

Wesley Kepke

CPE 400 – Computer Communication Networks

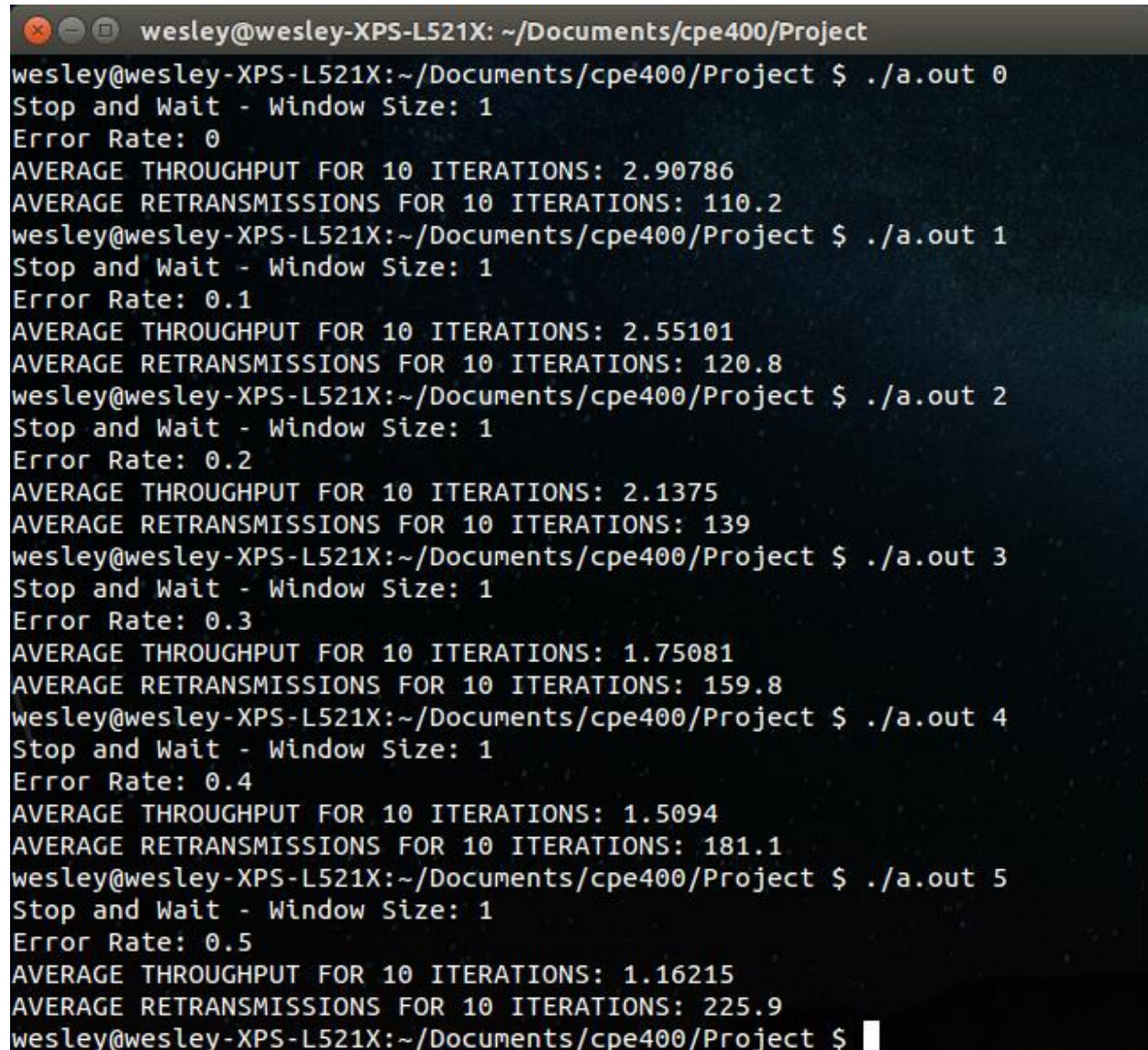
Project 1

Dr. Shamik Sengupta – October 8, 2014

**Assumptions – THIS PROGRAM WAS WRITTEN IN C++, COMPILED WITH G++ IN A LINUX ENVIRONMENT!**

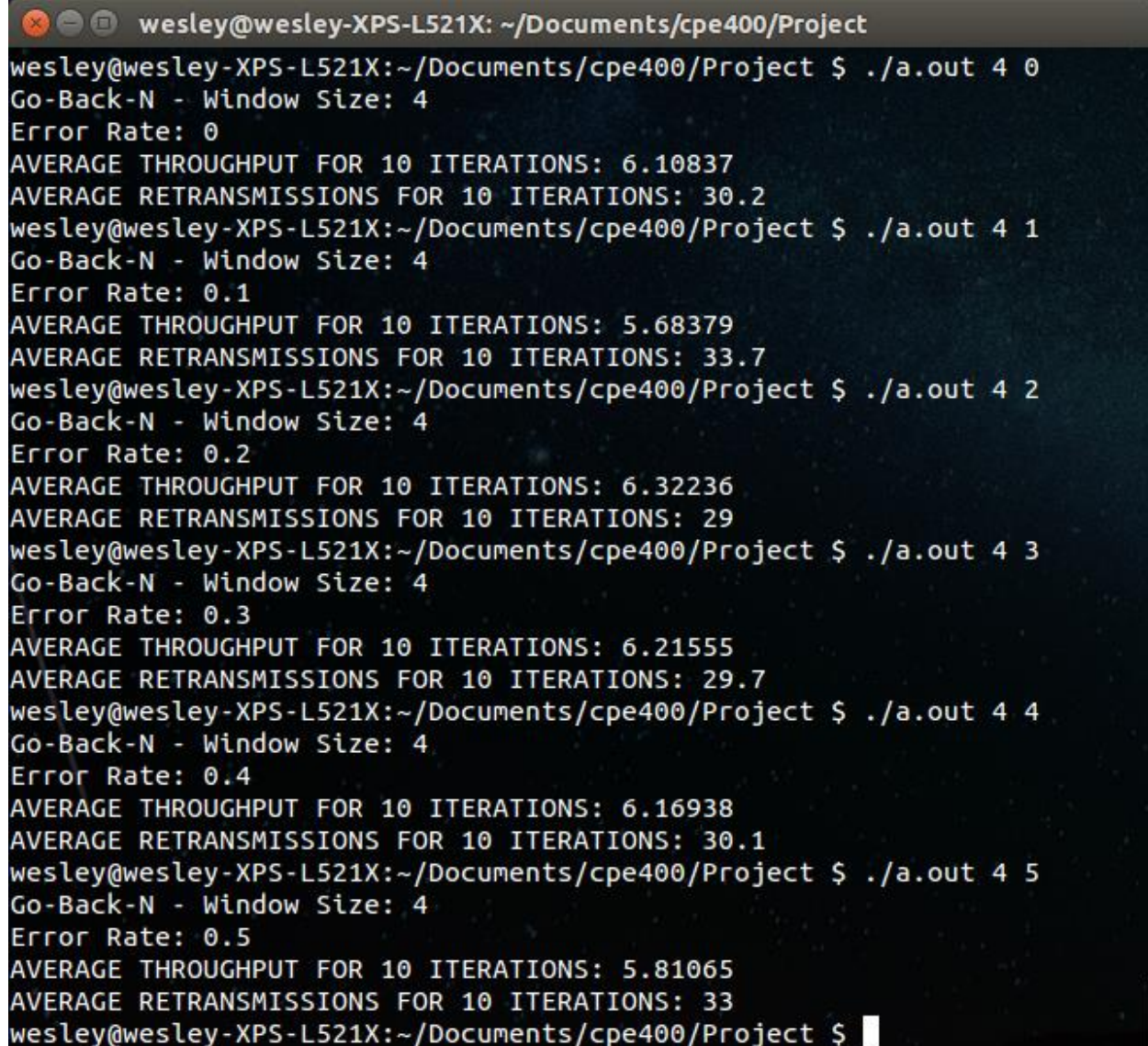
- Each of the 100 packets is 100 bytes.
- Each RTT is a random value between 10-50 (ms) with a timeout of 45 (ms).
- My RTT times did not include any type of delay or other irregularities in transmission time (I wasn't sure how to simulate this without a parallel environment).
- Packets are not sent in a parallel manner like they would be in a real network (it would have been intriguing to have a sender and a receiver process though).

**Stop and Wait:**



```
wesley@wesley-XPS-L521X: ~/Documents/cpe400/Project
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $ ./a.out 0
Stop and Wait - Window Size: 1
Error Rate: 0
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 2.90786
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 110.2
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $ ./a.out 1
Stop and Wait - Window Size: 1
Error Rate: 0.1
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 2.55101
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 120.8
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $ ./a.out 2
Stop and Wait - Window Size: 1
Error Rate: 0.2
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 2.1375
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 139
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $ ./a.out 3
Stop and Wait - Window Size: 1
Error Rate: 0.3
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 1.75081
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 159.8
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $ ./a.out 4
Stop and Wait - Window Size: 1
Error Rate: 0.4
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 1.5094
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 181.1
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $ ./a.out 5
Stop and Wait - Window Size: 1
Error Rate: 0.5
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 1.16215
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 225.9
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $
```

