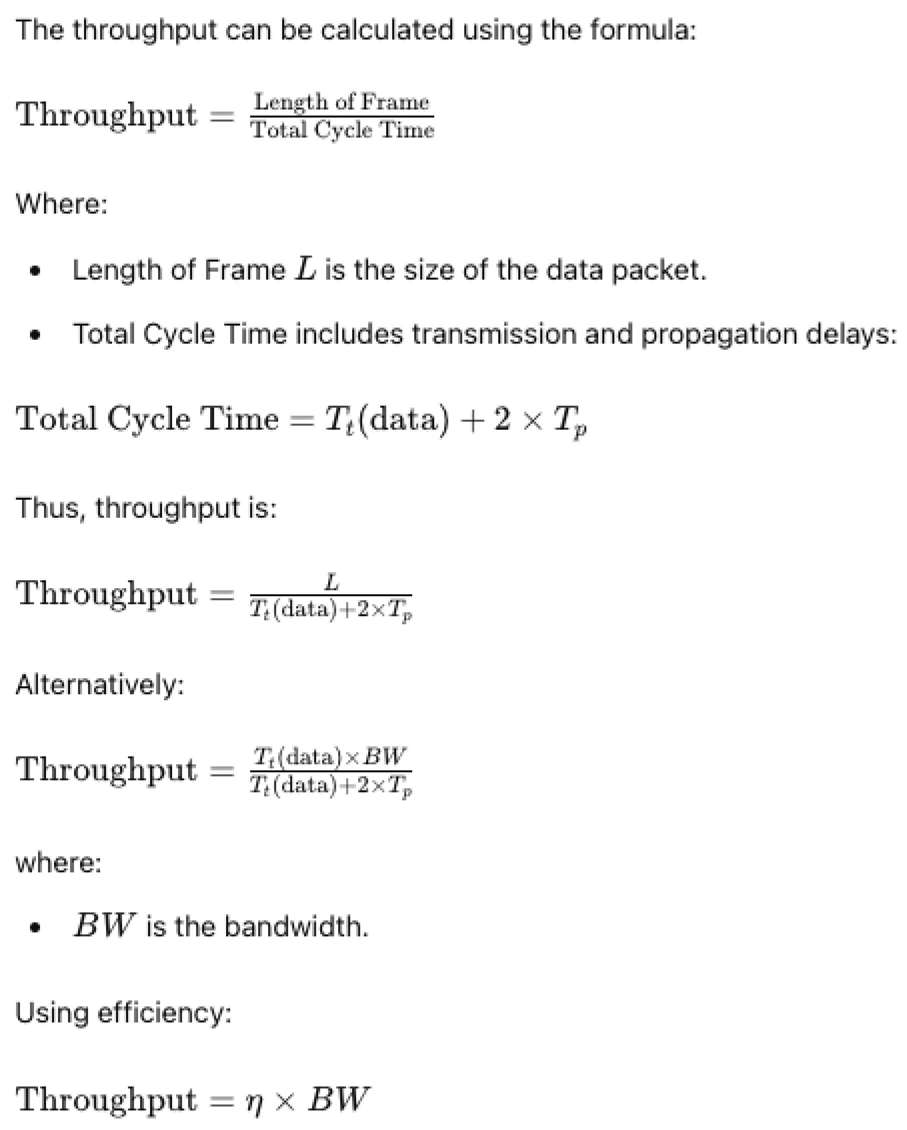
$$\eta = \frac{1}{1+2\times a}$$

where:

- $a = \frac{T_p}{T_t(\text{data})}$

**Throughput:** Throughput measures the rate at which data is successfully transmitted over the network. It is also known as effective bandwidth or bandwidth utilization.

**Calculation:**



The throughput can be calculated using the formula:

$$\text{Throughput} = \frac{\text{Length of Frame}}{\text{Total Cycle Time}}$$

Where:

- Length of Frame $L$ is the size of the data packet.
- Total Cycle Time includes transmission and propagation delays:

$$\text{Total Cycle Time} = T_t(\text{data}) + 2 \times T_p$$

Thus, throughput is:

$$\text{Throughput} = \frac{L}{T_t(\text{data}) + 2 \times T_p}$$

Alternatively:

$$\text{Throughput} = \frac{T_t(\text{data}) \times BW}{T_t(\text{data}) + 2 \times T_p}$$

where:

- $BW$ is the bandwidth.

Using efficiency:

$$\text{Throughput} = \eta \times BW$$

**Example Calculation**

Given:

- Transmission Delay $T_t = 1 \text{ ms}$
- Propagation Delay $T_p = 2 \text{ ms}$
- Bandwidth $BW = 6 \text{ Mbps}$

**Efficiency:**

$$\eta = \frac{1}{1+2\times\frac{T_p}{T_t}} = \frac{1}{1+2\times\frac{2}{1}} = \frac{1}{5} = 0.2 \text{ or } 20\%$$

**Throughput:**

$$\text{Throughput} = \eta \times BW = 0.2 \times 6 \text{ Mbps} = 1.2 \text{ Mbps}$$

**Summary:**

- **Efficiency** reflects the proportion of time the channel is used for data transmission and is given by $\eta = \frac{1}{1+2\times a}$.
- **Throughput** represents the effective data rate and is calculated as $\text{Throughput} = \eta \times BW$.

**3. Capacity of the Link:**

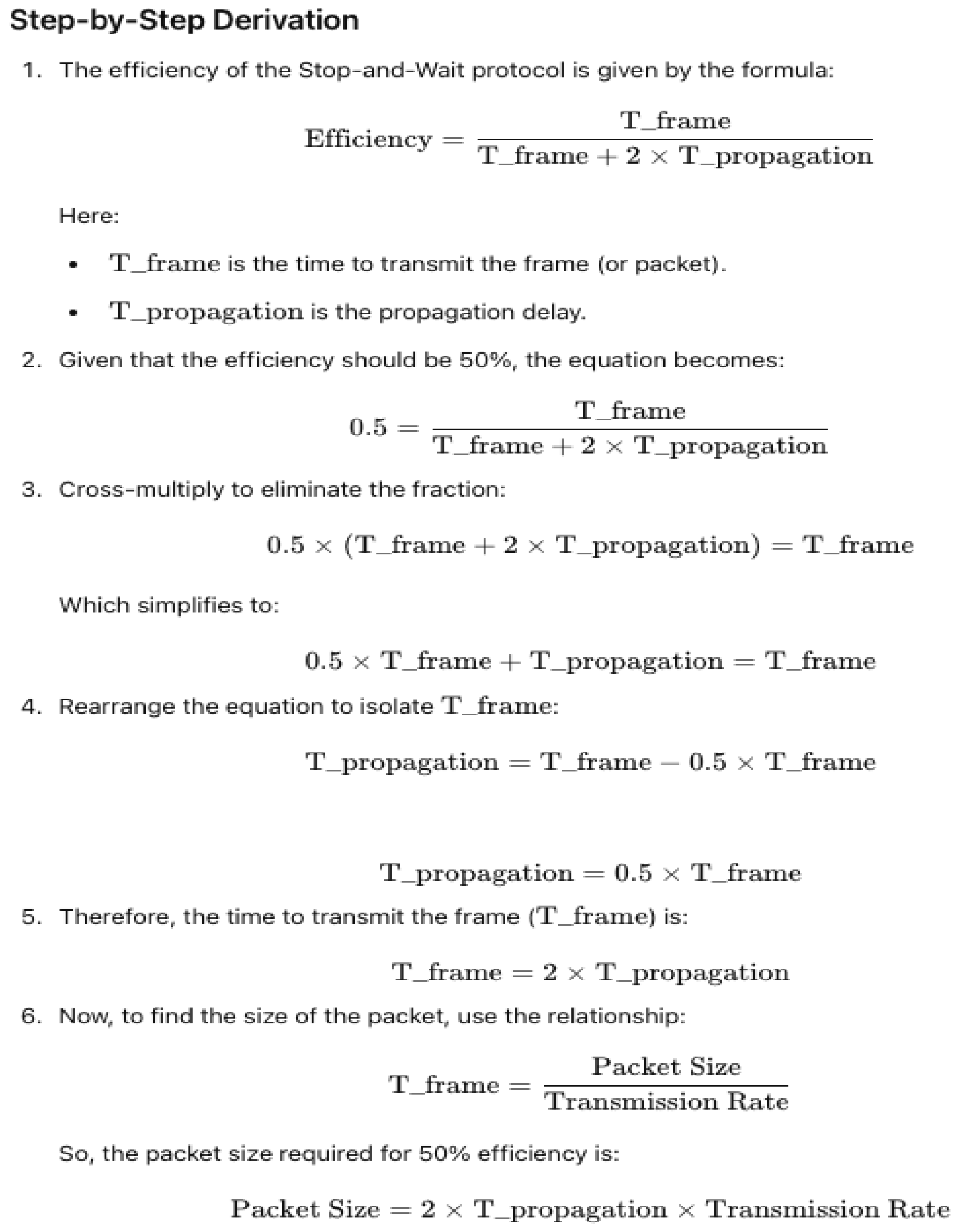
- **Definition:** Maximum number of bits that a channel or link can hold at any given time.
- **Formula:**

$$\text{Capacity} = B \times Tp$$

- **For Full Duplex Channels:**
  - In full duplex channels, data can be transmitted in both directions simultaneously without collisions.
  - **Formula:**

$$\text{Capacity (Full Duplex)} = 2 \times B \times Tp$$

Let's derive the size of the packet required to achieve 50% efficiency in the Stop-and-Wait protocol.



### Step-by-Step Derivation

1. The efficiency of the Stop-and-Wait protocol is given by the formula:

$$\text{Efficiency} = \frac{\text{T\_frame}}{\text{T\_frame} + 2 \times \text{T\_propagation}}$$

   Here:

   - $\text{T\_frame}$ is the time to transmit the frame (or packet).
   - $\text{T\_propagation}$ is the propagation delay.

