Program 2: Sliding Window Algorithm

Raghu Tirumala

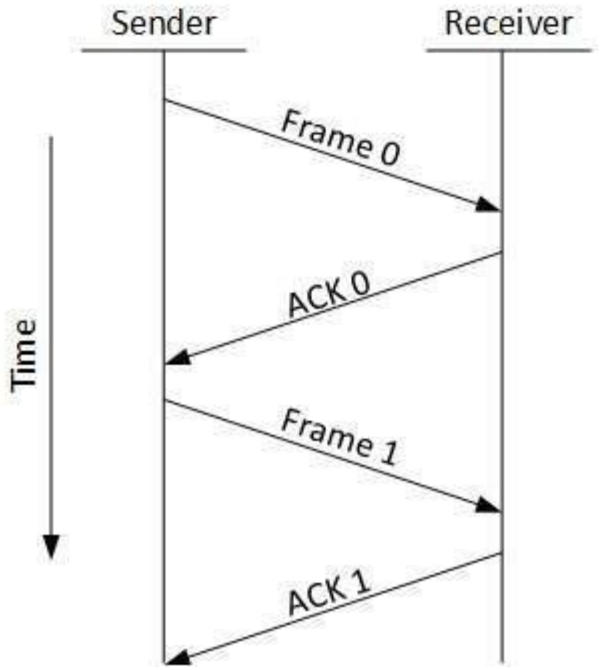
Documentation

This program transmits messages between a server and client using a UDP socket with 3 different sending algorithms.

1. Unreliable test
   - Client sends 20,000 udp packets to server, does not wait for acknowledgement
2. Stop-and-wait
   - Client sends udp packet to server waits to receive acknowledge before sending next packet. If ack is not received before 1500 us has passed, client retransmits packet.
3. Sliding window
   - Sends and receives udp packets using sliding window algorithm

Stop and Wait

1. UDP packet sent to server by client.
2. Client waits for ack until a timeout of 1500 us.
3. If ack arrives send next packet.
4. If timeout occurs resend current packet.



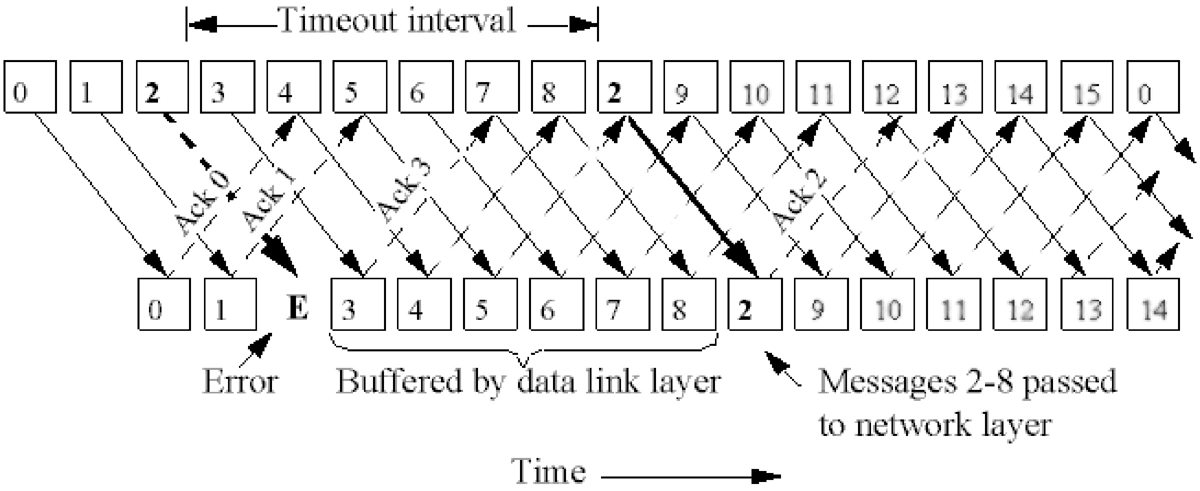
Sliding Window

Client

Assigns sequence number to each frame and keeps sending frames until the number of in transit messages is equal to the window size. If this is the case the client waits till a timeout for an acknowledge from the server. If an ack does not arrive before the timeout the client resends all the unacknowledged frames in the window.

