**Program 2 Sliding Window Description & Test Result**

Eunmin Lee

|  |  |
| --- | --- |
| **STOP-and-Wait**  When the receiver(server) and sender(client) are connected, the sender sends message to ACK. In this program, it sends message[] which contains the sequence number at message[0].  Once the receiver gets the message, receiver checks if the message contains the sequence number they were waiting for, if it’s the message they were waiting for, they send the ACK back to the sender. The sender will receive ACK back, then the sender start sends the packet with the next sequence number.  Case 1:  The receiver and sender send and receive messages and ACK without a problem.  Case2:  The receiver successfully received the message from the sender, however, ACK was dropped on the way to go to the sender.  Case 3:  The receiver didn’t get the message from the senders, so it doesn’t send ACK.  In case2 and 3, the sender cannot determine whether the receiver already got the message or not. The sender just knows that it didn’t get the packet in 1500usec, then it declares a timeout, and sends the packet again to the receiver. |  |
| **Stop-and-wait Data:**   |  |  |  | | --- | --- | --- | | Trial | Elapsed Time | Retransmission | | 1 | 13572857 | 375 | | 2 | 1839548 | 0 | | 3 | 1873123 | 0 | | 4 | 1835840 | 0 | | 5 | 1799960 | 0 | | AVE | 1837118 | 0 (Outlier was excluded) | |  |
| **Sliding Window**  The sender first sends message from the 0 to window number in a row, and wait for the receiver’s ACK.  The receiver has a buffer and bound the acceptable message with LFS <= message[0] < LAF. If the receiver receives the message smaller than LFR, it requests a message within the bound by sending ACK LFR.  When the sender received ACK larger than its LAR, the receiver knows that the receiver already got the pocket, and start sending from ACK + 4. The illustration skipped, but the buffer of the receiver is filled up to 12, so it sends ACK 12 to the sender.  The receiver will save the packet received to buffer. If the receiver gets packet less than LFR, then it will then LFR - 1 as ACK to request the sender to send the packet that the receiver hasn’t got.  The receiver wouldn’t send ACK back unless it gets the desired packet (message[0] == LFR). As soon as it gets the desired packet, the receiver will send ACK with the largest seq it buffered. (if there is black in between largest, it will send the largest before the blank), and move LFR and LAF to the ACK it sent + 1.  **Sliding Window Data:**  When the window size is 1, it shows similar performance as stop-and-wait protocol    This graph was generated with 5 trials. In the program, if bool excelFile is set true, the program will print out .cvs file including window size (1-30), elapsed time, and retransmission. The 3 outlier where the retransmission is over 13000 is excluded.  The elapsed time shows exponential decreases as the window size increases. When it’s over the size of 10, the elapsed time doesn’t show much difference. As long as the window size is 1 <, the transmission shows better performance than stop-and-wait protocol. |  |