2. Given that the efficiency should be 50%, the equation becomes:

$$0.5 = \frac{\text{T\_frame}}{\text{T\_frame} + 2 \times \text{T\_propagation}}$$

3. Cross-multiply to eliminate the fraction:

$$0.5 \times (\text{T\_frame} + 2 \times \text{T\_propagation}) = \text{T\_frame}$$

   Which simplifies to:

$$0.5 \times \text{T\_frame} + \text{T\_propagation} = \text{T\_frame}$$

4. Rearrange the equation to isolate $\text{T\_frame}$:

$$\text{T\_propagation} = \text{T\_frame} - 0.5 \times \text{T\_frame}$$

$$\text{T\_propagation} = 0.5 \times \text{T\_frame}$$

5. Therefore, the time to transmit the frame ($\text{T\_frame}$) is:

$$\text{T\_frame} = 2 \times \text{T\_propagation}$$

6. Now, to find the size of the packet, use the relationship:

$$\text{T\_frame} = \frac{\text{Packet Size}}{\text{Transmission Rate}}$$

   So, the packet size required for 50% efficiency is:

$$\text{Packet Size} = 2 \times \text{T\_propagation} \times \text{Transmission Rate}$$

## Constraints in Stop and Wait ARQ

### 1. Efficiency Issues:

● **Low Efficiency:** Stop and Wait ARQ can be inefficient because the sender must wait for an acknowledgment before sending the next data frame. This leads to idle time, especially in high-bandwidth and high-latency environments.

● **Single Frame Buffer:** With only one frame being sent and acknowledged at a time, the channel bandwidth is underutilized, leading to poor performance when dealing with large amounts of data.

### 2. High Bandwidth and Propagation Delay:

● **High Latency:** In cases where there is high bandwidth but also significant propagation delay (e.g., long-distance communication), Stop and Wait ARQ performs poorly. The sender's constant waiting for acknowledgments means that it is unable to utilize the available bandwidth effectively.

- **Example Scenario:** For instance, if you are connected to a server across the world with a high-speed connection, the high propagation delay can severely impact the performance of Stop and Wait ARQ.

**3. Potential Solutions:**

- **Window Size Increase:** Increasing the window size can improve efficiency. However, Stop and Wait ARQ itself is limited to a window size of 1.
- **Alternative Protocols:** To address the inefficiencies of Stop and Wait ARQ, protocols such as **Go-Back-N** and **Selective Repeat** can be used. These protocols allow for multiple frames to be sent before requiring an acknowledgment, thus better utilizing the available bandwidth and handling higher delays more effectively.

**4. Applicability:**

- **LAN Connections:** Stop and Wait ARQ works well in environments with low propagation delay, such as Local Area Networks (LANs).
- **Satellite Connections:** For distant connections, such as satellite links, Stop and Wait ARQ may not be efficient due to high propagation delay and the consequent underutilization of bandwidth.

**Advantages of Stop and Wait ARQ**

- **Simple Implementation:** Easy to implement with minimal resources.
- **Error Detection:** Uses checksums or CRC to detect errors and request retransmissions.
- **Reliable Data Transmission:** Ensures data is transmitted correctly and in sequence.
- **Flow Control:** Manages data flow to prevent receiver buffer overflow.
- **Compatibility:** Widely compatible with existing systems for unreliable channels.

**Disadvantages of Stop and Wait ARQ**

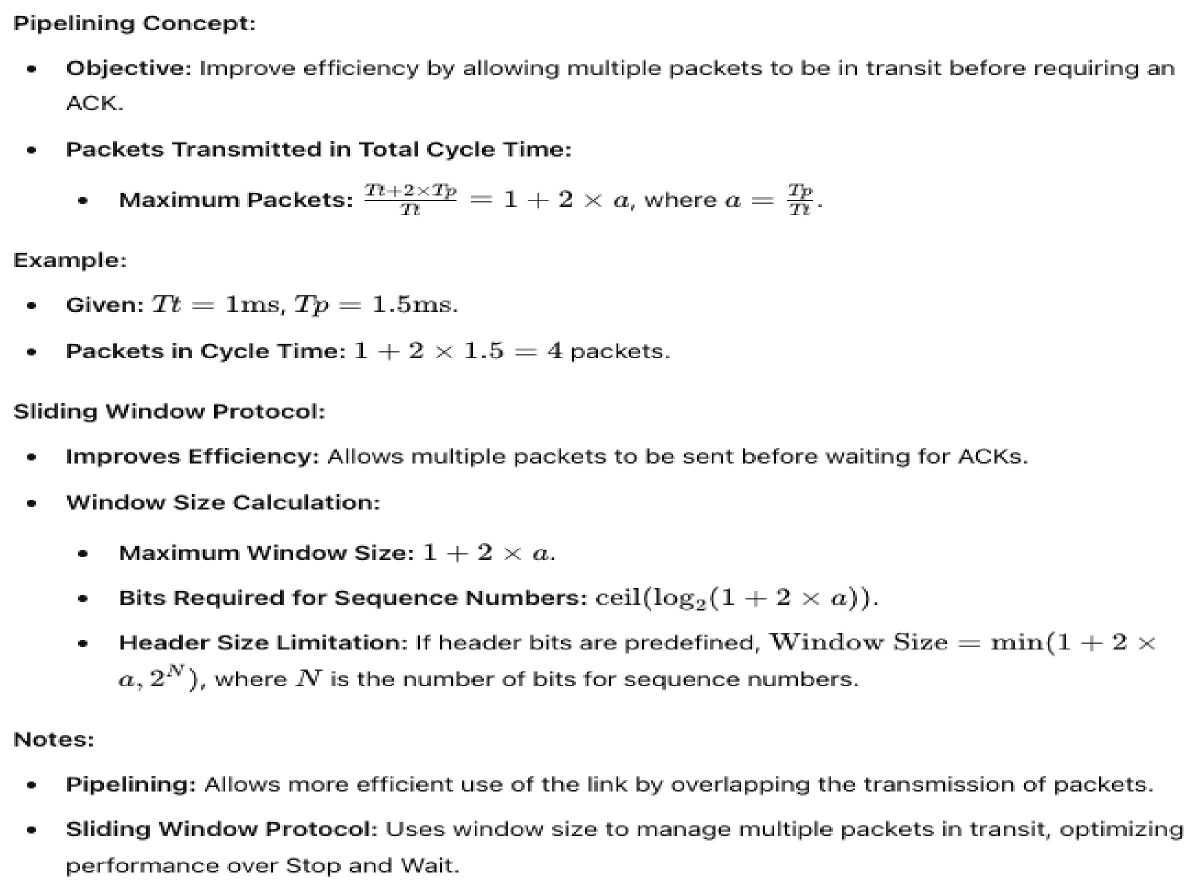
- **Low Efficiency:** Waiting for acknowledgments before sending the next packet reduces efficiency, especially in high-latency networks.
- **High Latency:** Introduces delays, problematic for time-sensitive applications.
- **Limited Bandwidth Utilization:** Single-frame transmission approach does not fully utilize available bandwidth.
- **Limited Error Recovery:** Retransmits the entire frame if lost or corrupted, causing delays.
- **Vulnerability to Channel Noise:** Susceptible to errors from noise, leading to frequent retransmissions.

## <u>Sliding Window Protocol</u>

- The Sliding Window Protocol is a method used in networking to manage data transmission between sender and receiver. It allows multiple frames to be sent before needing an acknowledgment for the first one, improving efficiency over the Stop and Wait protocol.

**Key Concepts:**

1. **Window Size:**
   - **Definition:** The window size determines the number of frames that can be sent before receiving an acknowledgment. It is a crucial factor in the sliding window protocol, impacting throughput and efficiency.
   - **Fixed or Variable:** The size of the window can be fixed or adjusted dynamically based on network conditions and protocol implementation.
2. **Sequence Numbers:**
   - **Assignment:** Frames are assigned unique sequence numbers in a cyclic manner, ranging from 0 to $2^{N-1}$, where N is the number of bits used for sequence numbers.
   - **Cyclic Nature:** After reaching the maximum sequence number, numbering restarts from 0.
3. **Sender's Window:**
   - **Range:** Contains the sequence numbers of frames that have been sent but not yet acknowledged.
   - **Advancement:** The window slides forward as acknowledgments are received. New frames are given the next sequence number, and the upper edge of the window moves forward by one frame.
4. **Receiver's Window:**
   - **Range:** Contains the sequence numbers of frames that the receiver is prepared to accept. The receiver acknowledges frames within this window.
   - **Acknowledgment:** The acknowledgment process allows the sender to advance the window and send new frames.
5. **Buffer Requirements:**
   - **Sender:** Requires a number of buffers equal to the window size to hold unacknowledged frames.
   - **Receiver:** May also require buffers to manage frames within its window.
6. **Operation:**
   - **Sending Frames:** The sender can send multiple frames up to the window size before needing to wait for an acknowledgment.
   - **Acknowledging Frames:** As the receiver processes frames, it sends acknowledgments for received frames, allowing the sender's window to advance.
7. **Efficiency:**
   - **Throughput:** Sliding window protocols improve throughput by allowing multiple frames to be in transit simultaneously, reducing idle time.
   - **Window Size Impact:** A larger window size can increase throughput but requires more buffer space and may introduce more complexity in managing the window and acknowledgments.
8. **Protocols:**
   - **Go-Back-N (GBN):** A type of sliding window protocol where if an error is detected, all subsequent frames are retransmitted.
   - **Selective Repeat (SR):** A sliding window protocol where only the erroneous frames are retransmitted, allowing for more efficient use of the window.



**Pipelining Concept:**

- **Objective:** Improve efficiency by allowing multiple packets to be in transit before requiring an ACK.
- **Packets Transmitted in Total Cycle Time:**
  - **Maximum Packets:** $\frac{Tt+2\times Tp}{Tt} = 1 + 2 \times a$, where $a = \frac{Tp}{Tt}$.

**Example:**

- **Given:** $Tt = 1\text{ms}$, $Tp = 1.5\text{ms}$.
- **Packets in Cycle Time:** $1 + 2 \times 1.5 = 4$ packets.

**Sliding Window Protocol:**

- **Improves Efficiency:** Allows multiple packets to be sent before waiting for ACKs.
- **Window Size Calculation:**
  - **Maximum Window Size:** $1 + 2 \times a$.
  - **Bits Required for Sequence Numbers:** $\text{ceil}(\log_2(1 + 2 \times a))$.
  - **Header Size Limitation:** If header bits are predefined, $\text{Window Size} = \min(1 + 2 \times a, 2^N)$, where $N$ is the number of bits for sequence numbers.

**Notes:**

- **Pipelining:** Allows more efficient use of the link by overlapping the transmission of packets.
- **Sliding Window Protocol:** Uses window size to manage multiple packets in transit, optimizing performance over Stop and Wait.

