REPORT

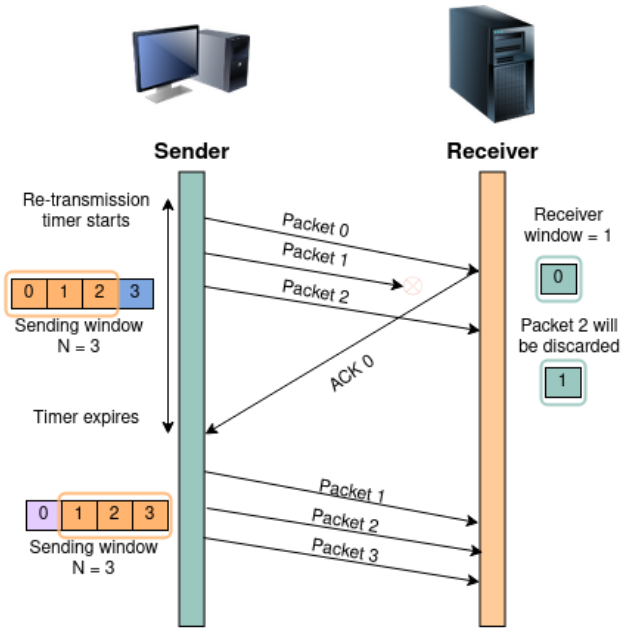
Go-Back-N Protocol: Efficiency Analysis

- ## Introduction

  This report focuses on the implementation of the GBN protocol simulation in C++, aiming to assess its efficiency and behaviour under various conditions. The simulation incorporates sliding window-based data transmission, packet loss simulation, timeout handling, and tracking of packet transmission times.

- ## Go-Back-N

  The Go-Back-N (GBN) protocol is a crucial sliding window-based approach used in data communication to ensure reliable data transfer over unreliable channels. It operates by allowing the sender to transmit multiple packets before receiving acknowledgments, enhancing throughput while managing potential packet loss.



- **Data Analysis :**

Step 1. Collecting data from the simulation, we collect the data of total data sent and total time taken for sending all data.

Step 2. Calculate Utilised bandwidth using formula

Utilised bandwidth = total data sent / total time taken

Step 3. Calculate Efficiency

Efficiency is the ratio of utilised bandwidth with total available bandwidth

Efficiency = Utilised Bandwidth / Available Bandwidth

- **Implementation overview**

Input parameters: The simulation incorporates user-defined parameters such as packet size, propagation time, timeout duration, window size, the total number of packets, packet loss probability, acknowledgment loss probability, and bandwidth considerations.

- **Results**

Experiment 1:

The first experiment is testing the effect of transmission window and packetloss rate has on efficiency when the bitrate is 100Mbps

Simulation of go-back-n with below input parameters:

Packet size = 1024 Bytes
Bandwidth = 100 mbps
Propagation Time = 0.5 ms
Total no of packet sent = 1000
Timeout = 2.6ms

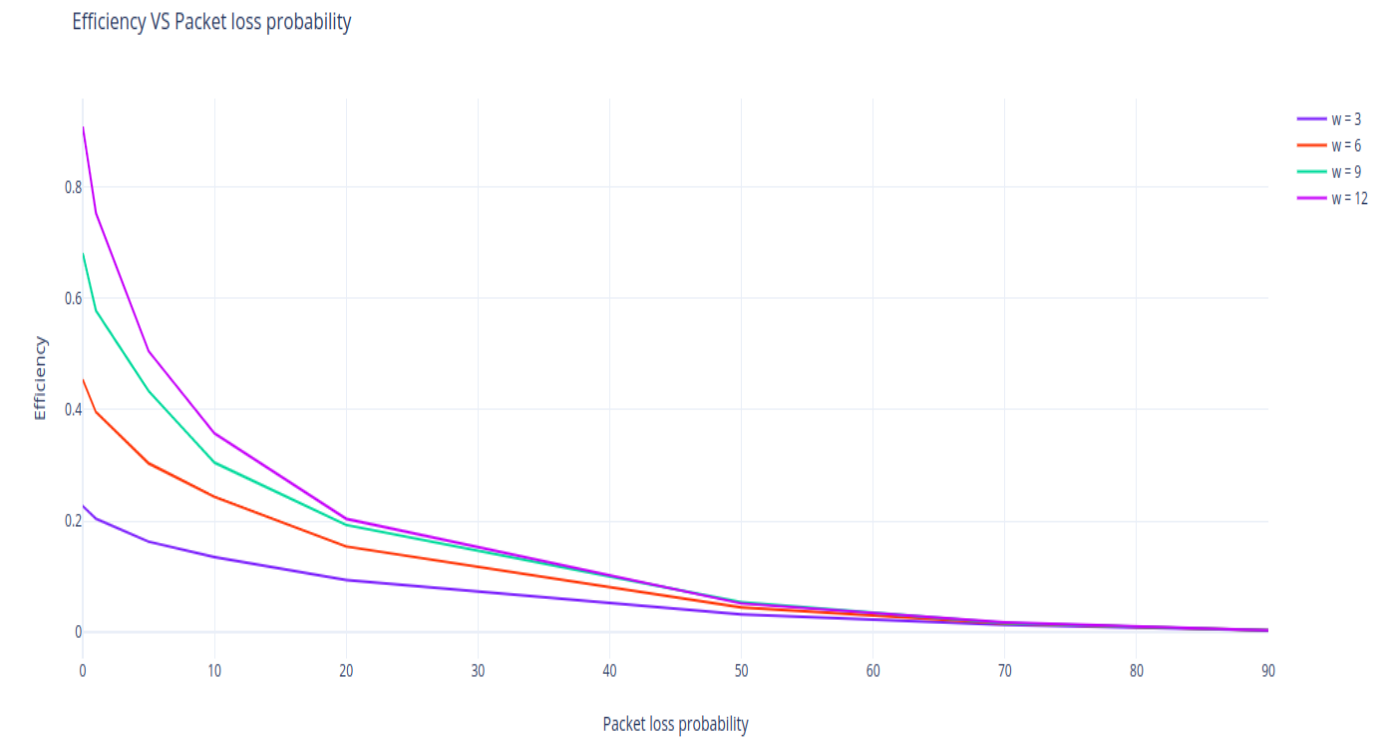
Below is table for calculated efficiency of protocol at different window size and packet loss rate:

Window Size	P_L= 0 P_A = 0	P_L= 1 P_A = 1	P_L= 5 P_A = 5	P_L= 10 P_A = 10	P_L= 20 P_A = 20	P_L= 50 P_A = 50	P_L= 70 P_A = 70	P_L= 90 P_A = 90
3	0.2271	0.2043	0.1631	0.1356	0.0942	0.0326	0.0135	0.0035

6	0.4543	0.3963	0.3038	0.2437	0.1544	0.0448	0.0162	0.0038
9	0.6814	0.5781	0.4335	0.3051	0.1931	0.0541	0.0167	0.0039
12	0.9086	0.7534	0.5055	0.3576	0.2040	0.0521	0.0181	0.0037

Table -1

Below graph show us variation in efficiency when window size and packet loss rate probability



Graph -1

Experiment 2

Packet size = 1024 Bytes
Bandwidth = 1024 mbps
Propagation Time = 0.5 ms
Total no of packet sent = 1000
Timeout = 2.16

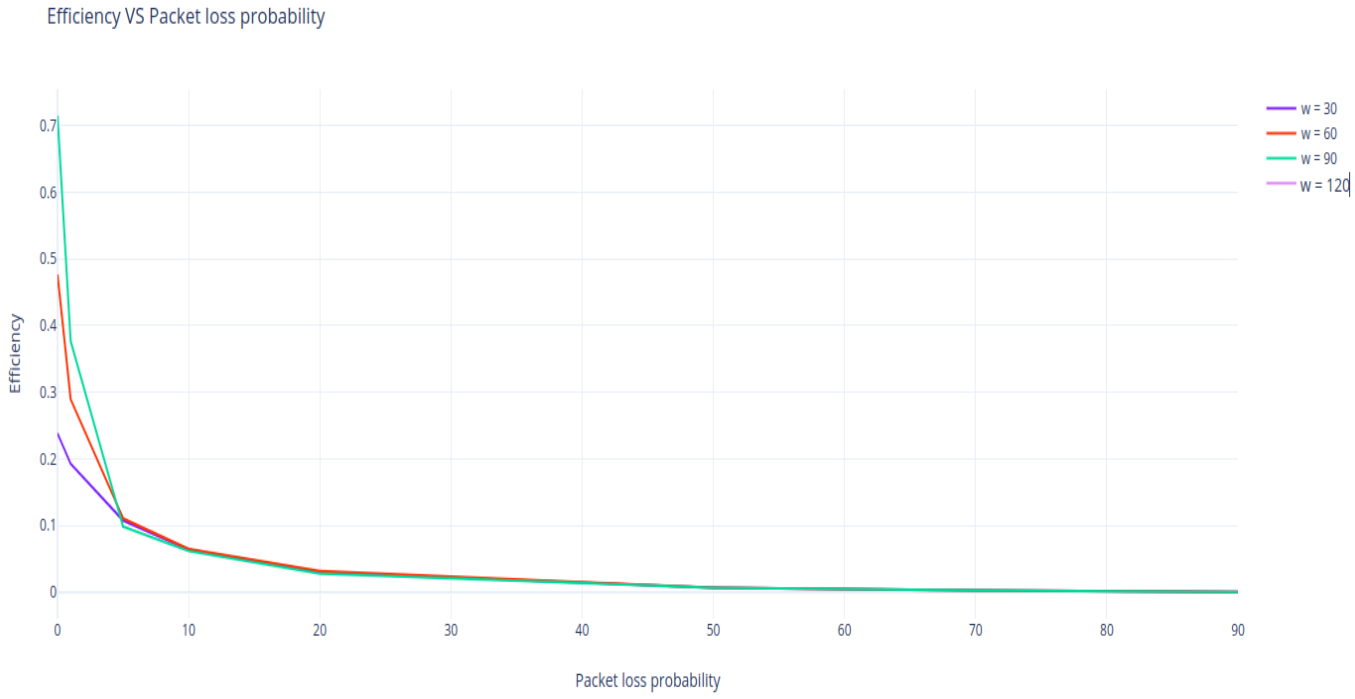
Below is table for calculated efficiency of protocol at different window size and packet loss rate:

Window Size	P_L= 0 P_A = 0	P_L= 1 P_A = 1	P_L= 5 P_A = 5	P_L= 10 P_A = 10	P_L= 20 P_A = 20	P_L= 50 P_A = 50	P_L= 70 P_A = 70	P_L= 90 P_A = 90
30	0.2383	0.1932	0.1079	0.0641	0.0301	0.0071	0.0029	0.0004
60	0.4766	0.2896	0.1110	0.0651	0.0321	0.0071	0.0031	0.0004

90	0.7149	0.3769	0.0986	0.0619	0.0280	0.0071	0.0032	0.0005
120	0.9533	0.3735	0.1270	0.0709	0.0320	0.0070	0.0033	0.0005

Table 2

Below graph show us variation in efficiency when window size and packet loss rate probability



Graph 2

- **Analysis**

The derived efficiency is based upon that the transmission window is the required size for the propagation delay and transmission speed. A slight variation in packet loss rate show the effect on efficiency of protocol. At higher probability it show a very slight variation in efficiency

Impact of Altered Parameters on Efficiency:

1. Window Size Variations:

- **Increasing Window Size**: As shown in Graph 1 and Table 1, enlarging the window size typically results in improved efficiency and throughput. The larger window accommodates more unacknowledged packets, optimizing throughput. However, substantial increases in packet loss rate might lead to increased retransmissions, impacting overall efficiency adversely.
- **Decreasing Window Size:** Contrarily, reducing the window size introduces a distinct pattern, highlighted in Graph 1 and Table 1. The most notable difference in efficiency occurs when the packet loss rate tends toward zero. Smaller windows exhibit persistent discrepancies in efficiency compared to the baseline, particularly noticeable at higher bit error rates.

2. Bandwidth Alterations:

- **High Bandwidth:** Higher bandwidth, as observed, correlates with increased transmission efficiency, evident from the trends in the Table 1 and Table 2. Greater bandwidth facilitates quicker packet transmission and acknowledgment, reducing the impact of latency or packet loss on overall efficiency.
- **Low Bandwidth**: Conversely, limited bandwidth constrains the protocol's efficiency. Decreased throughput due to longer transmission times and increased vulnerability to packet loss or acknowledgment delays are evident, resulting in decreased efficiency, as depicted in the graphs.

3. Packet Loss Probability Adjustments:

- **Low Packet Loss Probability:** Lower probabilities of packet loss typically coincide with higher protocol efficiency, aligning with the observations in the provided graphs. Reduced packet loss ensures fewer retransmissions, optimizing bandwidth utilization and throughput.
- **High Packet Loss Probability:** Elevated probabilities of packet loss significantly impact protocol efficiency. The increase in retransmissions due to higher loss probabilities results in reduced throughput and overall efficiency, as evidenced in the graphs.

- **Conclusion**

Maximizing Throughput and minimizing packet retransmission:

The analysis showcases distinctive trends concerning alterations in window size, bandwidth, and packet loss probability. While larger window sizes and higher bandwidth generally lead to improved efficiency, optimal values must be sought to avoid excessive retransmissions. Additionally, lower packet loss probabilities consistently result in enhanced efficiency, emphasizing their critical role in protocol performance. Thus, we can use below upgradation in classic go-back-n to improve it's throughput and minimize packet loss rate:

- **Optimize window size:** Choose an appropriate window size that balances throughput and reliability. A larger window size generally increases throughput, but it also increases the risk of packet loss and retransmissions. Experiment with different window sizes to find the optimal value window size.
- **Dynamic Window Size Adjustment:** Implement mechanisms for dynamic adjustment of the window size based on the observed network conditions. Adaptive algorithms that adjust the window size dynamically can help optimize throughput in varying network conditions specific network conditions.
- **Optimize Timeout Values**: Tune the timeout values carefully. Set timeout values based on the estimated Round-Trip Time (RTT) and adjust dynamically to adapt to changing network conditions. Optimizing timeout values can help reduce unnecessary retransmissions and improve throughput.
- **Efficient Retransmission Handling**: Handle retransmissions efficiently. Instead of retransmitting the entire window, selectively retransmit only the necessary packets. This can reduce the amount of redundant data sent, improving efficiency.