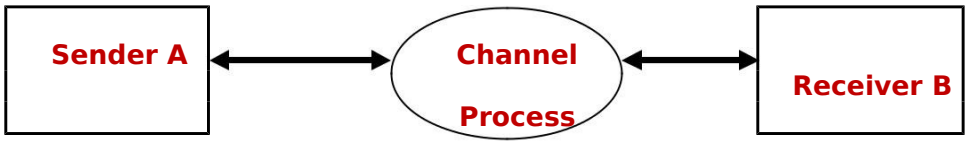
NAWP - Lab 3 - Simulation of Sliding Window Go-back-N protocol



Assume that two stations A and B are connected by a half-duplex point-to-point links, which implements the sliding window protocol Go-Back-N for flow control. The size of the window should be controllable by user specified parameter. Details are as follows:

- The data read from the local disk file at the sender looks like:
  struct msg { char data[20000]; };
- The unit of data passed between the sender-receiver routines is a frame, which is declared like:
  struct frame { int seq; int type; char payload[1500]; };
- The format for frame has a type field, a sequence number field, and up to 1500 bytes of data. 0 in the type field indicates a data frame, while a 1 indicates ACK frame. The actual frame size is a user input.
- The range of the sequence number depends on the sender window size.
- Receive window size is 1, i.e., the receiver does not accept out-of-order frames.
- The Channel Process selectively drops data/ACKs frame. This is characterized by a single parameter, the *loss rate p* (*0 <= p <= 1*). The process functions as follows. Let *p* represents the probability with which a received frame will be dropped and is a user input. For every frame received, generates a random number *v* between 0 and 1. If *v* > *p*, this frame can be transmitted; otherwise, this frame is dropped.
- The Delay() routine at the receiver models the delays which occur as part of frame processing, ACK generation, etc. The delays are uniformly distributed between <minRTT> and <maxRTT>. The delay is simulated as associated with the acknowledgements as follows. Generate a random number *u* between 0 and 1 when an acknowledgement is ready. Then, simply delay the transmission of this acknowledgement by *minRTT + (maxRTT-minRTT)u* amount of time.
- The Timer() routine implement s a single timer at the sender with a timeout value = <minRTT>*window size. When the timer expires, the sender retransmits all outstanding frames.
- Print appropriate messages whenever an event occurs at sender or receiver (data frame/ACK arrival, or a timer interrupt) as well as any action taken in response. Upon receiving a data frame in order, the program prints "RxDXX" where "XX" is the sequence number received. Upon dropping an out-of-order data frame, the program prints "DrDXX". Upon sending any acknowledgement, the program prints "TxAXX".
- Use UDP sockets to communicate between the processes.
- Test the protocol to transfer a file content read from the local disk which will be divided to many frames The sender is invoked by: **$sender <Receiver IP> <Filename> <Frame Size> <minRTT> <N> <p>** and receiver is invoked by: **$receiver <minRTT> <maxRTT>**
- Conduct the simulations with different scenarios and perform the following:
  - Set p to zero and frame size to 1024 bytes. Vary the window size N from 1 to 16 (suggested increments: 1, 2, 4, 8, and 16). Draw a graph with window size on the x-axis and the time required to transfer the file on the y-axis. Comment on your results.
  - Set p to zero and N to 8. Vary the frame size from 64 bytes to 1024 bytes (suggested increments: 64, 128, 256, 512, and 1024). Draw a graph with frame size on the x-axis and the time required to transfer the file on the y-axis. Comment on your results.
  - Set frame size to 1024 bytes and N to 8. Consider the following values of p: 0, 0.002, 0.004, 0.008, 0.016, 0.032, and 0.064. Draw a graph with p on the x-axis and the time required to transfer the file on the y-axis. Comment on your results.