**Project-1**

**Research on pipelining protocols Go-Back-N and Selective Repeat**

**Course: 5800 (Advanced Computer Networking and security)**

**Professor:** Yiheng Liang

**Name:** M. Praneeth Kumar

**Student Id:** 700644073

**Introduction:**

For a reliable data transfer between sender and receiver there are multiple factors that contribute. Protocols that contribute reliability have to include several functions like Error detection, Acknowledgements, Retransmissions. In computer networking, reliable data transfer based on such retransmissions are known as **ARQ** (**Automatic repeat request**) protocols. The major terms are explained below which play important role in transferring data from sender to receiver.

**Error detection:** Detect bit errors, packet loss or duplication.

**Acknowledgements:** Feedback from the receiver to the sender. A cumulative ACK lets the sender know that several sequential packets are received correctly. In contrast a selective ACK acknowledges just one individual packet.

**Retransmissions:** lost packets are automatically retransmitted from the sender after a timeout, when no ACK is received.

In any communication protocol based on automatic repeat request for error control, the receiver must acknowledge received packets. If the transmitter does not receive an acknowledgment within a reasonable time, it re-sends the data. This process of transmission and waiting for acknowledgment is followed for each and every packet until all are done. So inorder to increase the stop and wait efficiency we will go for **Sliding window protocols** which uses the concept of pipelining.

**Protocol pipelining** is a technique in which multiple packets are sent out to a single link without waiting for the corresponding responses. This approach can be used for improvement in protocol performance, especially over high latency connections which reduces waiting time of a packet.

So, Sliding window protocols which is a theoretical concept is implemented practically in two ways following the pipelining approach to increase the stop and wait efficiency.

* Go-Back-N
* Selective Repeat

In this report, I have mentioned the design approach I followed to design the protocols and compare the performance varying different parameters of the network at sender and receiver and know how they work and choose the best possible depending on the circumstances and availability of the resources.

**Parameter Definitions:**

Before going into detail explanation of Go-Back-N and Selective Repeat we will define various parameters that are used to describe and simulate the protocols.

**Packet:** A Packet is a unit of data that is to be transmitted from sender to receiver or vice-versa. In the simulation we will be using two terms like mentioned below:

* Data packet: It is an information packet that is transmitted from sender to receiver
* Ack packet: It is a packet that is used to deliver the response message(ACK)

**Acknowledgment:** It is a response message that is used to indicate when a packet is received.

**Window size:** Window size represents the logical boundary of the total number of packets to be transmitted at the sender or the number of packets to be acknowledged at the receiver. Usually it is an integer.

**Sequence number:** These are the packet numbers inorder to identify which packet is received.

**Transmission delay:** Transmission delay is the amount of time required by the router to push out the packet. It is mathematically calculated as below:

**dtransmission = L/R**

where L is length of packet(bits)

R is transmission rate of link (bits/sec)

**Propagation delay:** Propagation delay is the time it takes a bit to propagate from router to next. It is mathematically calculated as below:

**dpropagation = d/s**

where d is length of physical link

s is propagation speed in medium (~2\*108 m/sec)

**Timeout setting:** A Timer is used at receiver or sender inorder to keep some timeout time. When the current time exceeds the timeout period, the packet is retransimitted if any acknowledgement is not received.

**Packet loss rate:** Packet loss occurs when one or more packets travelling across the network fail to reach the destination. The rate at which packet loss occurs is called packet loss rate.

**Round trip time (RTT):** It is the total time taken by the packet to reach the receiver and an acknowledgement back to sender that the packet has received successfully.

**1 RTT = 2\*dend-end**

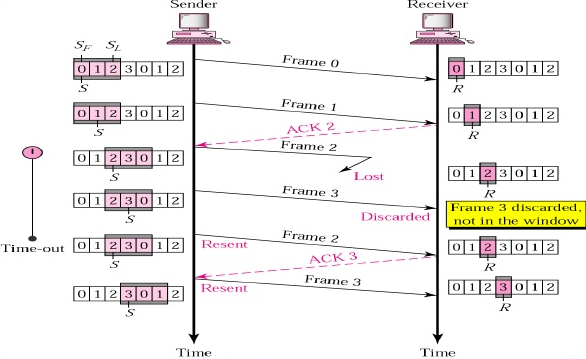
**Throughput:** It is the number of packets received per second.