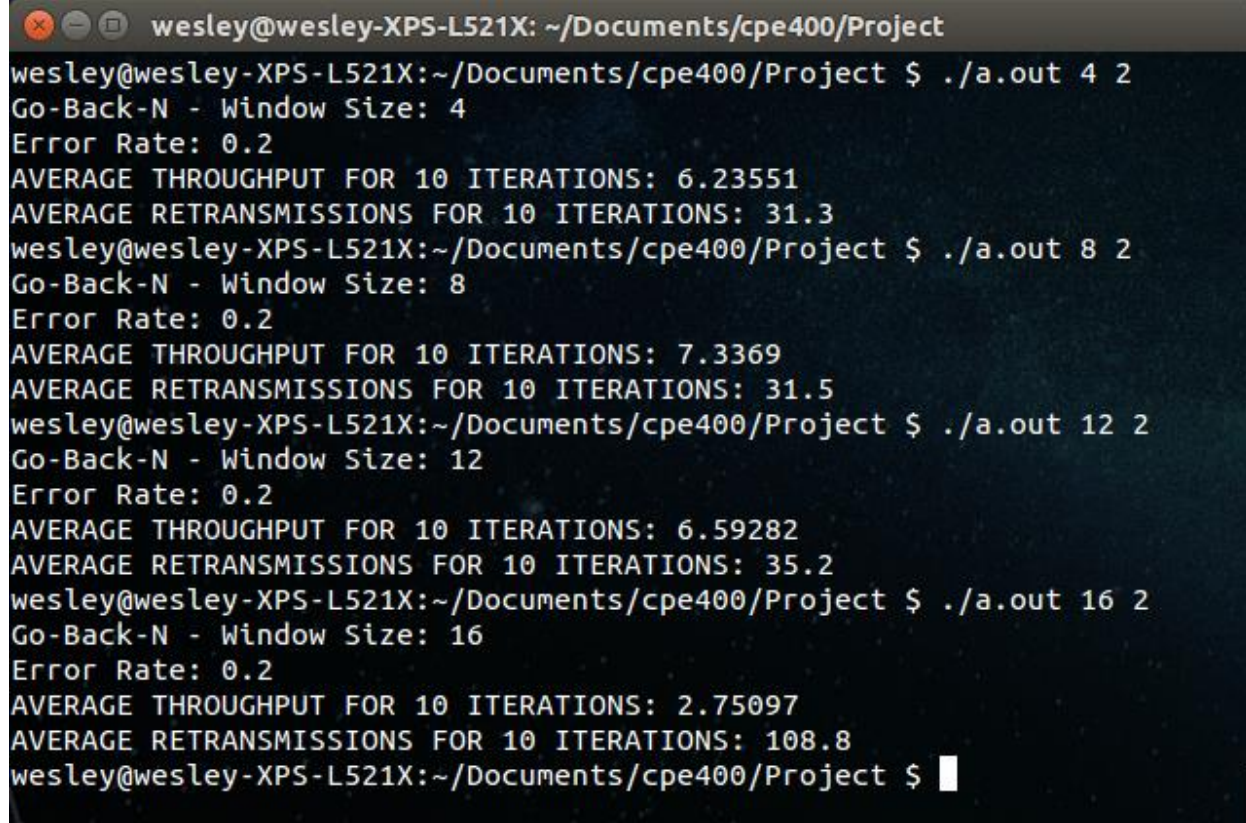
- This screenshot shows the output of my stop and wait program with varying error rates. Note that since this is the stop and wait protocol, the window size remains at 1.

Go-Back-N:A terminal window titled 'wesley@wesley-XPS-L521X: ~/Documents/cpe400/Project' displays the output of a Go-Back-N program. The program is executed with a constant window size of 4 and varying error rates from 0 to 0.5. For each error rate, the output includes the error rate itself, the average throughput for 10 iterations, and the average retransmissions for 10 iterations. The throughput values are relatively stable, ranging from approximately 5.68 to 6.32. The retransmission values increase as the error rate increases, starting at 30.2 for 0 error rate and reaching 33 for 0.5 error rate.

```
wesley@wesley-XPS-L521X: ~/Documents/cpe400/Project
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $ ./a.out 4 0
Go-Back-N - Window Size: 4
Error Rate: 0
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 6.10837
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 30.2
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $ ./a.out 4 1
Go-Back-N - Window Size: 4
Error Rate: 0.1
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 5.68379
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 33.7
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $ ./a.out 4 2
Go-Back-N - Window Size: 4
Error Rate: 0.2
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 6.32236
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 29
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $ ./a.out 4 3
Go-Back-N - Window Size: 4
Error Rate: 0.3
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 6.21555
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 29.7
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $ ./a.out 4 4
Go-Back-N - Window Size: 4
Error Rate: 0.4
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 6.16938
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 30.1
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $ ./a.out 4 5
Go-Back-N - Window Size: 4
Error Rate: 0.5
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 5.81065
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 33
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $
```

