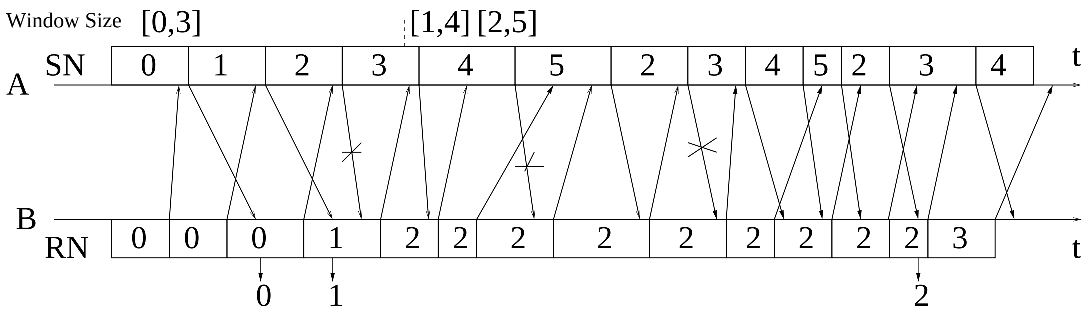
Faculty of Arts and Science
University of Toronto
CSC 358 - Computer Networks, Spring 2013

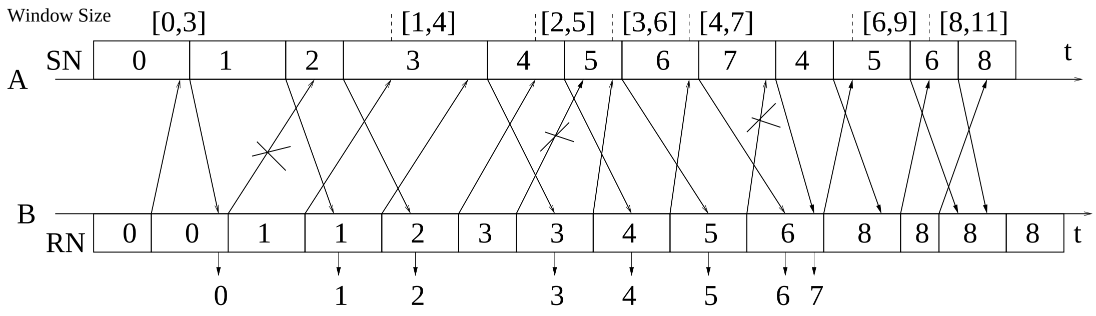
Solutions for Assignment 2

Question 1 (20 Points): Go-Back n ARQ
In this question, we review how Go-Back n ARQ can be used to ensure a reliable data transfer. Use $n = 4$ for Question (a)-(c). Use the convention that when A has to retransmit packets it starts with the SN at the beginning of the window and retransmits packets in order of their sequence number.

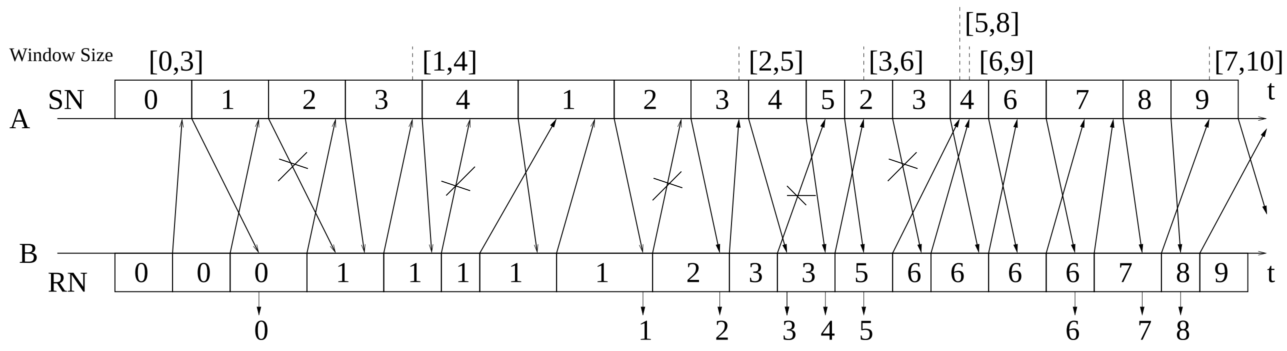
(a) (Error in Transmission from A to B) Fill in the values for SN and RN, indicate the window size, as well as the packets delivered to the next higher layer.