Server

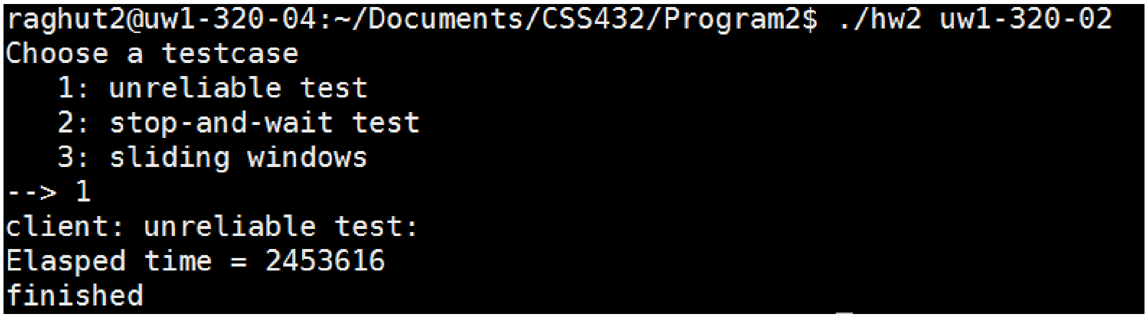
Receives messages from the client and uses the sequence numbers of the messages to update a Boolean array. The server then sends all acknowledgements for the messages it has received so far within the window.



Source:
http://www.site.uottawa.ca/~elsaddik/abedweb/applets/Applets/Sliding_Window/sample2.gif