**Explanation and definition of protocols:**

**Go-Back-N:**

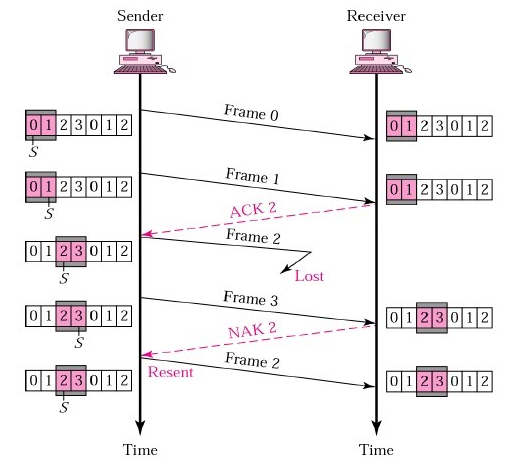
In a **Go-Back-N (GBN) protocol**, the sender continues to send a number of packets specified by a window size(N) even without receiving an acknowledgement (ACK) packet from the receiver. As the protocol operates, this window slides forward over the sequence number space. For this reason, Nis often referred to as the **window size** at the sender and window size of 1 at receiver and the GBN protocol itself as a **sliding-window protocol**.

The receiver keeps track of the sequence number of the next packet it expects to receive, and sends that number with every ACK it sends. The receiver will discard any packet that does not have the exact sequence number it expects and will resend an ACK for the last correct in-order packet. In result the sender fills the window starting with the sequence number for which it received last ACK and continues the process again. The above process is followed when a packet faces timeout at sender.



**Selective Repeat :**

**Selective repeat** protocols avoid unnecessary retransmissions by having the sender retransmit only those packets that it suspects were lost. This individual retransmissions require that the receiver individually acknowledges correctly received packets. The receiver accepts out-of-order frames and buffers them. The sender individually retransmits frames that have timed out.

****

**Keypoints:**

|  |  |  |
| --- | --- | --- |
|  | **Sender** | **Receiver** |
| **Go Back N** | * Sender window size is >1(say window size N) * Sender must buffer up to N packets while waiting for their ACK. * Sender has to resend entire window incase of an error. * A timer runs here, and retransmits a packet if it reaches timeout | * Receiver window size is = 1 * No buffering of packets at receiver. * Receiver can only accept packets in order. Out of order packets are discarded. * It wont accept packet p+1 before p is received and ACKd. So this removed need for buffering * It introduces need to resend the entire window |
| **Selective Repeat** | * Sender window size is >1(say window size N) * Sender must buffer up to N packets while waiting for their ACK. * Sender has to resend only those which does not receive proper acknowledgement | * Similar to sender window with size N. * Unlike Go Back N receiver, SR receiver has buffering capacity since receiver size >1. * Selectively acknowledges the packet |