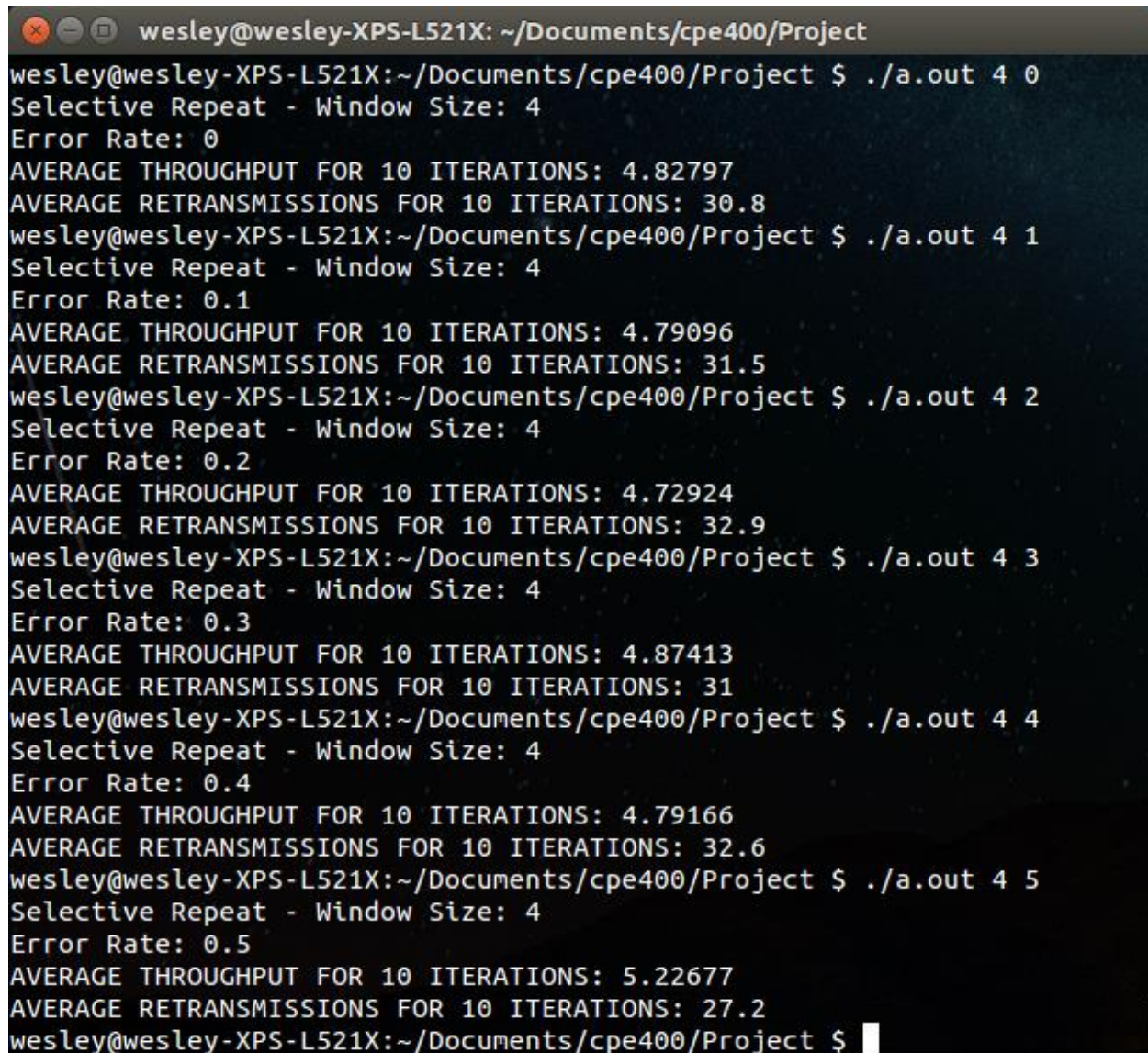
- This screenshot shows the output of my go-back-n program with a constant window size of 4 and varying error rates.



**Go-Back-N (continued):**A terminal window with a dark background and light-colored text. The title bar shows the user 'wesley' on a machine named 'wesley-XPS-L521X' in the directory '~/Documents/cpe400/Project'. The terminal displays the output of a program named 'a.out' for four different window sizes: 4, 8, 12, and 16. For each window size, the program reports an error rate of 0.2, the average throughput for 10 iterations, and the average retransmissions for 10 iterations. The throughput increases with window size, while retransmissions remain relatively constant. The terminal text is as follows:

```
wesley@wesley-XPS-L521X: ~/Documents/cpe400/Project
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $ ./a.out 4 2
Go-Back-N - Window Size: 4
Error Rate: 0.2
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 6.23551
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 31.3
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $ ./a.out 8 2
Go-Back-N - Window Size: 8
Error Rate: 0.2
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 7.3369
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 31.5
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $ ./a.out 12 2
Go-Back-N - Window Size: 12
Error Rate: 0.2
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 6.59282
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 35.2
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $ ./a.out 16 2
Go-Back-N - Window Size: 16
Error Rate: 0.2
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 2.75097
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 108.8
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $
```