(b) (Error in Transmission from B to A): Repeat (a).



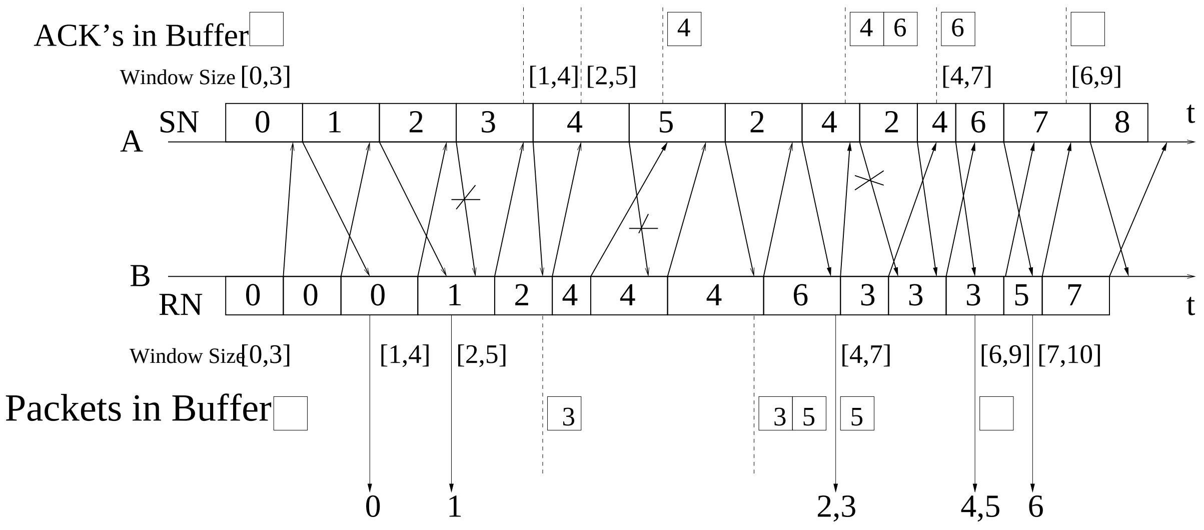
(c) Repeat (a).



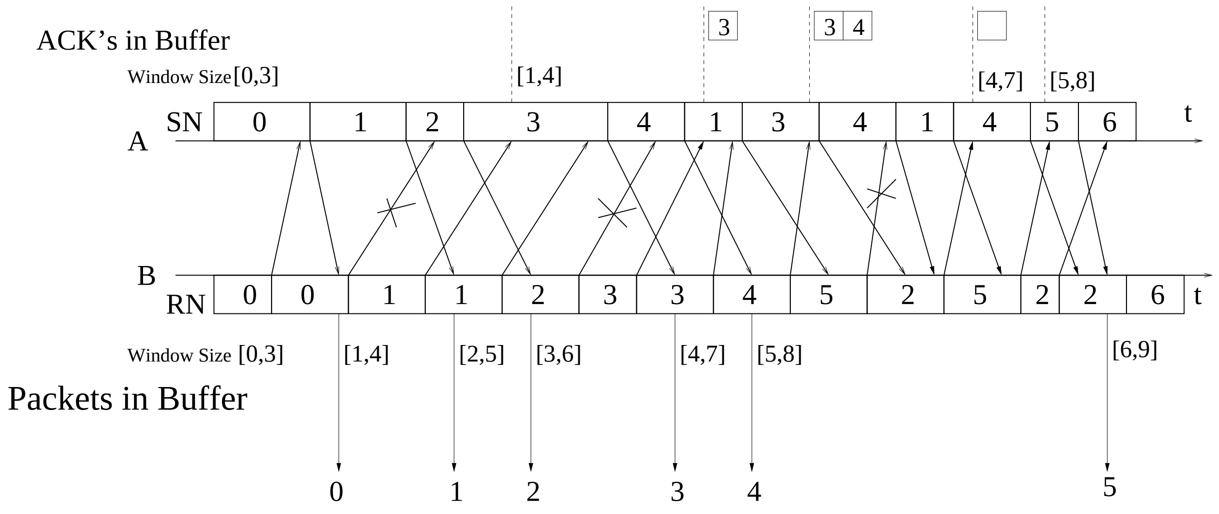
Window Size
[0,3]
[1,4]
[2,5]
[3,6]
[5,8]
[6,9]
[7,10]
A
SN
0 1 2 3 4 1 2 3 4 5 2 3 4 6 7 8 9
t
B
RN
0 0 0 1 1 1 1 1 2 3 3 5 6 6 6 6 7 8 9
t
0 1 2 3 4 5 6 7 8