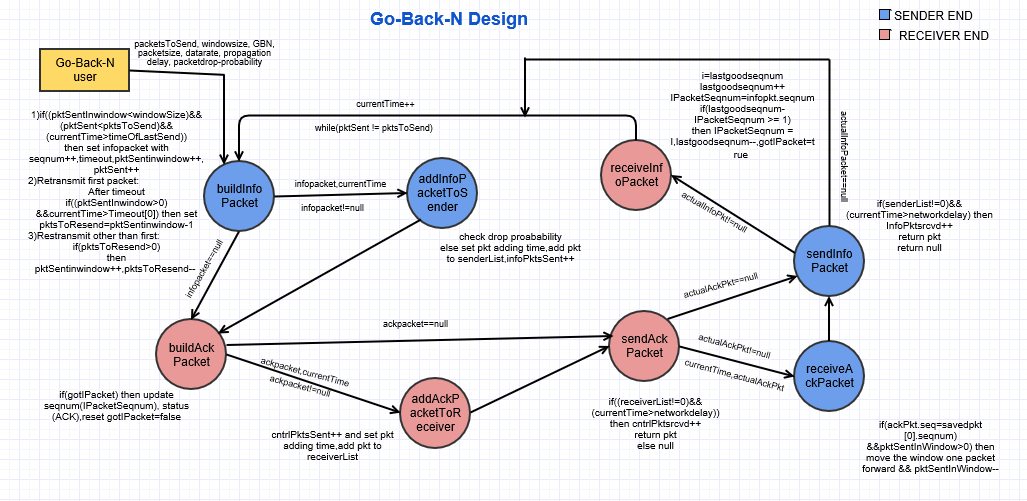
**Assumptions:**

* I assumed that network delay comprises only propagation delay and transmission delay. And other delays like processing delay and queuing delay are neglected as they are usually too small (like nano seconds).
* Assuming timeout period to be 5 msec fixed in the entire simulation to plot the charts
* Assuming packet loss as a function probability using random function. I have assumed in such a way because in the code the packet has to be dropped automatically. Random function best suits this as it is a Gaussian function that provides numbers between 0 and 1 randomly.
* Statuses of the packets as 0 initially, when transmitting 2, and when it is transmitted successfully the status is 1.

**Design of the protocols:**

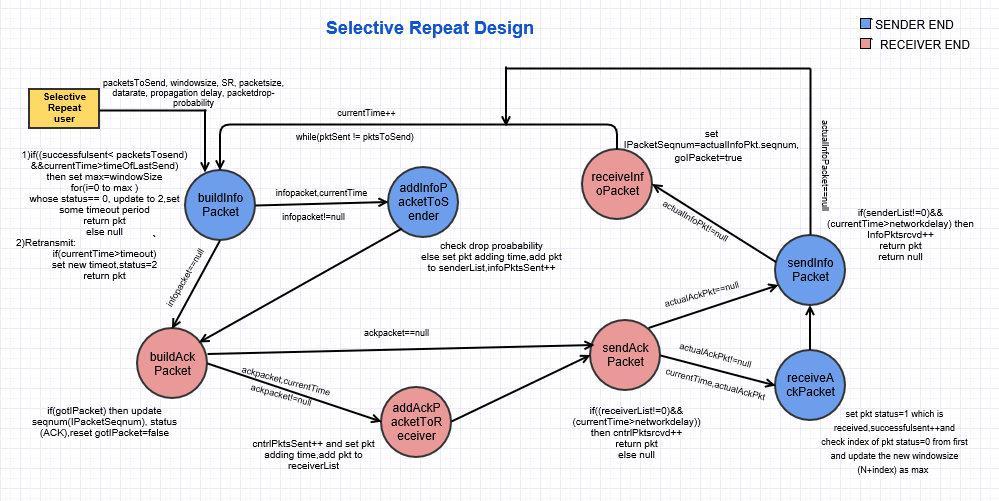
**Go Back N:**

The design approach I followed is indicated in the form of design chart. I considered a while loop which will be running continuously until the total packets successfully sent is equal to number of packets to send. Each iteration is considered like 1msec. For Go Back N when a packet is transmitted successfully, the packets sent in window is decreased by 1 and a new packet is added. When a packet fails to reach the destination, the entire window with packets are transmitted from last successfully sent packet number. Default timeout of 5msec is used for the packets at sender side. When a Information packet reaches timeout, the entire window packets start transmitting from sender.



**Selective Repeat:**

To implement selective repeat, I considered status of the packets. Similar approach is followed running in a while loop. When a packet is successfully sent, the status of the packets is updated as 1. For those that are not 1 in the window, the timeout period is checked and retransmitted after timeout. Once a packet in the window is done, the window moves forward and other packets are added to the window.

****

**Pseudocode:**

It is clearly mentioned in the design chart above. But I’ll put it in a shorter and crisp way.