## Go-Back-N ARQ (Automatic Repeat reQuest)

● **Go-Back-N ARQ** (Automatic Repeat reQuest) is a protocol used for reliable data transfer. It falls under the category of sliding window protocols, where the sender can send multiple frames before needing an acknowledgment (ACK) from the receiver. This protocol is commonly used in scenarios requiring in-order delivery of packets over potentially unreliable or noisy communication channels.

### 1. Window Size and Range of Sequence Numbers

● **Sender's Window Size (N):**

○ The sender can send up to N frames before pausing for an acknowledgment. The size of the sender's window determines the number of outstanding frames (frames sent but not yet acknowledged) that the sender can handle.

○ The sender window size is crucial because it controls the flow of data and the amount of data the sender can transmit without waiting for an ACK.

● **Receiver's Window Size:**

○ In Go-Back-N ARQ, the receiver's window size is always 1, meaning it can only accept the next expected frame in sequence.

○ This ensures that frames are received in the correct order. If a frame is lost or arrives out of order, the receiver discards it and waits for the correct frame.

● **Sequence Numbers:**

○ The sequence numbers range from 0 to $2^n$ -1, where n is the number of bits used for the sequence number field. The sequence numbers are used cyclically, and each frame is assigned a unique sequence number within this range.

**2. Sender's Operation**

- **Transmission of Frames:**
  - The sender transmits frames within its window size, without waiting for an immediate acknowledgment.
  - If the sender's window size is 4, it can send frames 0, 1, 2, and 3 without waiting for an ACK.
- **Sliding the Window:**
  - Once the sender receives an ACK for the first frame (e.g., frame 0), it slides the window forward, allowing it to send a new frame (e.g., frame 4).
  - This sliding window mechanism helps in maintaining a continuous flow of data, optimizing the use of the communication link.
- **Sender's Window Range:**
  - The window range is from the lowest unacknowledged sequence number to the highest sequence number that can be sent. The window slides forward as ACKs are received.

**3. Receiver's Operation**

- **Receiving Frames:**
  - The receiver only accepts the next expected frame in sequence. If it receives a frame out of order, it discards it.
  - For example, if the receiver is expecting frame 1 but receives frame 2, it will discard frame 2 and wait for frame 1.
- **Acknowledgments (ACK):**
  - The receiver sends an ACK for the last correctly received frame. This ACK is cumulative, meaning it acknowledges the receipt of all previous frames up to that point.
  - For example, if the receiver successfully receives frames 0, 1, and 2, it will send an ACK for frame 2. This implies that frames 0 and 1 have also been received correctly.

**4. Error Handling**

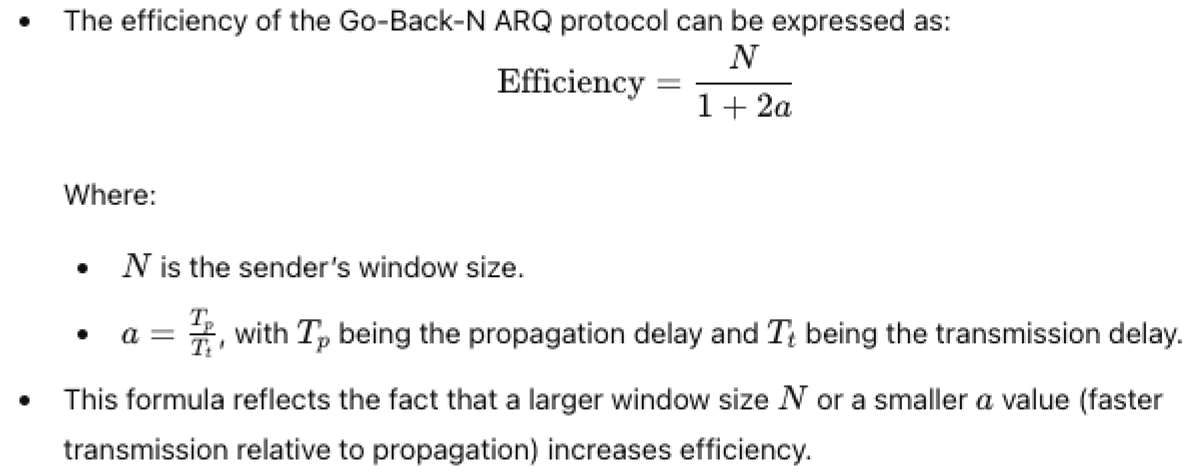
- **Lost or Damaged Frames:**
  - If a frame is lost or damaged, the receiver will not send an ACK for that frame. The sender will eventually notice the missing ACK when its timer expires.
  - Upon timer expiration, the sender retransmits all frames starting from the lost or damaged frame, even if subsequent frames were received correctly by the receiver.
- **Timer Mechanism:**
  - The sender sets a timer for each transmitted frame. If the timer expires before an ACK is received, the sender retransmits the frame and all subsequent frames within the window.

**5. Cumulative Acknowledgment**

- **Cumulative ACK:**
  - Go-Back-N ARQ uses cumulative ACKs. This means that an ACK for a frame implicitly acknowledges all previous frames.
  - For example, if the sender receives an ACK for frame 4, it knows that frames 0, 1, 2, 3, and 4 have been received successfully.

## 6. Efficiency and Formula

- **Efficiency (η):**
  - Efficiency is the ratio of the time spent transmitting useful data to the total time taken, including the time lost due to errors and retransmissions.
  - The efficiency of Go-Back-N ARQ can be calculated using the following formula:



- The efficiency of the Go-Back-N ARQ protocol can be expressed as:

$$\text{Efficiency} = \frac{N}{1+2a}$$

Where:

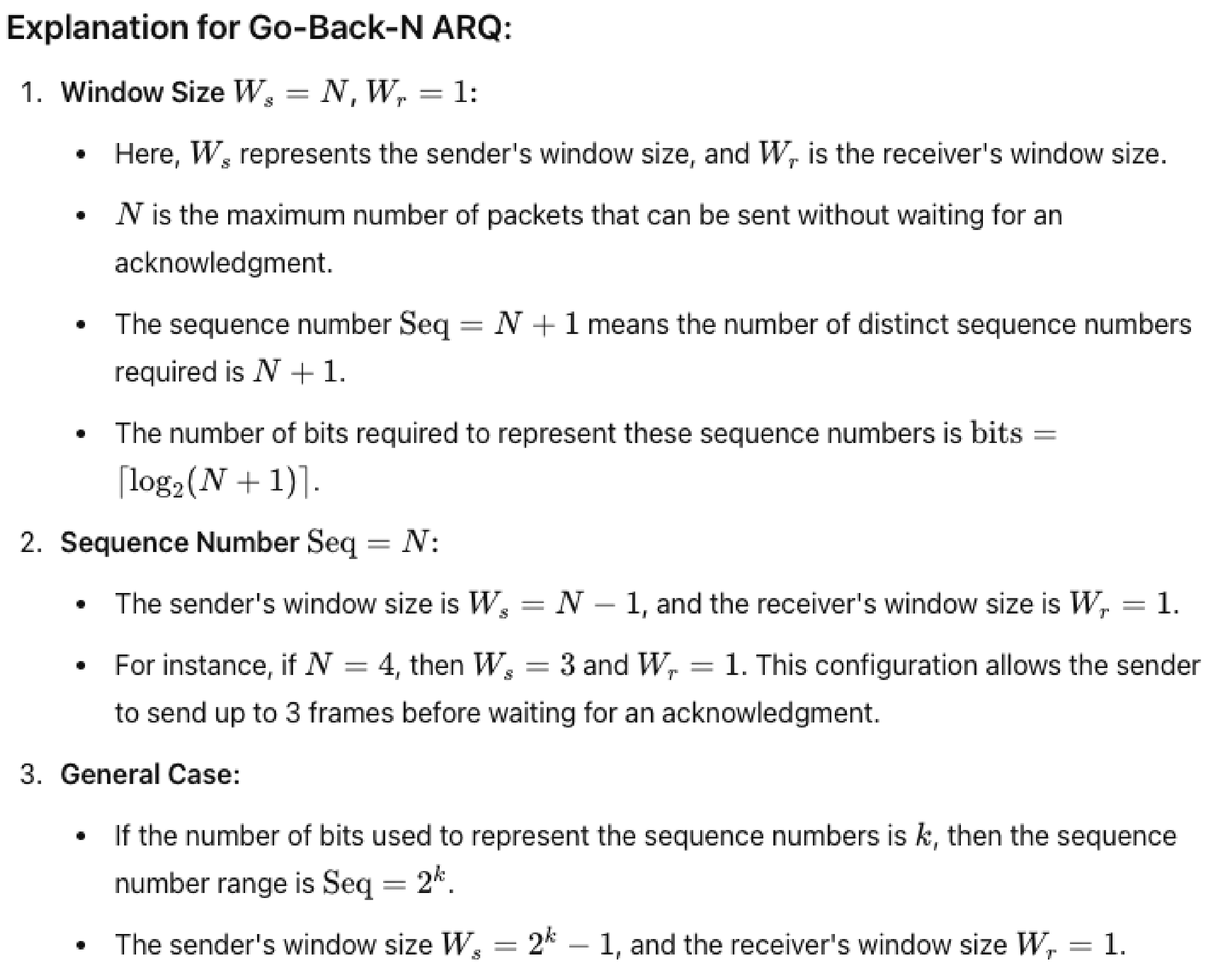
- $N$ is the sender's window size.
- $a = \frac{T_p}{T_t}$, with $T_p$ being the propagation delay and $T_t$ being the transmission delay.

- This formula reflects the fact that a larger window size $N$ or a smaller $a$ value (faster transmission relative to propagation) increases efficiency.

- **Explanation:**
  - Higher window sizes (N) increase efficiency as they allow more frames to be in transit before needing an acknowledgment.
  - However, the efficiency decreases with an increase in propagation delay (Tp). High propagation delays lead to longer wait times for ACKs, reducing overall efficiency.
- **Effective Bandwidth (EB):**
  - The effective bandwidth or throughput is the number of bits successfully sent per second.
  - It can be calculated as EB=Efficiency×BandwidthEB

## 7. Practical Considerations

- **Use in Noisy Channels:**
  - Go-Back-N ARQ is suitable for use in noisy channels where errors are frequent. However, it can lead to inefficiency if the error rate is high, as many frames may need to be retransmitted.
- **Bandwidth Utilization:**
  - In scenarios with high error rates, Go-Back-N may lead to poor bandwidth utilization due to the need to retransmit large sequences of frames.