Question 2 (20 Points): Selective Repeat ARQ
In this question, we review how Selective Repeat ARQ can be used to ensure a reliable data transfer. Use $n = 4$ for Question (a)-(c). Use the convention that B always acknowledges the last error-free packet from A, and when A has to retransmit packets it starts with the SN at the beginning of the window and retransmits packets that have not yet been acknowledge by B in order of their sequence number.

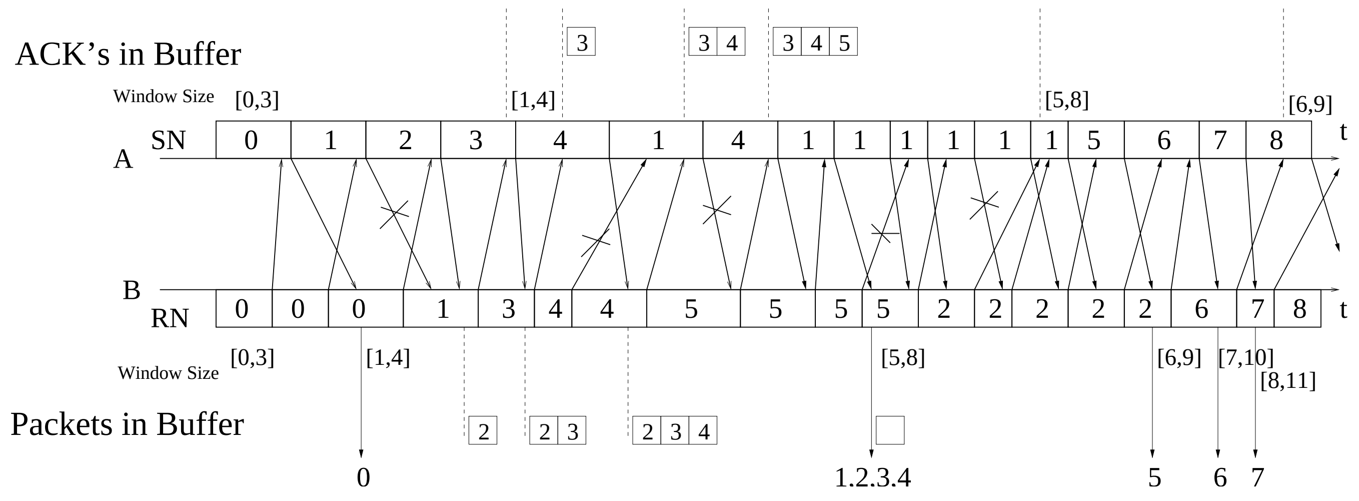
(a) (Error in Transmission from A to B) Fill in the values for SN and RN, indicate the window size at A and B, as well as the packets delivered to the next higher layer.