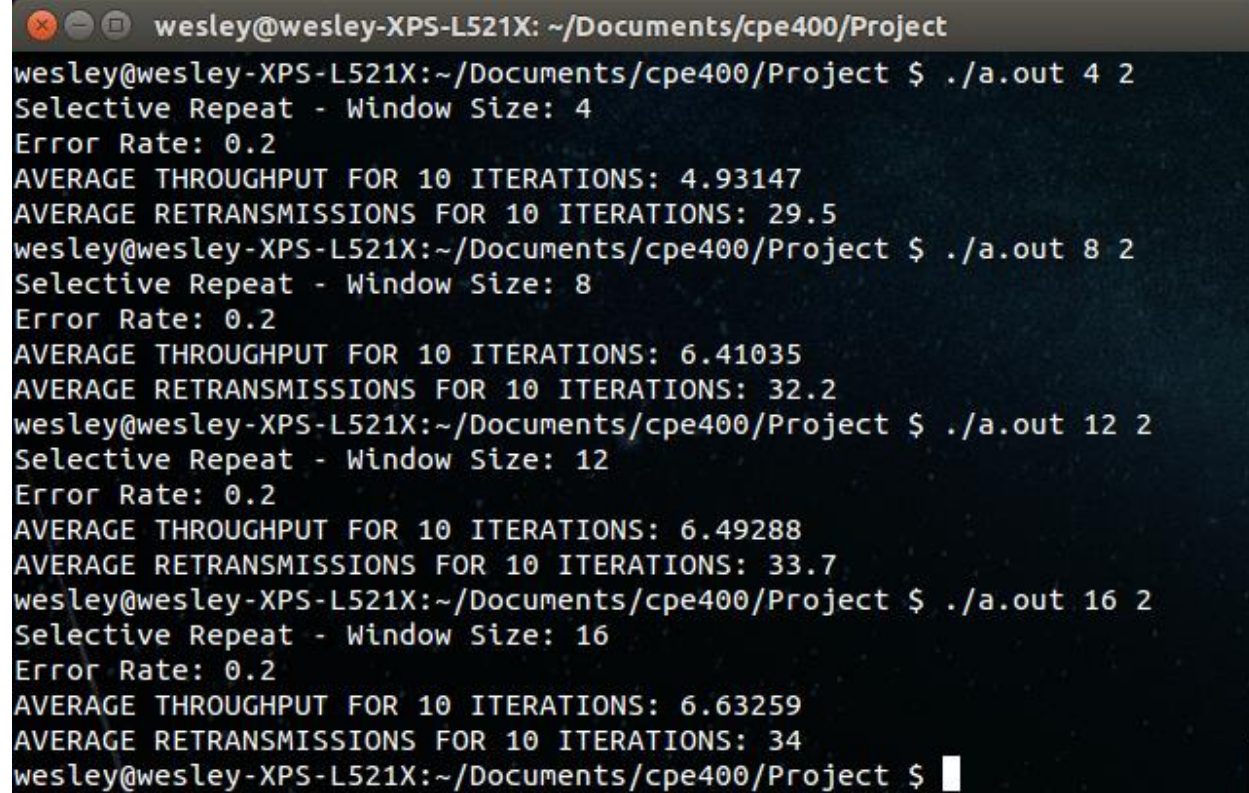
- This screenshot shows the output of my go-back-n program with a constant error rate of 0.2 and varying window sizes.

Selective Repeat:

```
wesley@wesley-XPS-L521X: ~/Documents/cpe400/Project
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $ ./a.out 4 0
Selective Repeat - Window Size: 4
Error Rate: 0
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 4.82797
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 30.8
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $ ./a.out 4 1
Selective Repeat - Window Size: 4
Error Rate: 0.1
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 4.79096
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 31.5
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $ ./a.out 4 2
Selective Repeat - Window Size: 4
Error Rate: 0.2
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 4.72924
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 32.9
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $ ./a.out 4 3
Selective Repeat - Window Size: 4
Error Rate: 0.3
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 4.87413
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 31
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $ ./a.out 4 4
Selective Repeat - Window Size: 4
Error Rate: 0.4
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 4.79166
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 32.6
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $ ./a.out 4 5
Selective Repeat - Window Size: 4
Error Rate: 0.5
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 5.22677
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 27.2
wesley@wesley-XPS-L521X:~/Documents/cpe400/Project $
```

- This screenshot shows the output of my selective repeat program with a constant window size of 4 and varying error rates.



Selective Repeat (continued):A terminal window with a dark background and light-colored text. The title bar at the top reads 'wesley@wesley-XPS-L521X: ~/Documents/cpe400/Project'. The terminal shows four separate runs of a program, each for a different window size (4, 8, 12, and 16). Each run displays the error rate (0.2), average throughput, and average retransmissions for 10 iterations. The output for each run is as follows:  
Run 1 (Window Size: 4):  
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 4.93147  
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 29.5  
Run 2 (Window Size: 8):  
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 6.41035  
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 32.2  
Run 3 (Window Size: 12):  
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 6.49288  
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 33.7  
Run 4 (Window Size: 16):  
AVERAGE THROUGHPUT FOR 10 ITERATIONS: 6.63259  
AVERAGE RETRANSMISSIONS FOR 10 ITERATIONS: 34  
The terminal ends with the prompt 'wesley@wesley-XPS-L521X:~/Documents/cpe400/Project \$' and a cursor.