### Explanation for Go-Back-N ARQ:

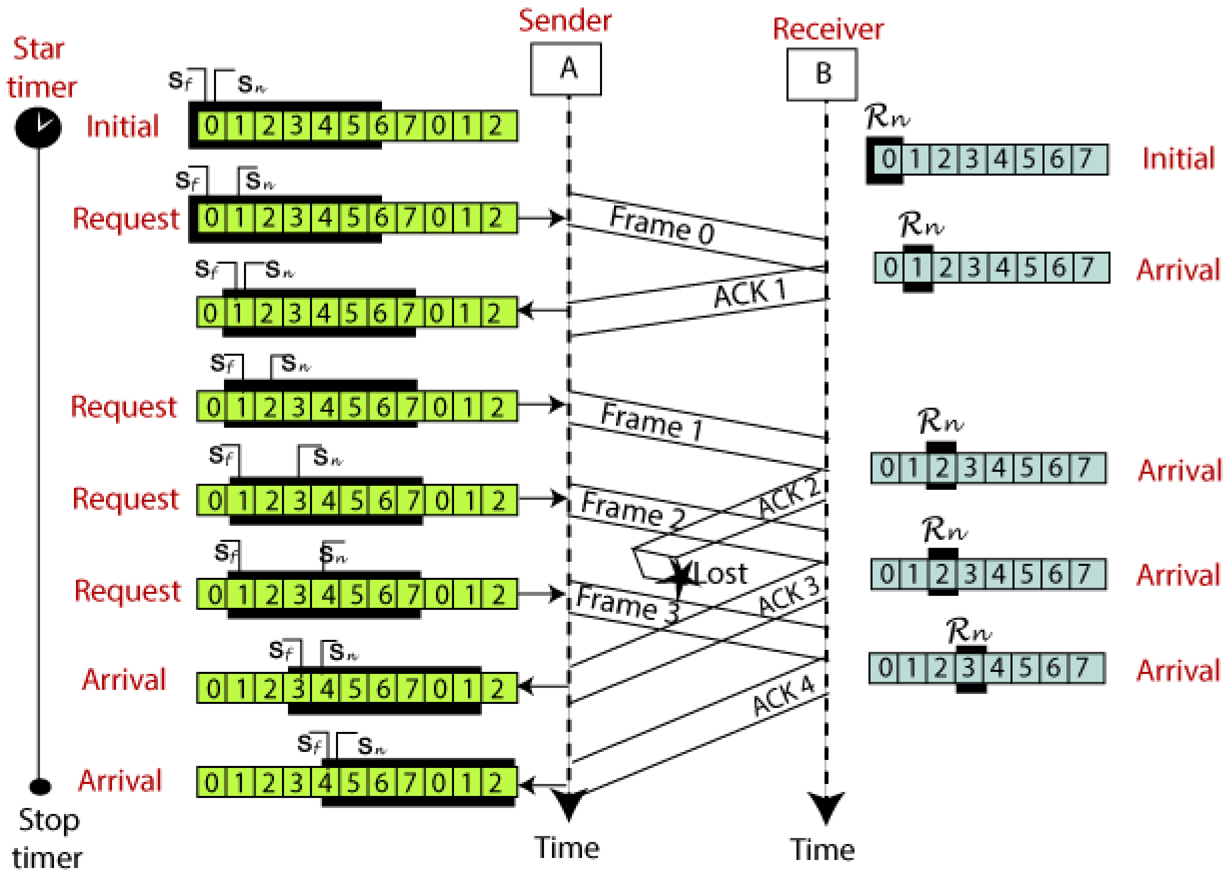
1. **Window Size $W_s = N$, $W_r = 1$:**
   - Here, $W_s$ represents the sender's window size, and $W_r$ is the receiver's window size.
   - $N$ is the maximum number of packets that can be sent without waiting for an acknowledgment.
   - The sequence number $\text{Seq} = N + 1$ means the number of distinct sequence numbers required is $N + 1$.
   - The number of bits required to represent these sequence numbers is $\text{bits} = \lceil \log_2(N + 1) \rceil$.
2. **Sequence Number $\text{Seq} = N$:**
   - The sender's window size is $W_s = N - 1$, and the receiver's window size is $W_r = 1$.
   - For instance, if $N = 4$, then $W_s = 3$ and $W_r = 1$. This configuration allows the sender to send up to 3 frames before waiting for an acknowledgment.
3. **General Case:**
   - If the number of bits used to represent the sequence numbers is $k$, then the sequence number range is $\text{Seq} = 2^k$.
   - The sender's window size $W_s = 2^k - 1$, and the receiver's window size $W_r = 1$.

**Conclusion**

Go-Back-N ARQ is a robust protocol that balances efficiency and reliability, making it suitable for a range of communication environments. Understanding its window management, error handling, and efficiency metrics is crucial for optimizing data transfer in networked systems.

**Summary:**

Go-Back-N ARQ is effective for improving data transfer efficiency by allowing multiple frames in transit before needing acknowledgments. However, it may be less efficient in environments with high error rates, as errors necessitate retransmission of multiple frames. Understanding the protocol's operation and its impact on performance helps in optimizing data communication in various network scenarios.



Working on Go Back N

**Walkthrough of the Diagram:**

- **Initial State**:
  - Both sender A and receiver B have their initial sequence numbers. The sender's window includes frames 0 to 7.
  - The receiver's window is expecting the first frame, which is frame 0.
- **Sending Frames**:
  - **Frame 0** is sent by A and correctly received by B, which sends ACK 1 (indicating that it expects frame 1 next).
  - **Frame 1** is sent and ACK 2 is received.
  - **Frame 2** is sent, but here, let's assume that the frame is lost (as shown in the diagram with a cross mark).
- **Loss and Retransmission**:
  - Despite the loss of frame 2, the sender A continues to send frame 3.
  - The receiver B does not acknowledge frame 3 because it was expecting frame 2, not frame 3.
  - After a timeout or upon receiving an ACK that indicates a missing frame, sender A realizes that frame 2 was lost.
  - As per the Go-Back-N protocol, sender A must go back to frame 2 and retransmit frame 2 and all subsequent frames (even though frame 3 was successfully transmitted earlier).
- **Resynchronization**:
  - Once the lost frame (frame 2) and all subsequent frames are retransmitted and acknowledged, the sender and receiver windows resynchronize.

## Selective Repeat (SR) ARQ Protocol

● Selective Repeat Automatic Repeat reQuest (SR ARQ) is a sliding window protocol that improves upon Go-Back-N ARQ by allowing the receiver to accept and acknowledge out-of-order frames. This protocol ensures reliable and efficient data transmission, especially in networks with a high error rate. SR ARQ is particularly effective in maximizing the use of available bandwidth by reducing unnecessary retransmissions.

### Key Concepts

1. **Window Size:**

○ **Sender's Window Size (N):** In SR ARQ, the sender's window size is N = $2^{(n-1)}$, where n is the number of bits in the sequence number. The window size is usually half the range of sequence numbers to avoid ambiguity in frame identification.

○ **Receiver's Window Size (N):** The receiver's window size is the same as the sender's, allowing it to store out-of-order frames until the missing ones arrive.

2. **Range of Sequence Numbers:**

○ The sequence numbers in SR ARQ range from 0 to $2^{n-1,}$ where n is the number of bits used. This cyclical sequence numbering helps manage frames efficiently within the sliding window.

### Sender's Operation

1. **Transmission:**

○ The sender can transmit up to N frames (where N is the window size) without waiting for an acknowledgment. Each frame is assigned a unique sequence number within the range.

○ After sending a frame, the sender continues to transmit additional frames within its window until the window is full or until it receives an acknowledgment for the earliest unacknowledged frame.

2. **Window Sliding:**

○ The sender's window slides forward each time an acknowledgment (ACK) is received for the earliest outstanding frame. This allows the sender to send new frames while retaining unacknowledged frames within the window.

### Receiver's Operation

1. **Frame Reception:**

○ The receiver can accept frames that arrive out of order and store them in a buffer until the missing frames arrive. The receiver's window tracks which frames have been received and which are still pending.

2. **Independent Acknowledgment:**

○ In SR ARQ, acknowledgments are sent independently for each frame. Each ACK contains the sequence number of the frame being acknowledged, allowing the sender to identify exactly which frames have been successfully received.

○ This independent acknowledgment system allows the sender to retransmit only the frames that were lost or corrupted, improving efficiency.

3. **Window Sliding:**

○ The receiver's window slides forward as frames are delivered to the higher layers. This allows the receiver to continue accepting new frames while waiting for any missing frames.

**Error Handling**

1. **Timeout and Retransmission:**
   - The sender sets a timer for each frame it sends. If an acknowledgment is not received before the timer expires, the sender retransmits only the specific frame that was not acknowledged.
2. **ACK Timer:**
   - The ACK timer ensures that the sender does not wait indefinitely for an acknowledgment. If the timer expires without receiving an ACK, the sender retransmits the corresponding frame.
3. **Selective Retransmission:**
   - Unlike Go-Back-N, SR ARQ only retransmits the frames that are lost or corrupted. This selective retransmission mechanism reduces the number of unnecessary retransmissions, making SR ARQ more efficient.

4. **Negative Acknowledgment (NAK)** is used to indicate that a specific frame has been received with errors or has been lost and needs retransmission.Receiver sends NAK for problematic frames; sender retransmits only those frames.

**Efficiency and Formula**

1. **Efficiency Formula:**
   - The efficiency of SR ARQ is higher than that of Go-Back-N ARQ because it avoids unnecessary retransmissions. The efficiency of SR ARQ can be expressed as:

$$\text{Efficiency} = \frac{N}{1+2a}$$

Where:

- $N$ is the sender's window size.
- $a = \frac{T_p}{T_t}$, with $T_p$ being the propagation delay and $T_t$ being the transmission delay.

   - The same formula applies as in Go-Back-N, but SR ARQ typically results in higher efficiency in practice due to its selective retransmission mechanism.
2. **Throughput:**
   - The throughput of SR ARQ is generally higher than that of Go-Back-N ARQ because it minimizes the number of retransmissions. The throughput can be calculated as Throughput = Efficiency×Bandwidth
   - Throughput is maximized when the sender can continuously send frames without waiting for acknowledgments and without retransmitting entire windows of frames.