(b) (Error in Transmission from B to A): Repeat (a).



(c) Repeat (a).



ACK's in Buffer
3
3 4
3 4 5
Window Size
[0,3]
[1,4]
[5,8]
[6,9]
A
SN
0 1 2 3 4 1 4 1 1 1 1 1 1 5 6 7 8
t
B
RN
0 0 0 1 3 4 4 5 5 5 5 2 2 2 2 2 6 7 8
t
Window Size
[0,3]
[1,4]
[5,8]
[6,9]
[7,10]
[8,11]
Packets in Buffer
2
2 3
2 3 4
0
1.2.3.4
5
6
7

Question 3 (15 points)

Consider the situation where two peer processes, A and B, implement Stop-and-Wait ARQ to send data from A to B. The communication channel (link) between A and B (and B and A) is such that all packets that arrive are in the same order as transmitted. Below, we make different assumptions whether packets can arrive with errors and whether packets can be lost. For the different assumptions, indicate **whether it is necessary** to use sequences numbers SN for the packets sent from A to B, as well **whether it is necessary** to use request numbers RN for ACK's sent from B to A. To answer each sub-question, first indicate your answer by checking either "YES" or "NO". If you checked "YES" for SN and/or RN, then give a counter-example that shows that the protocol fails if we do not use SN and/or RN.

(a) Assume that packets (from A to B) and ACK's (from B to A) have a fixed and known delay. Furthermore assume that packets and ACK's are **never lost** and always arrive **error free**.

	YES	NO
SN		X
RN		X

(b) Assume that packets (from A to B) and ACK's (from B to A) can have an arbitrary and variable delay. Furthermore assume that packets and ACK's are **never lost** and that **ACK's always arrive error free**. However, **packets might arrive with errors**.

If the receiver sends ACK or NAK, then we do not need to number the ACK, NAK or the packets, i.e.,

	YES	NO
SN		X
RN		X

if NAK is not used, then we need to number the ACK's, i.e.,

	YES	NO
SN		X
RN	X	

Otherwise, the sender would not know from the ACK whether the receiver received an error-free packet or not, thus would not know whether to send a new packet of re-send the old packet.

(c) Assume that packets (from A to B) and ACK's (from B to A) can have an arbitrary and variable delay, but the delay is always finite. Furthermore assume that **ACK's**

are never lost and **ACK's always arrive error free**. However, **packets might be lost or arrive with errors**.

	YES	NO
SN	X	
RN		X

Because the delay can be arbitrary, when the receiver is waiting for a packet from the sender, it can not not for sure whether the packet has been lost or whether it has been delayed and will arrive later. As a result, it can not be avoided that the sender might have to send the same packet twice either the time-out at the sender expires, or because the receiver requests the same packet twice. In order to be able to able to detect re-sent packets (to avoid to deliver them more than once to the next higher-layer), SN's are needed. However, RN's are not needed. Here is an example for such a protocol.

Suppose that the sender sends packet with sequence number SN. If the sender receives a packet (ACK) from the receiver within a given time-out period, then it increases SN by 1 and sends the next packet. If the sender does not receive a packet (ACK) from the receiver within this time-out period, then it retransmits packet SN. Whenever the receiver receives an error-free packets with sequence number SN for the first time, then it delivers the packet to the next higher layer and sends an ACK. If it receives a packet that has an error, or if it receives a packet with a sequence number SN that has already been delivered to the next higher layer, then it doesn't do anything.

Note that this protocol works because we know that ACK's always arrive error-free and are never lost.

Question 4 (15 Points): Modulus m

For Go-Back n ARQ and Selective Repeat ARQ we have to be careful how we apply a modulus m to the sequence number. In this question, we illustrate this issue.

(a) Give an example where Go-Back n with modulus m fails if $m = n$. Use $n = 5$ for your example.