**🡪 Sender:**

**if** (successfulsent<packetsTosend)

**do** send\_packets\_with\_attributes // set timeout of packet and check if packet is dropped

**else if** (currentTime > timeout\_of\_packet) based on probability

**do** retransmit\_first\_packet\_in\_window

packetsToresend = windowsize-1

**else if** (packetsToresend>1)

**do** retransmit\_other\_packets\_in\_window

packetsToresend=-1

//when correct ACK packet is received slide the window by 1 packet

**if** (correct\_seq\_number\_isobtaned)

**do** packets[0] = packets[0+1] //assigning next packet from array

* **Receiver:**

**if** (info\_packet\_is\_received\_successfully)

**do** update\_relevant\_parameters (like seq\_num, info\_pkt\_rcvd=’true’)

**if** (info\_pkt\_rcvd)

**do** send\_acknowledgment

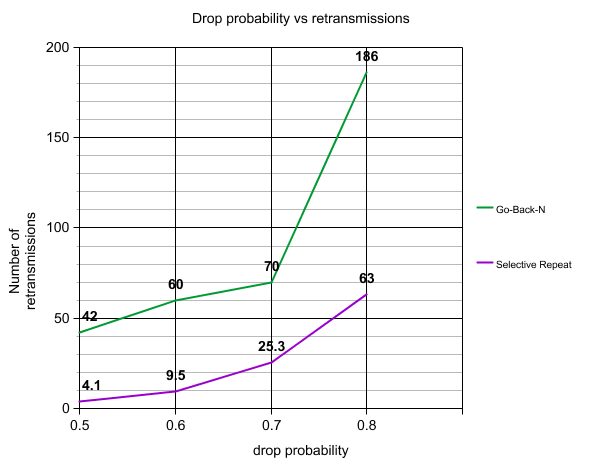
**Comparisons:**

Using the above designed simulation I have compared the performances of the two pipelining protocols varying parameters like window size, network delay, timeout, drop probability.

1. **In a noisy link:**

In a noisy link, the packet drop probability is high. So considering this I’ll give packet drop probability as 0.8. I will now check the number of retransmissions in Go Back N and selective repeat. So for Numofpackets = 10, Sender window = 3, Propagation-delay = 49 msec, Data-rate = 1000 bps, Packet size = 1 bits, Drop probability = 0.5, 0.6, 0.7, 0.8

No. of average retransmissions observed incase of Go Back N are 186 and incase of selective repeat are 63. Similarly for 0.7 probability 70, 25.3 respectively. 0.6🡪60,9.5



So, we can say that Go Back N is inefficient for noisy link, the converse is the case of selective repeat.

1. In Go Back N the sender window size is N, and receiver size is 1. Incase of selective repeat the sender window size is N and receiver size is N. so number of buffers required to store the packet data are N+1 and N+N respectively. So, Go Back N is less complicated than selective repeat interms of buffer maintenance. Considering GBN and SR with same window size and both are running in non-noisy link without any error in packet.

Parameters considered:

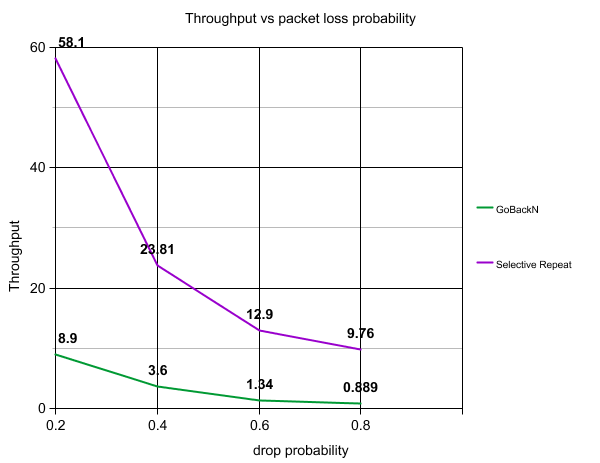
NumofPackets = 100, Sender window = 15, Propagation delay = 49 msec, Data-rate = 1000 bps, Packet size = 1bits, Drop probability = 0.05

Time taken to transmit is observed as **4878 msec** with **129** retransmissions in Go Back N and **2127 msec** and **4** retransmissions in SR. This is because the receiver window size in Go Back N is 1 and there is no much buffer to save the packet data. But in case of SR, as the receiver window is same as sender window the transmission is little fast (if we do not consider the cumulative and selective acknowledgments).

1. **Drop probability vs throughput with same window size**

The throughput is calculated increasing packet drop probability from 0.1 to 0.3 with 0.05 variations. **Note**: Throughput is the number of packets received per second).