**Advantages and Disadvantages**

- **Advantages:**
  - Higher efficiency and throughput compared to Go-Back-N ARQ, especially in networks with higher error rates.
  - Better utilization of bandwidth due to selective retransmission, reducing the need to resend entire windows of frames.
- **Disadvantages:**
  - More complex to implement due to the need for additional buffering and management of out-of-order frames.

○ Requires more memory at the receiver to store out-of-order frames, which may be a concern in memory-constrained environments.

**Practical Considerations**

**1. Use in Noisy Channels:**

- **Efficient Error Handling:** Handles errors more effectively by retransmitting only specific erroneous frames.
- **Buffering Requirements:** Requires significant buffering at both sender and receiver to handle out-of-order frames.
- **Complexity:** More complex to implement compared to Stop-and-Wait ARQ due to the management of multiple frame states and acknowledgments.

**2. Bandwidth Utilization:**

- **Improved Utilization:** Maximizes the use of available bandwidth by allowing multiple frames to be in transit.
- **Effective in High Bandwidth-Delay Products:** Performs well in networks with high bandwidth and long propagation delays due to its ability to keep multiple frames in the pipeline.
- **Reduced Idle Time:** Minimizes idle times on the link by enabling continuous frame transmission.

Selective Repeat ARQ provides an efficient method for managing frame transmission in networks where errors are common, and order needs to be preserved while balancing complexity and performance.

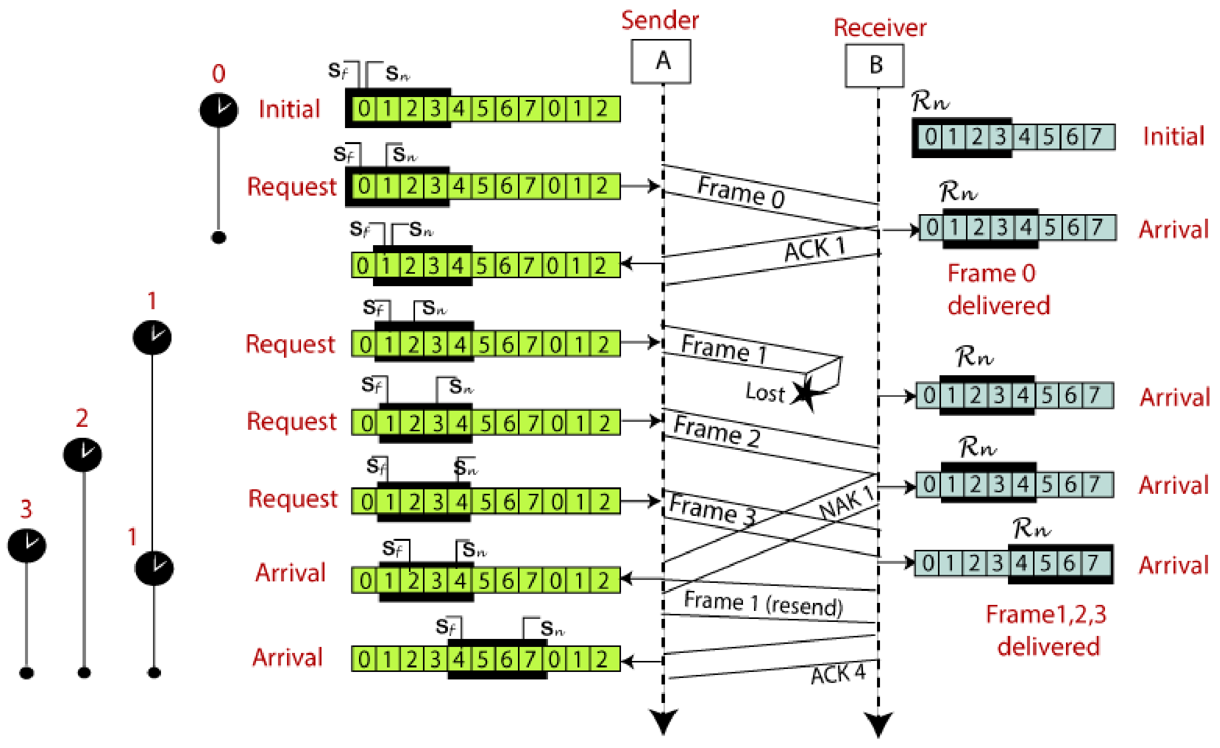


Diagram to explain how SR works

**Explanation of Selective Repeat (SR) in Sliding Window Protocol From Above Diagram**

The Selective Repeat (SR) protocol is another type of sliding window protocol that allows for reliable data transmission. Unlike Go-Back-N, SR only retransmits frames that were received in error or lost, rather than retransmitting all subsequent frames.

**Key Concepts:**

1. **Sliding Window**:
   - Both the sender and receiver maintain a window, but in SR, the receiver can accept out-of-order frames and buffer them until the missing frames are received.
   - The sender's window has a size N, meaning it can send up to N frames before needing an acknowledgment.
2. **Sequence Numbers**:
   - Like in GBN, frames are assigned sequence numbers (0, 1, 2, …).
   - The receiver uses these sequence numbers to identify and reorder frames correctly.
3. **Acknowledgment (ACK) and Negative Acknowledgment (NAK)**:
   - The receiver sends an ACK for each frame received correctly.
   - If a frame is lost or received in error, the receiver can send a Negative Acknowledgment (NAK) to prompt the sender to resend that specific frame.

**Walkthrough of the Diagram:**

- **Initial State**:
  - Sender A starts with the ability to send frames 0 through 7 (its window).
  - Receiver B is initially expecting frame 0.
- **Sending Frames**:
  - **Frame 0** is sent by A and correctly received by B, which sends ACK 1, indicating that frame 0 was received correctly.
  - **Frame 1** is then sent by A, but this frame is **lost** during transmission.
- **Handling Frame Loss**:
  - Sender A continues by sending **Frame 2**, which is correctly received and acknowledged by B with ACK 3 (acknowledging the next expected frame).
  - Similarly, **Frame 3** is sent and correctly received, prompting ACK 4 from the receiver.
- **Negative Acknowledgment (NAK)**:
  - Since Frame 1 was lost, the receiver detects that it is missing. It responds with **NAK 1**, asking the sender to resend frame 1.
  - The sender then retransmits **Frame 1**.
- **Frame Delivery**:
  - Once Frame 1 is retransmitted and received correctly, the receiver updates its window to expect the next set of frames.
  - With all frames (Frame 0, 1, 2, and 3) now successfully delivered, the receiver sends **ACK 4**.

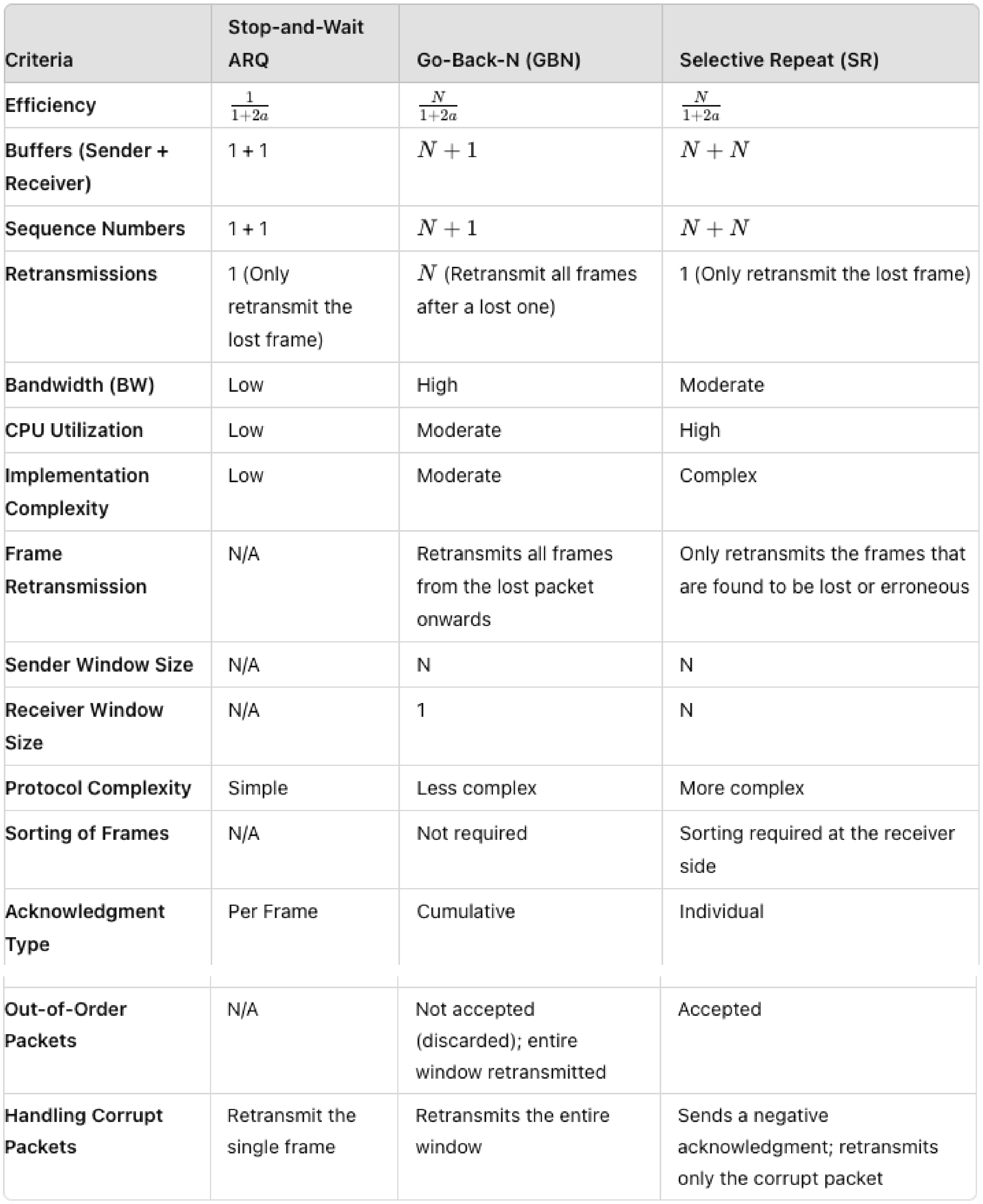
**Summary:**

- **Selective Repeat** is more efficient than Go-Back-N because it only requires the retransmission of frames that are lost or erroneous, not all subsequent frames.