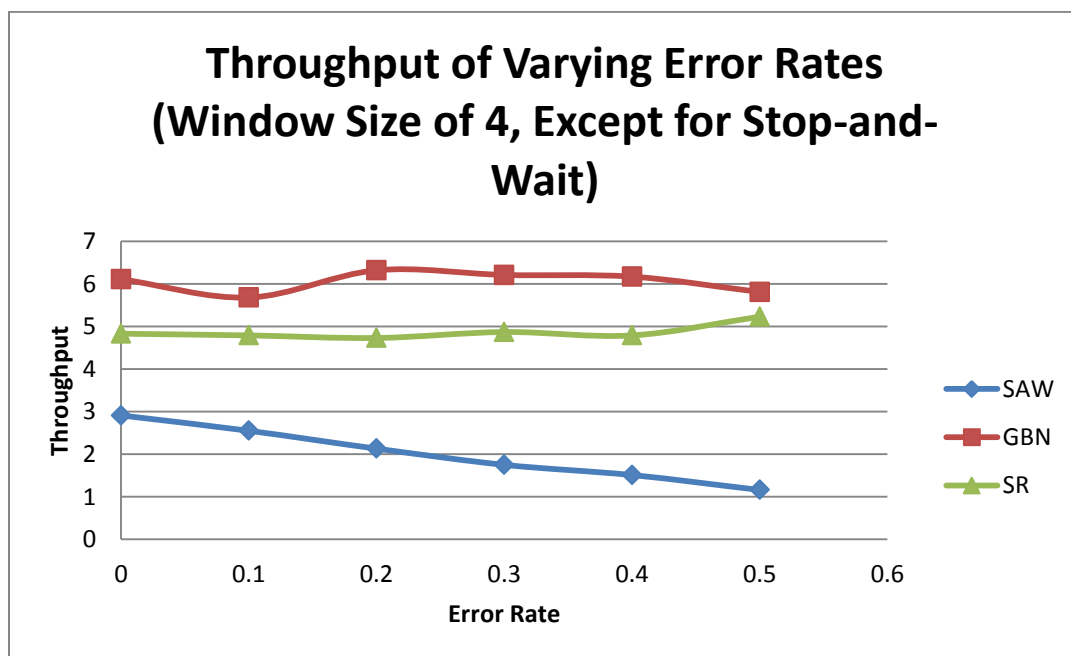
- This screenshot shows the output of my selective repeat program with a constant error rate of 0.2 and varying window sizes.

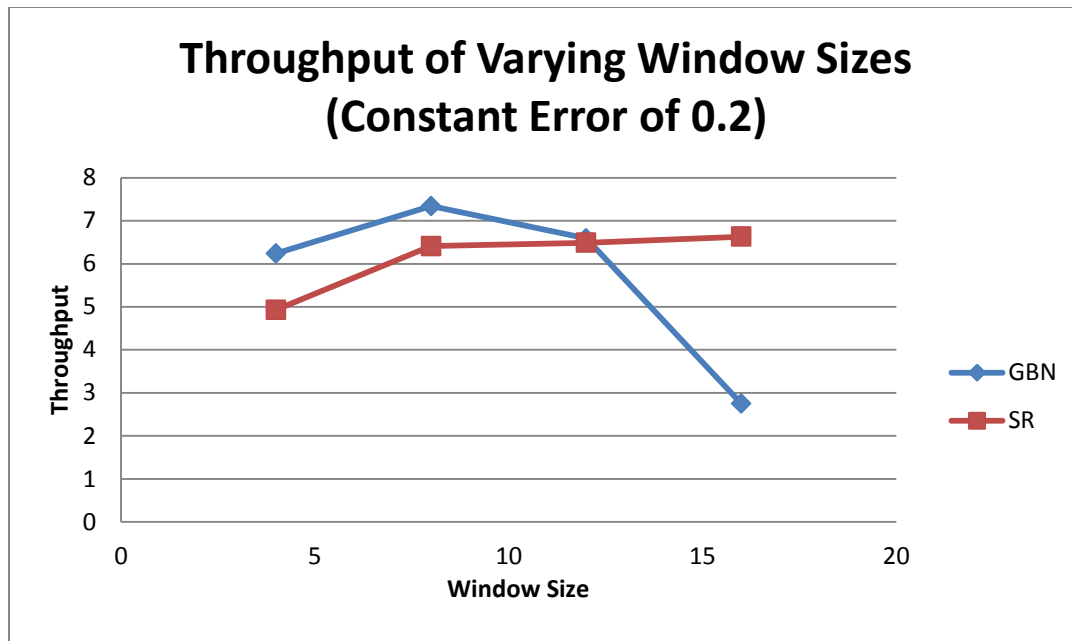
### Questions/Graphs:

**Note:** The data for each point on the following graphs was acquired by running each simulation 15 times and taking the average of the point of interest (either throughput or retransmissions).

A. The following two graphs represent the throughput results of all three protocols with respect to varying packet error rate and varying window size.

- Note: I have included the throughput of varying error rates of my stop-and-wait program in the same graph as the other two protocols (the first graph below). Even though the other two protocols have a window size of 4, the reader should take note that the data in the graph is representative of stop-and-wait's window size of 1.
- As we can see from the first graph below, with a constant window size of 4, the go-back-n protocol produces the highest throughput. In fact, we can see from the output of my programs that the average throughput for the go-back-n protocol is consistently higher than the other protocols.
- As we can see from the second graph below, with a constant error rate of 0.2, the go-back-n protocol seems to perform better than the selective repeat protocol when the window sizes are smaller. Furthermore, we notice that as the go-back-n protocol is simulated with a window size of 16, the throughput is quite small. This is to be expected, especially since we discussed in class that as the window size of go-back-n increases, we have more packets in the pipeline. Because of this, the probability of a single packet failing, and thus causing many other packets to be resent, is fairly high (a majority of these packets will be sent unnecessarily).