The simplest example is for node A to send packet 0 through $n - 1$. In case of delayed acknowledgments (i.e. no return packets in the interim), node A goes back and retransmits packet 0 (see Figure 6). If the other node has received all the packets, it is waiting for packet n, and if the modulus m equals to n, this repeat of packet 0 is interpreted as packet n.

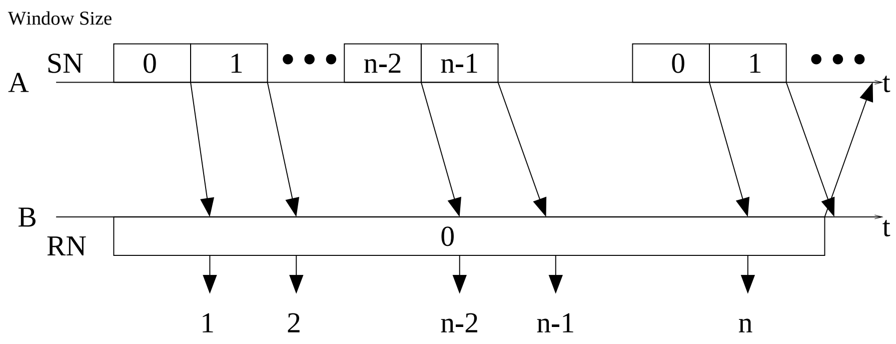


Figure 1: Examples where go back n ARQ fails when the modulus m is equal to n

(b) Give an example where Selective Repeat with modulus m fails if $m = 2n - 1$. Use $n = 5$ for your example.

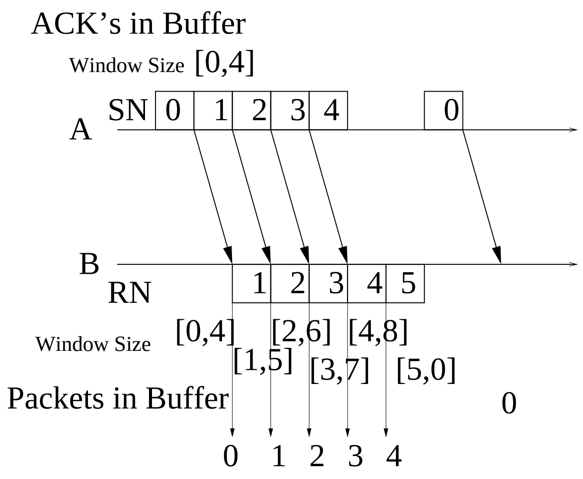


Figure 2: Note that in the above example all ACK's are lost. As a result, A retransmits packet 0 (after a time-out), but the receiver B interprets 0 as an out-of-order packets and stores it in the buffer.

Question 5 (15 Points): Stop-and-Wait ARQ
In class, we discussed that we can employ the ARQ protocol at any layer - for example, we could implement ARQ at the transport layer (end-to-end ARQ) or at the link layer (hop-by-hop ARQ). In this question, we study how the decision where we use ARQ impacts the network performance (in terms of average packet delay).
Consider a Stop-and-Wait protocol between two peer processes X (sender) and Y (receiver) where Y sends immediately an ACK (with the corresponding RN number) to X whenever it receives an error-free a packet from X.
Assume a host A wants to send packets to a (distant) host B. In order to communicate with host B, host A has to send its packets first to a switch C, where packets get forwarded to host B. All packets have the same size of N bits. Switch C must have received the complete packet before it can be forwarded to B (why?). During the transmission of a packet from host A to switch C, bits are corrupted independently with a bit error probability P_{bit}, $0 < P_{bit} < 1$. Similar, during the transmission of a packet from switch C to host B, bits are corrupted independently with the same bit error probability P_{bit}. The transmission rate of the link between host A and switch C is R bits per second - the transmission rate of the link between switch C and host B is also R bits per second.