- The receiver has the ability to store out-of-order frames and process them once the missing frames are received.

This diagram illustrates how SR handles the loss of a frame (Frame 1) without needing to retransmit subsequent frames (Frame 2 and Frame 3). Instead, it only requests the missing frame using NAK, and once it is received, all frames are correctly delivered.

# Comparison of the Stop-and-Wait ARQ, Go-Back-N (GBN), and Selective Repeat (SR) protocols



| Criteria | Stop-and-Wait ARQ | Go-Back-N (GBN) | Selective Repeat (SR) |
|---|---|---|---|
| Efficiency | $\frac{1}{1+2a}$ | $\frac{N}{1+2a}$ | $\frac{N}{1+2a}$ |
| Buffers (Sender + Receiver) | 1 + 1 | $N+1$ | $N+N$ |
| Sequence Numbers | 1 + 1 | $N+1$ | $N+N$ |
| Retransmissions | 1 (Only retransmit the lost frame) | $N$ (Retransmit all frames after a lost one) | 1 (Only retransmit the lost frame) |
| Bandwidth (BW) | Low | High | Moderate |
| CPU Utilization | Low | Moderate | High |
| Implementation Complexity | Low | Moderate | Complex |
| Frame Retransmission | N/A | Retransmits all frames from the lost packet onwards | Only retransmits the frames that are found to be lost or erroneous |
| Sender Window Size | N/A | N | N |
| Receiver Window Size | N/A | 1 | N |
| Protocol Complexity | Simple | Less complex | More complex |
| Sorting of Frames | N/A | Not required | Sorting required at the receiver side |
| Acknowledgment Type | Per Frame | Cumulative | Individual |
| Out-of-Order Packets | N/A | Not accepted (discarded); entire window retransmitted | Accepted |
| Handling Corrupt Packets | Retransmit the single frame | Retransmits the entire window | Sends a negative acknowledgment; retransmits only the corrupt packet |

## SOLVED EXAMPLES

**Problem 01:**

If the bandwidth of the line is 1.5 Mbps, RTT is 45 msec, and the packet size is 1 KB, then find the link utilization in stop-and-wait.

**Solution:**

**Given:**

- Bandwidth = 1.5 Mbps
- RTT = 45 msec
- Packet size = 1 KB

**Calculating Transmission Delay:**

Transmission delay (Tt) = Packet size / Bandwidth
= 1 KB / 1.5 Mbps
= (2^10 × 8 bits) / (1.5 × 10^6 bits per sec)
= 5.461 msec

**Calculating Propagation Delay:**

Propagation delay (Tp) = Round Trip Time / 2
= 45 msec / 2
= 22.5 msec

**Calculating Value Of 'a':**

a = Tp / Tt
a = 22.5 msec / 5.461 msec
a = 4.12

**Calculating Link Utilization:**

Link Utilization or Efficiency (η) = 1 / (1 + 2a)
= 1 / (1 + 2 × 4.12)
= 1 / 9.24
= 0.108
= 10.8 %

---

**Problem 02:**

A channel has a bit rate of 4 Kbps and a one-way propagation delay of 20 msec. The channel uses stop-and-wait protocol. The transmission time of the acknowledgment frame is negligible. To get a channel efficiency of at least 50%, the minimum frame size should be:

- 80 bytes
- 80 bits
- 160 bytes
- 160 bits

**Solution:**

**Given:**

- Bandwidth = 4 Kbps
- Propagation delay (Tp) = 20 msec
- Efficiency $\geq$ 50%

Let the required frame size = L bits.

**Calculating Transmission Delay:**

Transmission delay (Tt) = Packet size / Bandwidth
= L bits / 4 Kbps

**Calculating Value Of 'a':**

a = Tp / Tt
a = 20 msec / (L bits / 4 Kbps)
a = (20 msec × 4 Kbps) / L bits

**Condition For Efficiency To Be At Least 50%:**

For efficiency to be at least 50%, we must have:
$1 / (1 + 2a) \geq 1/2$
$a \leq 1/2$

Substituting the value of 'a', we get:
(20 msec × 4 Kbps) / L bits $\leq$ 1/2
L bits $\geq$ (20 msec × 4 Kbps) × 2
L bits $\geq$ (20 × 10^-3 sec × 4 × 10^3 bits per sec) × 2
L bits $\geq$ 20 × 4 bits × 2
$L \geq 160$

From here, the frame size must be at least 160 bits.

Thus, the correct option is (D).

**Problem 03:**

What is the throughput achievable in stop-and-wait protocol by a maximum packet size of 1000 bytes and a network span of 10 km? Assume the speed of light in cable is 70% of the speed of light in a vacuum.

**Solution:**

**Given:**

- Packet size (L) = 1000 bytes
- Distance (d) = 10 km = $10^4$ m
- Speed (v) = 70% of $3 \times 10^8$ m/sec = $2.1 \times 10^8$ m/sec

**Throughput Calculation:**

Throughput = L / [2 × d / v]
= 1000 bytes / [2 × $10^4$ m / ($2.1 \times 10^8$ m/sec)]
= $1.05 \times 10^7$ bytes per sec
= 10.5 MBps

**Problem 04**

**Question:**
If the packet size is 1 KB and propagation time is 15 ms, and the channel capacity is $10^9$ bps, then find the transmission time and utilization of the sender in the stop-and-wait protocol.

**Solution:**
Formula: Transmission Time = (Packet Size) / (Channel Capacity)

Packet Size: 1 KB = 1024 Bytes = 1024 * 8 bits = 8192 bits

Channel Capacity: $10^9$ bps (bits per second)

Calculation:

Transmission Time = 8192 bits / $10^9$ bps = $8.192 \times 10^{-6}$ seconds = 8.192 microseconds

Sender Utilization in Stop-and-Wait Protocol:Formula:

Utilization (U) = Transmission Time / (Transmission Time + 2 × Propagation Time)

Transmission Time: 8.192 microseconds

Propagation Time: 15 milliseconds = 15000 microseconds

Calculation:

$U = 8.192 / (8.192 + 2 \times 15000) = 8.192 / 30008.192 \approx 0.000273$

So, the sender utilization is approximately 0.0273%.

---

## Problem 05

**Question:**
Consider a MAN with an average source and destination 20 km apart and a one-way delay of 100 µs. At what data rate does the round-trip delay equal the transmission delay for a 1 KB packet?

**Solution:**
Given:

- Distance = 20 km
- Propagation delay (Tp) = 100 µs
- Packet size = 1 KB

**We need to have:**

Round Trip Time = Transmission delay

2 x Propagation delay = Transmission delay

**Substituting the values:**

2 x 100 µs = 1 KB / Bandwidth

Bandwidth = 1 KB / 200 µs

Bandwidth = (2^10 bytes / 200 x 10^-6 seconds)

Bandwidth = 5.12 MBps or 40.96 Mbps

---

## Problem 06

**Question:**
Consider two hosts X and Y connected by a single direct link of rate 10^6 bps. The distance between the two hosts is 10,000 km, and the propagation speed along the link is 2 x 10^8 m/s. Host X sends a file of 50,000 bytes as one large message to host Y continuously. Let the transmission and propagation delays be p milliseconds and q milliseconds, respectively. Then the value of p and q are:

- (a) p = 50 and q = 100
- (b) p = 50 and q = 400

- (c) p = 100 and q = 50
- (d) p = 400 and q = 50

**Solution:**
Given:

- Bandwidth = 10^6 bps
- Distance = 10,000 km
- Propagation speed = 2 x 10^8 m/s
- Packet size = 50,000 bytes

**Calculating Transmission Delay (Tt):**

Tt = Packet size / Bandwidth

= 50,000 bytes / 10^6 bps

= (5 x 10^4 x 8 bits) / 10^6 bps

= 0.4 sec

= 400 ms

**Calculating Propagation Delay (Tp):**

Tp = Distance / Propagation speed

= 10,000 km / (2 x 10^8 m/s)

= 10^7 meters / (2 x 10^8 m/s)

= 50 ms

Thus, Option (D) is correct: **p = 400 ms and q = 50 ms**.

---

## Problem 07

**Question:**
The values of parameters for the stop-and-wait ARQ protocol are as given below:

- Bit rate of the transmission channel = 1 Mbps
- Propagation delay from sender to receiver = 0.75 ms
- Time to process a frame = 0.25 ms
- Number of bytes in the information frame = 1980
- Number of bytes in the acknowledgment frame = 20
- Number of overhead bytes in the information frame = 20

Assume that there are no transmission errors. Then the transmission efficiency (in %) of the stop-and-wait ARQ protocol for the above parameters is ___________. (correct to 2 decimal places)

**Solution:**
Given:

Step 1: Transmission Time for the Information Frame

- Number of bytes in the information frame = 1980 bytes.
- Number of overhead bytes in the information frame = 20 bytes.
- Total size of the information frame = 1980 + 20 = 2000 bytes.
- Convert bytes to bits: 2000 × 8 = 16000 bits.
- Bit rate = 1 Mbps = 1 × 10^6 bits/second.

Now, the transmission time for the information frame:
T_info = 16000 bits ÷ 1 × 10^6 bits/sec = 0.016 seconds = 16 ms.

Step 2: Transmission Time for the Acknowledgment (ACK) Frame

- Number of bytes in the acknowledgment frame = 20 bytes.
- Convert bytes to bits: 20 × 8 = 160 bits.

Now, the transmission time for the acknowledgment frame:
T_ack = 160 bits ÷ 1 × 10^6 bits/sec = 0.00016 seconds = 0.16 ms.

Step 3: Total Round-Trip Delay

- Propagation delay from sender to receiver = 0.75 ms.
- Total propagation delay (round-trip) = 2 × 0.75 = 1.5 ms.
- Processing delay at the receiver for each frame = 0.25 ms.

Now, the total delay:
D_total = 16 ms (Tx time of info) + 0.16 ms (Tx time of ACK) + 1.5 ms (RTT) + 0.25 ms (processing delay) = 17.91 ms.