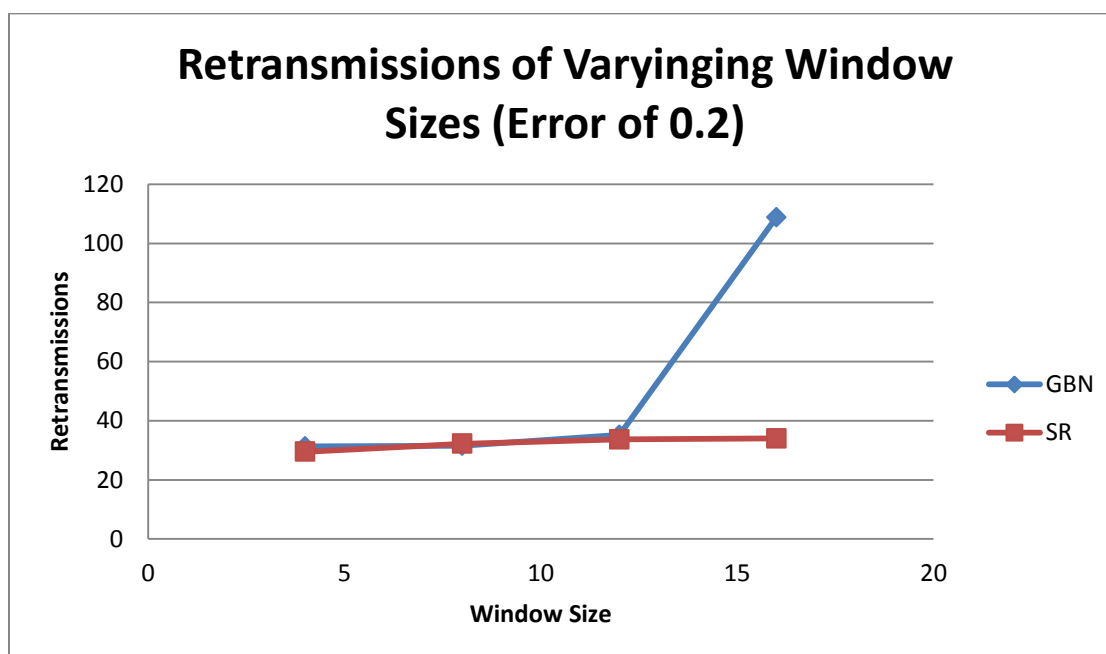
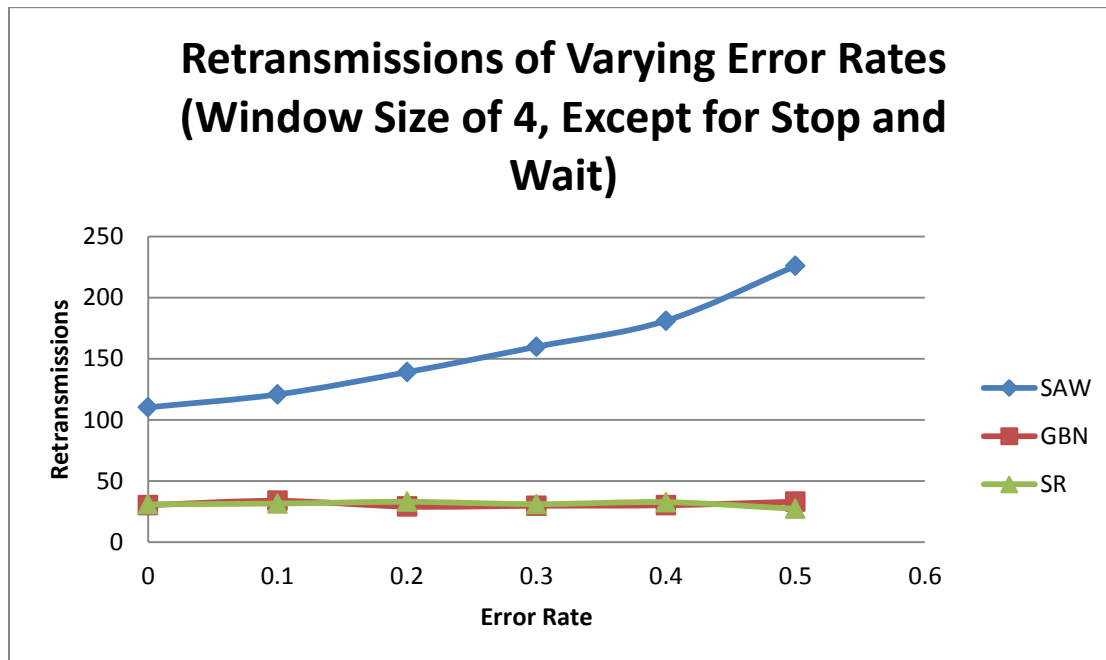