(a) What is the probability P_{packet} that a packet from host A arrives error-free at switch C?

$$P_{packet} = (1 - P_{bit})^N$$

(b) Assume that host A and switch C implement the above Stop-and-Wait ARQ. In addition, make the following assumption regarding the channel between A and C: ACK's from switch C always arrive error-free at A and no packets or ACK's are dropped. Assuming that host A never exceeds the time-out (when waiting for an ACK), what is the probability P_k, $k = 1, 2, ...$, that A has to transmit a packet k times to get it accepted at switch C?

$$P_k = (1 - P_{packet})^{k-1} P_{packet} = \left[1 - (1 - P_{bit})^N\right]^{k-1} (1 - P_{bit})^N.$$

(c) (Hop-by-Hop ARQ) Assume that switch C and host B also implement the above Stop-and-Wait ARQ to send packets from C to B, and that the same assumption that we made in (b) for the channel between A and C also hold for the channel between C and B. What is the average delay of a packet, i.e. the average length of the interval between the time that host A starts sending a packet for the first time and the time that host B passes the packet to the next higher layer? Ignore queueing and processing delays, as well as transmission delays of ACK's (but not of packets), and assume that host A and switch C never exceed the time-out (waiting for a ACK).

Let $E[T_{AC}]$ be the total delay for one hop from A to C, and let A_k be the event that

A has to send the packet k times to get it accepted. Then we have

$$\begin{aligned}
E[T_{AC}] &= \sum_{k=1}^{\infty} E[T \mid k] P_k \\
&= \sum_{k=1}^{\infty} k\frac{N}{R} P_k \\
&= \sum_{k=1}^{\infty} k\frac{N}{R} (1 - P_{packet})^{k-1} P_{packet} \\
&= \frac{L}{R}\frac{1}{P_{packet}} = \frac{N}{R}\frac{1}{(1-P_{bit})^N}
\end{aligned}$$

Let $E[T_{CB}]$ be the total delay for the hop from C to B. Note that

$$E[T_{CB}] = E[T_{AB}]$$

And the total delay $E[T]$ is equal to

$$E[T] = E[T_{AB}] + E[T_{CB}] = 2\frac{N}{R}\frac{1}{P_{packet}} = 2\frac{N}{R}\frac{1}{(1-P_{bit})^N}.$$

(d) (End-to End ARQ) Suppose that switch C does not check packets for errors, but forwards each packet from A immediately to host B (without sending an ACK to host A). However, host A and host B use the above Stop-and-Wait ARQ to ensure a reliable data transfer. Assume that ACK's from host B always arrive error-free at host A and that no ACK's are dropped. For this case, what is the average delay of a packet? Again, ignore queueing and processing delays, as well as transmission delays of ACK's, and assume that host A never exceeds the times-out (waiting for a ACK).

The probability P'_{packet} that a packet from A arrives error free at B is given by

$$P'_{packet} = (1-P_{bit})^N (1-P_{bit})^N = (1-P_{bit})^{2N}.$$

And the total delay $E[T']$ is given by

$$E[T'] = \frac{2N}{R}\frac{1}{P'_{packet}} = \frac{2N}{R}\frac{1}{(1-P_{bit})^{2N}}.$$

(e) Evaluate the answers for (c) and (d) for $P_{packet} = 0.99, 0.1, 0.001$. When would you use Hop-by-Hop ARQ (End-to-End ARQ)?

For $P_{packet} = 0.99$

$$E[T] = 2.02\frac{N}{R} \qquad \text{and} E[T'] = 2.04\frac{N}{R},$$

For $P_{packet} = 0.1$

$$E[T] = 20\frac{N}{R} \qquad \text{and} \qquad E[T'] = 200\frac{N}{R},$$

For $P_{packet} = 0.001$

$$E[T] = 2000\frac{N}{R} \qquad \text{and} \qquad E[T'] = 2 \cdot 10^6 \frac{N}{R},$$

If the bit error probability P_{bit} is quite high (such as for a wireless channel), then using Hop-by-Hop ARQ significantly reduces the exepcted delay. If the bit error probability P_{bit} is very low (such as for an optical channel), then using End-to-End ARQ does not significantly increase exepcted delay and is less expensive to implement.