Step 4: Efficiency Calculation
Efficiency is the ratio of the useful transmission time to the total time, expressed as a percentage. The useful time is the time spent transmitting the information frame.
Efficiency = (T_info ÷ D_total) × 100 = (16 ms ÷ 17.91 ms) × 100 = 89.38%.
Therefore, the transmission efficiency of the Stop-and-Wait ARQ protocol is 89.38%.

---

## Problem 08

**Question:**
A sender uses the stop-and-wait ARQ protocol for the reliable transmission of frames. Frames are of size 1000 bytes and the transmission rate at the sender is 80 Kbps. The size of an acknowledgment is 100 bytes, and the transmission rate at the receiver is 8 Kbps. The one-way propagation delay is 100 ms. Assuming no frame is lost, the sender throughput is __________ bytes/sec.

**Solution:**
Given:

- Frame size = 1000 bytes
- Sender bandwidth = 80 Kbps
- Acknowledgment size = 100 bytes
- Receiver bandwidth = 8 Kbps
- Propagation delay (Tp) = 100 ms

**Calculating Transmission Delay of Data Frame:**

Transmission delay (Tt)

= Frame size / Sender bandwidth

= 1000 bytes / 80 Kbps

= (1000 x 8 bits) / (80 x 10^3 bits per second)

= 100 ms

**Calculating Transmission Delay of Acknowledgment:**

Transmission delay (Tt)

= Acknowledgment size / Receiver bandwidth

= 100 bytes / 8 Kbps

= (100 x 8 bits) / (8 x 10^3 bits per second)

= 100 ms

**Calculating Useful Time:**

Useful Time

= Transmission delay of data frame

= 100 ms

**Calculating Total Time:**

Total Time

= Transmission delay of data frame + Propagation delay of data frame + Transmission delay of acknowledgment + Propagation delay of acknowledgment

= 100 ms + 100 ms + 100 ms + 100 ms

= 400 ms

**Calculating Efficiency (η):**

η = Useful time / Total time

= 100 ms / 400 ms

= 1 / 4

= 25%

**Calculating Sender Throughput:**

Sender throughput

= Efficiency (η) x Sender bandwidth

= 0.25 x 80 Kbps

= 20 Kbps

= (20 x 1000 / 8) bytes per second

= **2500 bytes/sec**

---

**Problem 09**

**Question:**
Using the stop-and-wait protocol, a sender wants to transmit 10 data packets to the receiver. Out of these 10 data packets, every 4th data packet is lost. How many packets will the sender have to send in total?

**Solution:**
Draw a timeline diagram and analyze.

The packets will be sent as:

1, 2, 3, 4, 4, 5, 6, 7, 7, 8, 9, 10, 10

The lost packets are: 4, 7, and 10.

Thus, the sender sends 13 packets in total.

---

**Problem 10**

**Question:**
A sender is using the stop-and-wait protocol with a round-trip time (RTT) of 100 ms and a bandwidth of 10 Mbps. If the packet size is 1500 bytes, what is the sender's utilization?

**Solution:**
Given:

- RTT = 100 ms
- Bandwidth = 10 Mbps
- Packet size = 1500 bytes

**Calculating Transmission Delay:**

Transmission delay (Tt)

= Packet size / Bandwidth

= 1500 bytes / 10 Mbps

= (1500 x 8 bits) / (10 x 10^6 bits per second)

= 1.2 ms

**Calculating Utilization (η):**

Utilization (η)

= Tt / RTT

= 1.2 ms / 100 ms

= 0.012

= **1.2%**

**Problem-11:**

If the transmission delay and propagation delay in a sliding window protocol are 1 msec and 49.5 msec, respectively, then:

1. What should be the sender window size to get the maximum efficiency?

2. What is the minimum number of bits required in the sequence number field?

3. If only 6 bits are reserved for sequence numbers, what will be the efficiency?

Solution:

Given:

- Transmission delay = 1 msec
- Propagation delay = 49.5 msec

1. Calculating the sender window size for maximum efficiency:

    - To get maximum efficiency, the sender window size should be 1 + 2a, where a is the ratio of propagation delay (Tp) to transmission delay (Tt).
    - a = Tp / Tt = 49.5 msec / 1 msec = 49.5
    - Sender window size = 1 + 2a = 1 + 2 * 49.5 = 1 + 99 = 100
2. Thus, for maximum efficiency, the sender window size = 100.
3. Calculating the minimum number of bits required in the sequence number field:
    - The minimum number of bits required in the sequence number field is given by the ceiling of log2(1 + 2a).
    - Minimum number of bits = log2(1 + 2a) = log2(100) = approximately 6.8
    - Therefore, the minimum number of bits required = 7 (since 6.8 is rounded up to the nearest whole number).
4. Calculating the efficiency with 6 bits reserved for sequence numbers:
    - If only 6 bits are reserved, the maximum sequence numbers possible is 2^6 = 64.
    - Efficiency = (Sender window size in the protocol) / (Optimal sender window size) = 64 / 100 = 0.64 or 64%

---

**Problem-12:**

If the transmission delay and propagation delay in a sliding window protocol are 1 msec and 99.5 msec, respectively, then:

1. What should be the sender window size to get the maximum efficiency?

2. What is the minimum number of bits required in the sequence number field?

3. If only 7 bits are reserved for sequence numbers, what will be the efficiency?

Solution:

Given:

- Transmission delay = 1 msec
- Propagation delay = 99.5 msec

1. Calculating the sender window size for maximum efficiency:
    - a = Tp / Tt = 99.5 msec / 1 msec = 99.5
    - Sender window size = 1 + 2a = 1 + 2 * 99.5 = 1 + 199 = 200
2. Thus, for maximum efficiency, the sender window size = 200.
3. Calculating the minimum number of bits required in the sequence number field:
    - Minimum number of bits = log2(1 + 2a) = log2(200) = approximately 7.64
    - Therefore, the minimum number of bits required = 8.
4. Calculating the efficiency with 7 bits reserved for sequence numbers:
    - If only 7 bits are reserved, the maximum sequence numbers possible is 2^7 = 128.

   - Efficiency = (Sender window size in the protocol) / (Optimal sender window size) = 128 / 200 = 0.64 or 64%

---

**Problem-13:**

Consider a 128 × 10^3 bits/sec satellite communication link with a one-way propagation delay of 150 msec. The Selective Retransmission (Repeat) protocol is used on this link to send data with a frame size of 1 KB. Neglect the transmission time of acknowledgment.

1. What is the minimum number of bits required for the sequence number field to achieve 100% utilization?

Solution:

Given:

- Bandwidth = 128 × 10^3 bits/sec
- Propagation delay = 150 msec
- Frame size = 1 KB

To achieve 100% utilization, the efficiency must be 100%. This is when the sender window size is optimal (1 + 2a).

1. Calculating Transmission Delay:
   - Transmission delay (Tt) = Frame size / Bandwidth
   - Tt = (1 KB) / (128 × 10^3 bits/sec)
   - Tt = (1 × 2^10 × 8 bits) / (128 × 10^3 bits/sec)
   - Tt = 64 msec
2. Calculating the value of 'a':
   - a = Tp / Tt = 150 msec / 64 msec = 2.34
3. Calculating Optimal Sender Window Size:
   - Optimal sender window size = 1 + 2a = 1 + 2 * 2.34 = 5.68, which rounds up to 6
4. Calculating the Number of Sequence Numbers Required:
   - In the SR Protocol, the sender window size and receiver window size are the same.
   - Sender window size = receiver window size = 6
   - For any sliding window protocol, the minimum number of sequence numbers required is the sum of the sender window size and receiver window size: 6 + 6 = 12
5. Calculating the Bits Required in the Sequence Number Field:
   - To have 12 sequence numbers, the minimum number of bits required in the sequence number field = log2(12)
   - This results in approximately 4 bits.

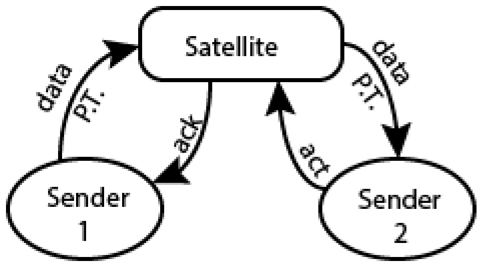
Thus:

Minimum number of bits required in the sequence number field = 4

**Problem-15:**

A 1 Mbps satellite link connects two ground stations. The altitude of the satellite is 36.504 km and speed of the signal is 3×103m/s. What should be the packet size for a channel utilization of 25% for a satellite link using go-back-127 sliding window protocol?

Solution:



Given:

So propagation delay = Propagation from S1 to S + Propagation from S + S2

=P.T+P.T

=2 P.T

Propagation delay

(TP)=DistanceSpeed

=2×36504×103m3×10−3msec

=24336×10−5sec

=243360μsec

Transmission delay

Tt=x μ sec

α=Tp / Tt=243360 / x

Efficiency

η= 1 / 1+2 a

$\Rightarrow 0.25 = 127x\ / x + 2 \times 243360$

$\Rightarrow x = 960$ bits=120Bytes

---

**Problem-16:**

Consider a network connecting two systems located 8,000 km apart. The bandwidth of the network is 500 Mbps. The propagation speed of the media is 4×1064 \times 10^64×106 m/sec. It is needed to design a Go-Back-N sliding window protocol for this network. The average packet size is 1,007 bits. The network is to be used to its full capacity.

Assume that processing delays at nodes are negligible. What is the minimum size in bits of the sequence number field?

Solution:

Given:

- Distance = 8,000 km
- Bandwidth = 500 Mbps
- Propagation speed = 4×1064 \times 10^64×106 m/sec
- Packet size = 1,007 bits

For full capacity usage, Efficiency ($\eta$) = 1 Efficiency = 1 when sender window size = 1 + 2a

1. Calculate Transmission Delay (Tt):

Transmission delay = Packet size / Bandwidth = 1,007 bits / 500 Mbps = 1,007 / ($500 \times 10^6$) sec = $2.014 \times 10^{-6}$ sec = 2.014 μsec

2. Calculate Propagation Delay (Tp):

Propagation delay = Distance / Speed = 8,000 km / ($4 \times 10^6$ m/sec) = $8{,}000 \times 10^3$ m / ($4 \times 10^6$ m/sec) = 2 sec

3. Calculate Value of 'a':

a = Propagation delay / Transmission delay = 2 sec / 2.014 μsec = 2,000,000 μsec / 2.014 μsec ≈ 991,072.58

4. Calculate Sender Window Size:

Sender window size = 1 + 2a = 1 + 2 × 991,072.58 ≈ 1,982,146.16

5. Calculate Minimum Size of Sequence Number Field:

Minimum number of bits required = $\lceil \log_2(1 + 2a) \rceil = \lceil \log_2(1{,}982{,}146) \rceil$ = 21 bits

Thus, the minimum size of the sequence number field is 21 bits.