The observations are put in a graph and can see that the Go Back N protocol has minimum throughput when compared with SR in a noisy channel. (Since packet has max drop probability)



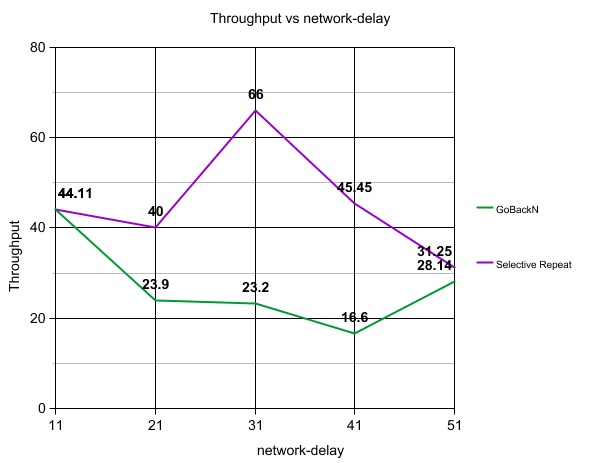
1. Window size is maximum in a noise free link.

In a noise free channel, when the window size is maximum both Go Back N and Selective repeat behave almost same.

For small window and large window compare.

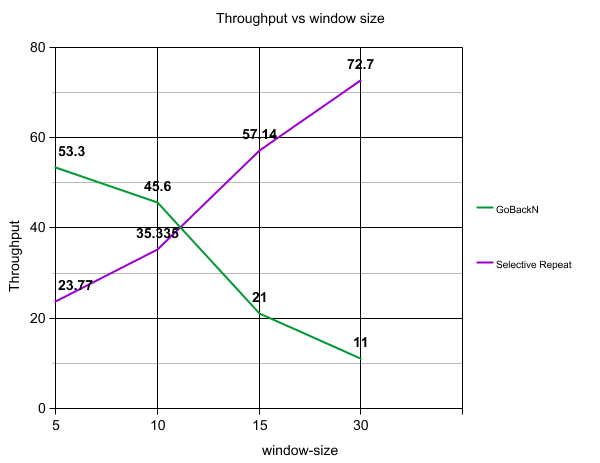
1. **Throughput vs network delay**

Considering propagation delay is varying, Window size is15, Number of packets to send are 100, Drop probability is minimum say 0.05. Bandwidth – 1000bps



1. **Throughput vs varying window size**.

Considering propagation delay is 5msec, Window size is varying, Number of packets to send are 100, Drop probability is minimum say 0.05. The throughput varies with increase in window size for both the protocols.

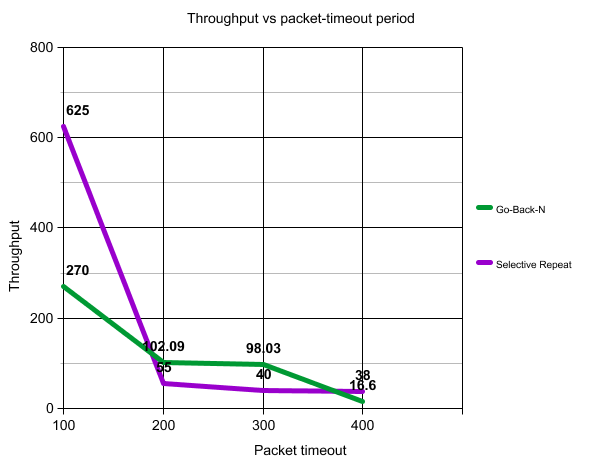


1. **Inorder to detect the duplicate packets:**

If N is sender window size, N sequence numbers are required to send the packets and 1 to detect the duplicate packets. So in total N+1 sequence numbers are needed for Go back n to detect the duplicate elements and work without any problem. So, available\_seq\_num >= N+1

1. **Timeout of packet is increasing vs throughput**

As the timeout increases the throughput decreases. If the window size is small, The bandwidth is not utilized properly as the sender sits idle for a long time, waiting for the ack or timeout of the packet to occur. As number of retransmissions occur, the throughput for GBN decreases, similar is the case of SR



