16.36 Communication Systems Engineering

Spring 2009

16.36: Communication Systems Engineering

ARQ Protocols: Go Back N and SRP

Eytan Modiano

Go Back N ARQ
(Sliding Window)

- Stop and Wait is inefficient when propagation delay is larger than the packet transmission time
  - Can only send one packet per round-trip time
- Go Back N allows the transmission of new packets before earlier ones are acknowledged

- Go back N uses a window mechanism where the sender can send packets that are within a "window" (range) of packets
  - The window advances as acknowledgements for earlier packets are received

Features of Go Back N

- Window size = N
  - Sender cannot send packet i+N until it has received the ACK for packet i
- Receiver operates just like in Stop and Wait
  - Receive packets in order
  - Receiver cannot accept packet out of sequence
  - Send RN = i + 1 => ACK for all packets up to and including i

- Use of piggybacking
  - When traffic is bi-directional RN's are piggybacked on packets going in the other direction
    - Each packet contains a SN field indicating that packet's sequence number and a RN field acknowledging packets in the other direction

<--Frame Header --------->

	SN	RN		Packet	CRC

Go Back N ARQ

- The transmitter has a "window" of N packets that can be sent without acknowledgements

- This window ranges from the last value of RN obtained from the receiver (denoted SN_{min}) to $SN_{min}+N-1$

- When the transmitter reaches the end of its window, or times out, it goes back and retransmits packet SN_{min}

Let SN_{min} be the smallest number packet not yet ACKed

Let SN_{max} be the number of the next packet to be accepted from the higher layer (I.e., the next new packet to be transmitted)

Go Back N
Sender Rules

- $SN_{min} = 0$; $SN_{max} = 0$
- Repeat
    - If $SN_{max} < SN_{min} + N$ (entire window not yet sent)
        - Send packet SN_{max} ;
        - $SN_{max} = SN_{max} + 1$;
    - If packet arrives from receiver with $RN > SN_{min}$
        - $SN_{min} = RN$;

    - If $SN_{min} < SN_{max}$ (there are still some unacknowledged packets) and sender cannot send any new packets
        - Choose some packet between SN_{min} and SN_{max} and re-send it
- The last rule says that when you cannot send any new packets you should re-send an old (not yet ACKed) packet
    - There may be two reasons for not being able to send a new packet
        - Nothing new from higher layer
        - Window expired ($SN_{max} = SN_{min} + N$)
    - No set rule on which packet to re-send
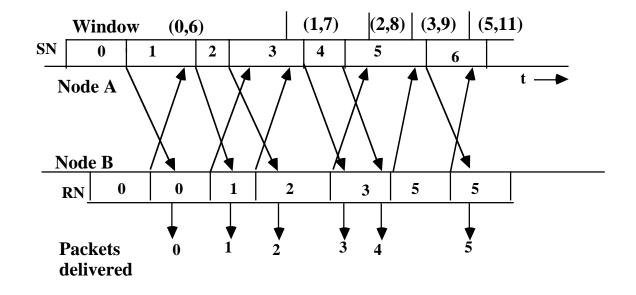   - Least recently sent

Receiver Rules

- RN = 0;
- Repeat
  - When a good packet arrives, if SN = RN
    Accept packet
    Increment RN = RN +1

- At regular intervals send an ACK packet with RN
  - Most DLCs send an ACK whenever they receive a packet from the other direction
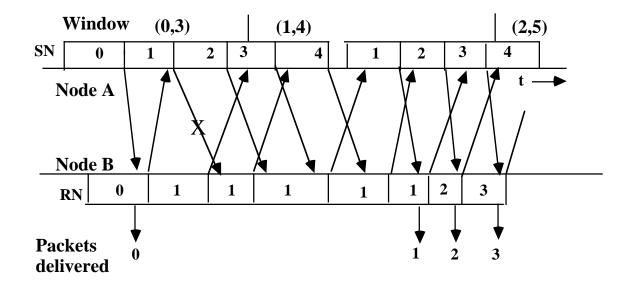   Delayed ACK for piggybacking
- Receiver reject all packets with SN not equal RN
  - However, those packets may still contain useful RN numbers
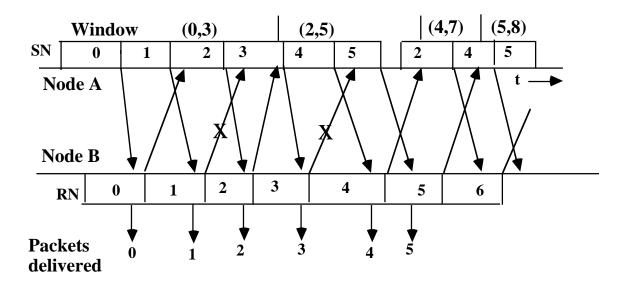
Example of Go Back 7 ARQ