**Problem-17:**

Station A needs to send a message consisting of 9 packets to station B using a sliding window (window size 3) and go back n error control strategy. All packets are ready and immediately available for transmission.

If every 5th packet that A transmits gets lost (but no ACKs from B ever get lost), then what is the number of packets that A will transmit for sending the message to B?

**Given:**

- **Total Number of Packets to be Sent:** 9
- **Protocol Used:** Go-Back-N
- **Window Size (N):** 3
- **Packet Loss Condition:** Every 5th packet gets lost

---

**Step 1: Initial Transmission**

- **Action:** Sender transmits packets 1, 2, and 3.
- **Packets Sent:** 3 (1, 2, 3)

---

**Step 2:**

- **Condition:** Acknowledgment (ACK) received for packet 1.
- **Action:** Sender slides the window and transmits packet 4.
- **Total Packets Sent:** 4 (1, 2, 3, 4)

---

**Step 3:**

- **Condition:** ACK received for packet 2.
- **Action:** Sender slides the window and transmits packet 5.
- **Total Packets Sent:** 5 (1, 2, 3, 4, 5)

---

**Step 4:**

- **Condition:** ACK received for packet 3.
- **Action:** Sender slides the window and transmits packet 6.
- **Total Packets Sent:** 6 (1, 2, 3, 4, 5, 6)

---

**Step 5:**

- **Condition:** ACK received for packet 4.
- **Action:** Sender slides the window and transmits packet 7.
- **Total Packets Sent:** 7 (1, 2, 3, 4, 5, 6, 7)

---

**Step 6: Packet Loss and Retransmission**

- **Condition:** Packet 5 is lost (as per the packet loss condition). Timeout occurs.
- **Action:** Sender retransmits packet 5, and due to Go-Back-N, packets 6 and 7 are also retransmitted.
- **Total Packets Sent:** 10 (1, 2, 3, 4, 5, 6, 7, 5, 6, 7)

---

**Step 7:**

- **Condition:** ACK received for packet 5.
- **Action:** Sender slides the window and transmits packet 8.
- **Total Packets Sent:** 11 (1, 2, 3, 4, 5, 6, 7, 5, 6, 7, 8)

---

**Step 8:**

- **Condition:** ACK received for packet 6.
- **Action:** Sender slides the window and transmits packet 9.
- **Total Packets Sent:** 12 (1, 2, 3, 4, 5, 6, 7, 5, 6, 7, 8, 9)

---

**Step 9: Packet Loss and Retransmission**

- **Condition:** Packet 7 is lost (as per the packet loss condition). Timeout occurs.
- **Action:** Sender retransmits packet 7 and the subsequent packets (8, 9).
- **Total Packets Sent:** 15 (1, 2, 3, 4, 5, 6, 7, 5, 6, 7, 8, 9, 7, 8, 9)

---

**Step 10:**

- **Condition:** ACK received for packet 7.
- **Action:** Sender slides the window.
- **Total Packets Sent:** 15 (no new packets sent, count remains the same)

---

**Step 11:**

- **Condition:** ACK received for packet 8.
- **Action:** Sender slides the window.
- **Total Packets Sent:** 15 (no new packets sent, count remains the same)

---

**Step 12: Final Packet Loss and Retransmission**

- **Condition:** Packet 9 is lost (as per the packet loss condition). Timeout occurs.
- **Action:** Sender retransmits packet 9.
- **Total Packets Sent:** 16 (1, 2, 3, 4, 5, 6, 7, 5, 6, 7, 8, 9, 7, 8, 9, 9)

---

**Summary:**

- **Total Packets Sent:** 16
- **Successful Transmission:** All 9 packets successfully transmitted.

**Problem-18:**

In Go-Back-N (GBN) with a window size of 4, if every 6th packet is lost and we need to send 20 packets, the number of transmissions required is? ( ANS: **22**.)

## Problem-19:

**Problem:**

In GB4, if every 6th packet that is being transmitted is lost and we have to send 10 packets, then how many transmissions are required?

**Solution:**

1. **Go-Back-N (GBN)** retransmits a lost packet and all subsequent packets in the window.

### Problem Breakdown

1. **Initial Transmission:**
   - Packets 1 to 10 are sent.

    - Packet 6 is lost.
2. **Resend Due to Loss:**
    - Since packet 6 is lost, packets 6, 7, 8, and 9 need to be retransmitted. Packet 10 will be retransmitted in the next window.
3. **Transmission After Resend:**
    - Packets 6, 7, 8, and 9 are retransmitted.
4. **Next Loss Occurrence:**
    - After retransmitting packets 6, 7, 8, and 9, packet 8 is lost this time.
    - Packets 8, 9, and 10 need to be retransmitted.
5. **Total Count:**
    - Initial Transmission: 10 packets.
    - Retransmissions due to packet 6 loss: 4 packets (6, 7, 8, 9).
    - Retransmissions due to packet 8 loss: 3 packets (8, 9, 10).

### Total Transmissions Calculation

**Initial Transmission:**

- 10 packets (1 through 10)

**Retransmission due to packet 6 loss:**

- 4 packets (6, 7, 8, 9)

**Retransmission due to packet 8 loss:**

- 3 packets (8, 9, 10)

**Total Transmissions:**

- 10 (initial) + 4 (first retransmission) + 3 (second retransmission) = 17 transmissions

**Initial Transmission: 1 2 3 4 5 6 7 8 9 10**

**Retransmit (after 6 lost): 6 7 8 9**

**Retransmit (after 8 lost): 8 9 10**

---

## Problem-20:

**Problem:**

The maximum window size for data transmission using the Selective Repeat protocol with n bit frame sequence numbers is:

- 2^n
- 2^n - 1
- 2^(n-1)
- 2^(n-2)

**Solution:**

With n bits, the total number of sequence numbers possible is 2^n.

In the Selective Repeat (SR) Protocol, the sender window size and receiver window size are both equal to `W`.

For the protocol to work correctly, the number of available sequence numbers must be at least the sum of the sender and receiver window sizes:

2^n = W + W.

This simplifies to:

2^n = 2W,

Therefore,

W = 2^(n-1).

Thus, the maximum window size possible for both sender and receiver is 2^(n-1).

So, the correct option is 2^(n-1).

---

## Problem-21:

**Problem:**

In SR protocol, suppose frames 0 to 4 have been transmitted. Now, imagine that frame 0 times out, frame 5 (a new frame) is transmitted, frame 1 times out, frame 2 times out, and frame 6 (another new frame) is transmitted. What will be the outstanding packets in the sender's window?

- 341526
- 3405126
- 0123456
- 654321

**Solution:**

In the Selective Repeat (SR) Protocol, only the required frame is retransmitted, not the entire window.

1. Frames transmitted initially are: 4, 3, 2, 1, 0.
2. Frame 0 times out and is retransmitted: 0, 4, 3, 2, 1.
3. Frame 5 (a new frame) is transmitted: 5, 0, 4, 3, 2, 1.
4. Frame 1 times out and is retransmitted: 1, 5, 0, 4, 3, 2.
5. Frame 2 times out and is retransmitted: 2, 1, 5, 0, 4, 3.
6. Frame 6 (another new frame) is transmitted: 6, 2, 1, 5, 0, 4, 3.

Thus, the outstanding packets in the sender's window are 654321.

So, the correct option is 654321.

---

## Problem-22:

**Problem:**

The Selective Repeat protocol is similar to Go-Back-N except in the following way:

- Frame formats are similar in both protocols.
- The sender has a window defining the maximum number of outstanding frames in both protocols.
- Both use piggybacked acknowledgments where possible and do not acknowledge every frame explicitly.
- Both use a piggyback approach that acknowledges the most recently received frame.

**Solution:**

- **Option (A):** Both protocols use similar frame formats as they are sliding window protocols. The difference lies in their coding and implementation.
- **Option (B):** In both protocols, the sender has a window defining the maximum number of outstanding frames.
- **Option (C):** Both protocols use piggybacked acknowledgments where possible. Go-Back-N uses cumulative acknowledgments, while Selective Repeat acknowledges each frame independently.
- **Option (D):** Go-Back-N acknowledges the most recently received frame with cumulative acknowledgments, including acknowledgments for all previous frames. Selective Repeat acknowledges each frame independently.

Thus, the correct options are (C) and (D).

---

## Problem-23:

**Problem:**

Consider a 128 x 10^3 bits/sec satellite communication link with a one-way propagation delay of 150 ms. Selective Repeat (SR) protocol is used on this link to send data with a frame size of 1 KB. Neglect the transmission time of acknowledgments. The minimum number of bits required for the sequence number field to achieve 100% utilization is ________.

**Solution:**

- **Given:**
  - Bandwidth = 128 x 10^3 bits/sec
  - Propagation delay (Tp) = 150 ms
  - Frame size = 1 KB
- **Calculate Transmission Delay (Tt):**
  Transmission delay (Tt) = Frame size / Bandwidth
  Frame size = 1 KB = 8 x 10^3 bits.
  Tt = (8 x 10^3 bits) / (128 x 10^3 bits/sec) = 64 ms.
- **Calculate Value of a:**
  a = Tp / Tt
  a = 150 ms / 64 ms ≈ 2.34.
- **Calculate Optimal Sender Window Size:**
  Optimal sender window size = 1 + 2a
  Optimal sender window size = 1 + 2 x 2.34 = 5.68 ≈ 6.

- **Calculate Number of Sequence Numbers Required:**
  In SR Protocol, the sender window size and receiver window size are the same.
  Minimum number of sequence numbers required = Sender window size + Receiver window size
  Minimum number of sequence numbers required = 6 + 6 = 12.
- **Calculate Bits Required in Sequence Number Field:**
  To have 12 sequence numbers, minimum bits required = $\lceil \log2(12) \rceil$ = 4.

Thus, the minimum number of bits required in the sequence number field is 4.