**Discussions and observations:**

From the computations and simulations, I can analyze and observe that:

* Go Back N is inefficient for noisy link, the converse is the case of selective repeat. This is because when the drop probability of a packet is high, then the number of retransmissions are high.
* We can also observe that the packets are not buffered at receiver incase of Go Back N and discards them but incase of SR protocol the receiver buffers the packets and asks for retransmission only those that did not receive ACK. So number of buffers required are N+1 and N+N for Go Back N and selective repeat respectively.
* When the sender window size is set to 1, the protocols work like a stop and wait protocol, so there is inefficiency in waiting for the acknowledgment to send another packet and bandwidth is wasted.
* I have also seen that when 1 packet is lost, then entire window must be retransmitted again in Go Back N and only selective packet in SR protocol. So in Go Back N I can say retransmissions are N and incase of SR it is 1.
* From the simulations and retransmission graphs I can tell that Bandwidth consumption is high incase of Go Back N because even 1 packet is lost, entire window is being retransmitted. But in SR protocol we can tell it is moderate since less number of retransmissions.
* As the packet loss probability increases, throughput decreases for both the protocols but if we compare Go Back N and selective repeat, selective repeat has high throughput.
* When the network delay increases, throughput is high for selective repeat when compared with Go Back N. Network delay can be computed with the sum of propagation delay and transmission delay where transmission delay is packet size divided by bandwidth.
* If the window size increases, the performance of Go Back N decreases when compared to selective repeat because if one packet is dropped we need to send the entire window again, so there will be maximum number of retransmissions possible when window is maximum, so less number of successful info packets reach receiver.
* As the timeout of the packet increases, the throughput decreases for both Go Back N and SR. but when we compare the above two selective repeat has high throughput. This is because when a packet ACK is not received, it waits until the timeout has reached. So the throughput decreases.
* Thus the observations can be tabularized from a **broader perspective** apart from performances when the parameters are changed.

|  |  |  |
| --- | --- | --- |
| **Requirement** | **Go Back N** | **Selective Repeat** |
| Bandwidth utilization | High | Moderate |
| Buffer capacity | N+1 | N+N |
| Retransmissions(for 1 packet) | N | 1 |
| CPU | Moderate | High |
| Implementation | Moderate | Complex |

**Conclusion:**

Thus the above observed results are analyzed and understood that both Go Back N and Selective Repeat protocols have their uniqueness. Depending on the circumstances and available resources the best protocol must be chosen. If buffer capacity and CPU resources are high then selective repeat is chosen otherwise Go Back N protocol fits the purpose. If number of retransmissions and elapsed time of the transmission is to be less then selective Repeat is chosen. If Bandwidth provided is high then Go Back N can be used otherwise Selective Repeat. Implementation of Selective Repeat is complex process since there is huge CPU resource involvement on both sender and receiver, so in situations like less cost and code, we can go with Go Back N. Thus we can conclude that there is no strict rule of choosing single protocol, depending on the circumstances and the requirements provided we can go for combination of both the pipelining protocols Go Back N and Selective Repeat.

**References:**

* Computer Networking, A Top down approach. James F. Kurose, Keith W. Ross
* <http://literacybase.com/go-back-n-sliding-window-protocol/>
* <http://www.myreadingroom.co.in/images/stories/docs/dcn/selective%20repeat%20automatic%20repeat%20request.pdf>
* <http://web.mit.edu/modiano/www/6.263/lec3-4.pdf>
* <https://en.wikipedia.org/wiki/Sliding_window_protocol>
* <ftp://gaia.cs.umass.edu/cs653-99/rel2.pdf>
* <http://www.slideshare.net/soumya604159/ch-11-1751132>

**Tools used:** I used below tools online to prepare necessary graphs and diagrams.

* For Graphs: <https://nces.ed.gov/nceskids/createagraph/default.aspx?ID=0ca0beff8e7d4f05a8ff20ce88bc1b51>
* For Design explanation chart:

<https://www.gliffy.com/go/oauthStart?provider=facebook&app=1b5094b0-6042-11e2-bcfd-0800200c9a66>