- Note that packet RN-1 must be accepted at B before a frame containing request RN can start transmission at B

RETRANSMISSION BECAUSE OF ERRORS FOR GO BACK 4 ARQ

- Note that the timeout value here is taken to be the time to send a full window of packets

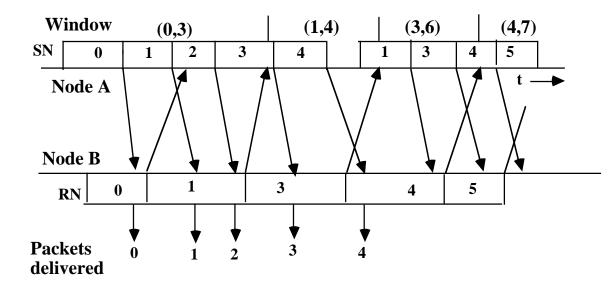- Note that entire window has to be retransmitted after an error

RETRANSMISSION DUE TO FEEDBACK ERRORS FOR GO BACK 4 ARQ

- When an error occurs in the reverse direction the ACK may still arrive in time. This is the case here where the packet from B to A with RN=2 arrives in time to prevent retransmission of packet 0

- Packet 2 is retransmitted because RN = 4 did not arrive in time, however it did arrive in time to prevent retransmission of packet 3

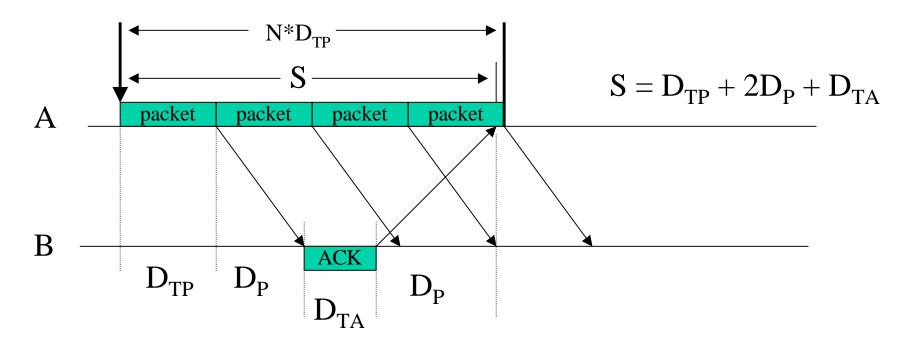 – Was retransmission of packet 4 and 5 really necessary?

    Strictly no because the window allows transmission of packets 6 and 7 before further retransmissions. However, this is implementation dependent

EFFECT OF LONG FRAMES

- Long frames in feedback direction slow down the ACKs
  - This causes a transmitter with short frames to wait or go back
- Notice again that the retransmission of packets 3 and 4 was not strictly required because the sender could have sent new packets within the window
  - Again, this is implementation dependent

Efficiency of Go Back N

$$S = D_{TP} + 2D_P + D_{TA}$$

- We want to choose N large enough to allow continuous transmission while waiting for an ACK for the first packet of the window,

$$N > S/ D_{TP}$$

- Without errors the efficiency of Go Back N is,

$$E = \min\{1, N*D_{TP}/S\}$$

Efficiency of Go Back N with transmission errors
Approximate analysis

Assume: $N = \left| \dfrac{S}{D_{TP}} \right|$ $\qquad$ $TO = N*D_{TP}$

- When an error occurs the entire window of N packets must be retransmitted

Let X = the number of packets sent per successful transmission

$$E[X] = 1*(1-P) + (X+N)*P$$

$$= 1 + N*P/(1-P)$$

$$Efficiency = 1/E[X]$$

Go Back N Requirements

- Go Back N is guaranteed to work correctly, independent of the detailed choice of which packets to repeat, if

  1) System is correctly initialized
  2) No failures in detecting errors
  3) Packets travel in FCFS order
  4) Positive probability of correct reception
  5) Transmitter occasionally resends Sn_{min} (e.g., upon timeout)
  6) Receiver occasionally sends RN

Notes on Go Back N

- Requires no buffering of packets at the receiver
- Sender must buffer up to N packets while waiting for their ACK
- Sender must re-send entire window in the event of an error
- Packets can be numbered modulo M where M > N
    - Because at most N packets can be sent simultaneously
- Receiver can only accept packets in order
    - Receiver must deliver packets in order to higher layer
    - Cannot accept packet i+1 before packet i
    - This removes the need for buffering
    - This introduces the need to re-send the entire window upon error