- (Part A – continued) Finally, Dr. Sengupta asked if the number of packets in the three programs is increased to 1000 instead of 100, will the trend be the same? The answer to this question is yes, and the data we would acquire from running the programs with 1000 packets instead of 100 packets would be more realistic since networks are obviously more complicated than 100 packets being sent at any particular given point in time.
- B. The following two graphs represent the retransmission results of all three protocols with respect to varying packet error rate and varying window size.
- According to the first of my two graphs below, with a constant window size of 4 (with the exception of stop-and-wait with its window size of 1), the protocol that had the highest number of retransmissions was the aforementioned stop-and-wait protocol. As expected, as the error rate increases. However, one interesting trend I would like to acknowledge is that both my go-back-n and selective repeat programs appear to have relatively constant retransmission rates, even as the error rate increases. I am suspicious of this activity despite the fact that I have double checked both of these programs for validity and that I have data that demonstrates that their throughput changes accordingly.
  - As we can see from the second graph, the number of retransmissions continues to remain relatively steadfast as the error rate was held constant at 0.2. One curious yet blatantly obvious observation in my graph lies at the window size of 16 for my go-back-n program. As the reader should be able to see, there is a sudden spike in the number of retransmissions in the program. Perhaps this was simply an outlier, but



nonetheless, anomalies can occur in a real network and so I have chosen not to omit this data, simply because it's representative of a true go-back-n protocol.

- Finally, Dr. Sengupta has asked us to once again reconsider the case where the number of packets has been increased to 1000 instead of 100. After running my programs and observing the results, I noticed that the number of retransmissions increased in proportion to the new amount of packets. Again, this information was to be expected and I feel confident with the results of my programs.



- C. Often times, there will be instances where “erroneous receiving” may occur, a side-effect of having a window size with a finite sequence number range. However, this can be avoided if and only if we structure our window size such that it is less than or equal to half the size of the range of possible sequence numbers. For example, consider the following diagram:

Sender	“Curtain”	Receiver
0 1 2 3 4 5 6 7 0 1 2 3 4	Pkt_0	0 1 2 3 4 5 6 7 0 1 2 3 4
0 1 2 3 4 5 6 7 0 1 2 3 4	Pkt_1	0 1 2 3 4 5 6 7 0 1 2 3 4
0 1 2 3 4 5 6 7 0 1 2 3 4	Pkt_2	0 1 2 3 4 5 6 7 0 1 2 3 4
0 1 2 3 4 5 6 7 0 1 2 3 4	Pkt_3	0 1 2 3 4 5 6 7 0 1 2 3 4
0 1 2 3 4 5 6 7 0 1 2 3 4	Ack_0	0 1 2 3 4 5 6 7 0 1 2 3 4
0 1 2 3 4 5 6 7 0 1 2 3 4	X (packet lost)	0 1 2 3 4 5 6 7 0 1 2 3 4
0 1 2 3 4 5 6 7 0 1 2 3 4	Ack_2	0 1 2 3 4 5 6 7 0 1 2 3 4
0 1 2 3 4 5 6 7 0 1 2 3 4	Ack_3	0 1 2 3 4 5 6 7 0 1 2 3 4
0 1 2 3 4 5 6 7 0 1 2 3 4	Pkt_2 TIMEOUT	0 1 2 3 4 5 6 7 0 1 2 3 4
0 1 2 3 4 5 6 7 0 1 2 3 4	Pkt_1	0 1 2 3 4 5 6 7 0 1 2 3 4
0 1 2 3 4 5 6 7 0 1 2 3 4	Ack_3 (Cumulative Ack)	0 1 2 3 4 5 6 7 0 1 2 3 4
0 1 2 3 4 5 6 7 0 1 2 3 4	Pkt_4	0 1 2 3 4 5 6 7 0 1 2 3 4
0 1 2 3 4 5 6 7 0 1 2 3 4	Pkt_5	0 1 2 3 4 5 6 7 0 1 2 3 4
0 1 2 3 4 5 6 7 0 1 2 3 4	Pkt_6	0 1 2 3 4 5 6 7 0 1 2 3 4
0 1 2 3 4 5 6 7 0 1 2 3 4	Pkt_7	0 1 2 3 4 5 6 7 0 1 2 3 4
0 1 2 3 4 5 6 7 0 1 2 3 4	Ack_4	0 1 2 3 4 5 6 7 0 1 2 3 4
0 1 2 3 4 5 6 7 0 1 2 3 4	Ack_5	0 1 2 3 4 5 6 7 0 1 2 3 4
0 1 2 3 4 5 6 7 0 1 2 3 4	Ack_6	0 1 2 3 4 5 6 7 0 1 2 3 4
0 1 2 3 4 5 6 7 0 1 2 3 4	Ack_7	0 1 2 3 4 5 6 7 0 1 2 3 4

- The table above shows that there will never be an overlapping sequence number in the buffers of the receiver and the sender.
- Dr. Sengupta has asked us what would be the ideal sequence number range for a varying window transmitter of 4-16 and about how many bits are required for the sequence number field.
  - If we follow the criteria above, our sequence numbers for window sizes of 4, 8, 12, and 16 should be 8, 16, 24, and 32 (respectively).
  - Additionally, we’ll need 3 ( $2^3 = 8$ ) bits for a sequence number range of 8 on a transmitter window size of 4.
    - Likewise, we’ll need 4 ( $2^4 = 16$ ) bits for a sequence number range of 16 on a transmitter windows size of 8.
    - Likewise, we’ll need 5 ( $2^5 = 32$ ) bits for a sequence number range of 24 and 32 on a transmitter windows sizes of 12 and 16, respectively.