COL334

Computer Networks

Lab Assignment II

Udit Jain - 2016CS10327
Pratyush Maini - 2016CS10412

Department of Computer Science, IIT-Delhi

September 30, 2018

Introduction

Go-Back-N is a specific instance of the automatic repeat request protocol, in which the sending process continues to send a number of frames specified by a window size even without receiving an acknowledgement (ACK) packet from the receiver. It is a special case of the general sliding window protocol with the transmit window size of N and receive window size of 1. It can transmit N frames to the peer before requiring an ACK.

The receiver process keeps track of the sequence number of the next frame it expects to receive, and sends that number with every ACK it sends. The receiver will discard any frame that does not have the exact sequence number it expects (either a duplicate frame it already acknowledged, or an out-of-order frame it expects to receive later) and will resend an ACK for the last correct in-order frame. Once the sender has sent all of the frames in its window, it will detect that all of the frames since the first lost frame are outstanding, and will go back to the sequence number of the last ACK it received from the receiver process and fill its window starting with that frame and continue the process over again.

Then we can additionally vary the parameters like Sliding window size, packet drop and error probability, latency, piggy-back acknowledgement packets and get an optimal version of Go-back-N protocol.

There are different concerns in the process like correcting / requesting additional packets in case of check-sum errors, choosing optimal window size etc.

Choosing a Window size (N) :
There are a few things to keep in mind when choosing a value for N:

- The sender must not transmit too fast. N should be bounded by the receiver's ability to process packets.
- N must be smaller than the number of sequence numbers (if they are numbered from zero to N) to verify transmission in cases of any packet (any data or ACK packet) being dropped.
- Given the bounds presented in (1) and (2), choose N to be the largest number possible.

Implementation

Pseudo Code :

```
N  = window size
Rn = request number
Sn = sequence number
Sb = sequence base
Sm = sequence max

Receiver:
Rn = 0
Do the following forever:
If the packet received = Rn and the packet is error free
        Accept the packet and send it to a higher layer
        Rn = Rn + 1
Else
        Refuse packet
Send a Request for Rn

Sender:
Sb = 0
Sm = N + 1
Repeat the following steps forever:
1. If you receive a request number where Rn > Sb
        Sm = (Sm - Sb) + Rn
        Sb = Rn
2.  If no packet is in transmission,
        Transmit a packet where Sb <= Sn <= Sm.
        Packets are transmitted in order.
```

1. **Transmission Delay (Tt)** – Time to transmit the packet from host to the outgoing link. If B is the Bandwidth of the link and D is the Data Size to transmit

   ```
   Tt = D/B
   ```

2. **Propagation Delay (Tp)** – It is the time taken by the first bit transferred by the host onto the outgoing link to reach the destination. It depends on the distance d and the wave propagation speed s (depends on the characteristics of the medium).

   ```
    Tp = d/s
   ```

3. **Efficiency** – It is defined as the ratio of total useful time to the total cycle time of a packet. For stop and wait protocol,

   ```
   Total cycle time = Tt(data) + Tp(data) +
                       Tt(acknowledgement) + Tp(acknowledgement)
                    =  Tt(data) + Tp(data) + Tp(acknowledgement)
                    =   Tt + 2*Tp
   ```

   Since acknowledgements are very less in size, their transmission delay can be neglected.

```
Efficiency = Useful Time / Total Cycle Time
           = Tt/(Tt + 2*Tp) (For Stop and Wait)
           = 1/(1+2a)  [ Using a = Tp/Tt ]
```

4. **Effective Bandwidth(EB) or Throughput** – Number of bits sent per second.

```
EB = Data Size(L) / Total Cycle time(Tt + 2*Tp)
Multiplying and dividing by Bandwidth (B),
       =  (1/(1+2a)) * B   [ Using a = Tp/Tt ]
       =  Efficiency * Bandwidth
```

5. **Capacity of link** – If a channel is Full Duplex, then bits can be transferred in both the directions and without any collisions. Number of bits a channel/Link can hold at maximum is its capacity.

```
 Capacity = Bandwidth(B) * Propagation(Tp)
 and
 Capacity = 2*Bandwidth(B) * Propagation(Tp)
```

for Full Duplex channels

Pipe-lining: In Stop and Wait protocol, only 1 packet is transmitted onto the link and then sender waits for acknowledgement from the receiver. The problem in this setup is that efficiency is very less as we are not filling the channel with more packets after 1st packet has been put onto the link. Within the total cycle time of Tt + 2*Tp units, we will now calculate the maximum number of packets that sender can transmit on the link before getting an acknowledgement.

```

 In Tt units ----> 1 packet is Transmitted.
 In 1 units  ----> 1/Tt packet can be Transmitted.
 In  Tt + 2*Tp units ----->  (Tt + 2*Tp)/Tt
                             packets can be Transmitted
                   ------>  1 + 2a  [Using a = Tp/Tt]
Maximum packets That can be Transmitted in total cycle time = 1+2*a
```

Sender Window Size (WS) It is N itself. If we say protocol is GB10, then Ws = 10. N should be always greater than 1 in order to implement pipelining. For N = 1, it reduces to Stop and Wait protocol.

```
Efficiency Of GBN = N/(1+2a) Where a = Tp/Tt

If B is the bandwidth of the channel, then Effective Bandwidth or
   Throughput = Efficiency * Bandwidth = (N/(1+2a)) * B.

Receiver Window Size (WR)
WR is always 1 in GBN.
```

Acknowledgements There are 2 kinds of acknowledgements namely :

- Cumulative Ack – One acknowledgement is used for many packets. Main advantage is traffic is less. Disadvantage is less reliability as if one ack is loss that would mean that all the packets sent are lost.

- Independent Ack – If every packet is going to get acknowledgement independently. Reliability is high here but disadvantage is that traffic is also high since for every packet we are receiving independent ack.

Go-Back-N uses Cumulative Acknowledgement. At the receiver side, it starts a acknowledgement timer whenever receiver receives any packet which is fixed and when it expires, it is going to send a cumulative Ack for the number of packets received in that interval of timer. If receiver has received N packets, then the Acknowledgement number will be N+1. Important point is Acknowledgement timer will not start after the expiry of first timer but after receiver has received a packet. Time out timer at the sender side should be greater than Acknowledgement timer.

Relationship Between Window Sizes and Sequence Numbers We already know that sequence numbers required should always be equal to the size of window in any sliding window protocol.

```
Minimum sequence numbers required in GBN is N+1.
Bits Required will be ceil(log2(N+1)).
```

The extra 1 is required in order to avoid the problem of duplicate packets.

Observations and Conclusion

We tested with different conditions namely different delay, bandwidths, error-rates, latency's etc. We started from a 'sender' and a 'receiver' which communicated through sockets emulating the physical layer and the network layer loaded file's (text, image etc) bytes and splitted it into packets.
These packets (DATA-frames and ACK-frames) were sent to the transport layer and added a proper **Header(sequence number, meta-data)** and **Footer(Acknowledgement in case of piggy-back)** by the protocol and forwarded to the below physical layer.

Then we introduced error probability in the sent packets and go-back-n handled them to a certain degree.

After integrating the sender and receiver into 1 Host file it implementing the **full duplex**, we had to multi-thread the sender and receiver functions. It had to be done because we had to listen to the incoming packets and at the same time cater to the requests by the network layer if any arise(whenever it has a file to send).

- With lower-bandwidths, limiting the channel capacity, the efficiency and through-put decreased.
- With some error probability (p) it decrease and the decrements compounded with the increment of p.
- There was similar result on increasing the propagation delay.
- Changing the packet rate, didn't largely affect the evaluation metrics.