- The major problem with Go Back N is this need to re-send the entire window when an error occurs. This is due to the fact that the receiver can only accept packets in order

Selective Repeat Protocol (SRP)

- Selective Repeat attempts to retransmit only those packets that are actually lost (due to errors)
  - Receiver must be able to accept packets out of order
  - Since receiver must release packets to higher layer in order, the receiver must be able to buffer some packets

- Retransmission requests
  - Implicit

    The receiver acknowledges every good packet, packets that are not ACKed before a time-out are assumed lost or in error

    Notice that this approach must be used to be sure that every packet is eventually received
  - Explicit

    An explicit NAK (selective reject) can request retransmission of just one packet

    This approach can expedite the retransmission but is not strictly needed
  - One or both approaches are used in practice

SRP Rules

- Window protocol just like GO Back N
    - Window size W
- Packets are numbered Mod M where M >= 2W
- Sender can transmit new packets as long as their number is with W of all un-ACKed packets
- Sender retransmit un-ACKed packets after a timeout
    - Or upon a NAK if NAK is employed
- Receiver ACKs all correct packets
- Receiver stores correct packets until they can be delivered in order to the higher layer

Need for buffering

- Sender must buffer all packets until they are ACKed
    - Up to W un-ACKed packet are possible
- Receiver must buffer packets until they can be delivered in order
    - I.e., until all lower numbered packets have been received
    - Needed for orderly delivery of packets to the higher layer
    - Up to W packets may have to be buffered (in the event that the first packet of a window is lost)
- Implication of buffer size = W
    - Number of un-ACKed packets at sender =< W
        - Buffer limit at sender
    - Number of un-ACKed packets at sender cannot differ by more than W
        - Buffer limit at the receiver (need to deliver packets in order)
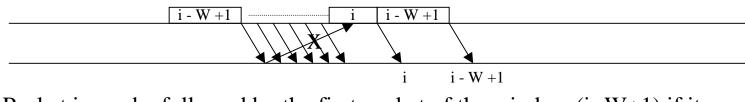   - Packets must be numbered modulo $M >= 2W$ (using $\log_2(M)$ bits)

EFFICIENCY

- For ideal SRP, only packets containing errors will be retransmitted
  - Ideal is not realistic because sometimes packets may have to be retransmitted because their window expired. However, if the window size is set to be much larger than the timeout value then this is unlikely
- With ideal SRP, efficiency = 1 - P
  - P = probability of a packet error
- Notice the difference with Go Back N where

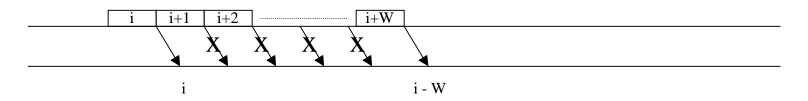$$\text{efficiency (Go Back N)} = 1/(1 + N*P/(1-P))$$

- When the window size is small performance is about the same, however with a large window SRP is much better
  - As transmission rates increase we need larger windows and hence the increased use of SRP

Why are packets numbered Modulo 2W?

- Lets consider the range of packets that may follow packet i at the receiver

Packet i may be followed by the first packet of the window (i -W+1) if it requires retransmission
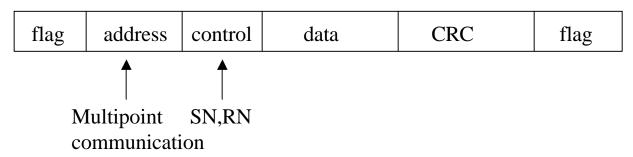

Packet i may be followed by the last packet of the window (i+W) if all Of the ACKs between i and i +W are lost

- Receiver must differentiate between packets i -W+1 ... i +W
    - These 2W packets can be differentiated using Mod 2W numbering

STANDARD DLC's

- HDLC, LAPB (X.25), and SDLC are almost the same
    - HDLC/ SDLC developed by IBM for IBM SNA networks
    - LAPB developed for X.25 networks

- They all use bit oriented framing with flag = 01111110
- They all use a 16-bit CRC for error detection
- They all use Go Back N ARQ with N = 7 or 127 (optional)

SDLC packet

flag	address	control	data	CRC	flag

Multipoint communication ↑ (address)   SN,RN ↑ (control)

- Older protocols (used for modems, e.g., xmodem) used stop and wait and simple checksums