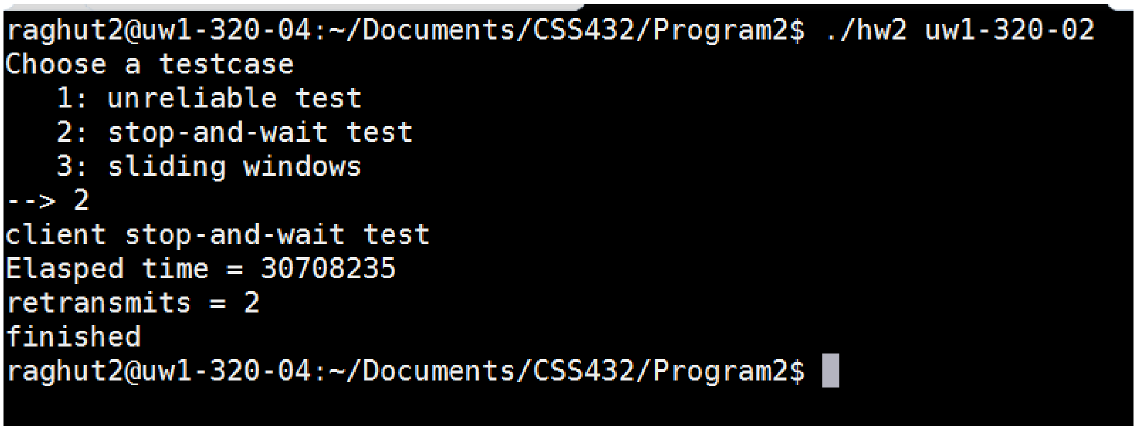
Execution Output

Case 1: Unreliable Test



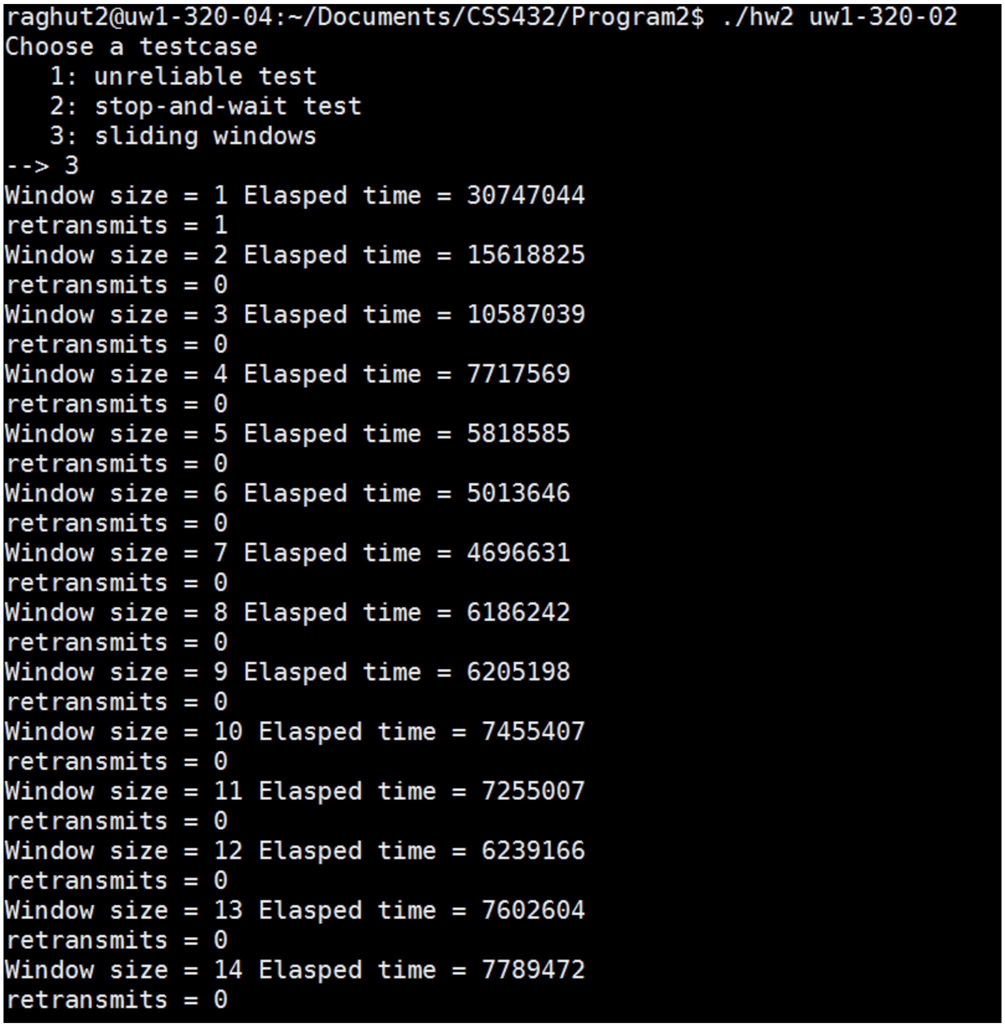
```
raghut2@uw1-320-04:~/Documents/CSS432/Program2$ ./hw2 uw1-320-02
Choose a testcase
   1: unreliable test
   2: stop-and-wait test
   3: sliding windows
--> 1
client: unreliable test:
Elasped time = 2453616
finished
```

Case 2: Stop and Wait



```
raghut2@uw1-320-04:~/Documents/CSS432/Program2$ ./hw2 uw1-320-02
Choose a testcase
   1: unreliable test
   2: stop-and-wait test
   3: sliding windows
--> 2
client stop-and-wait test
Elasped time = 30708235
retransmits = 2
finished
raghut2@uw1-320-04:~/Documents/CSS432/Program2$
```

Case 3: Sliding Window



```
raghut2@uw1-320-04:~/Documents/CSS432/Program2$ ./hw2 uw1-320-02
Choose a testcase
   1: unreliable test
   2: stop-and-wait test
   3: sliding windows
--> 3
Window size = 1 Elasped time = 30747044
retransmits = 1
Window size = 2 Elasped time = 15618825
retransmits = 0
Window size = 3 Elasped time = 10587039
retransmits = 0
Window size = 4 Elasped time = 7717569
retransmits = 0
Window size = 5 Elasped time = 5818585
retransmits = 0
Window size = 6 Elasped time = 5013646
retransmits = 0
Window size = 7 Elasped time = 4696631
retransmits = 0
Window size = 8 Elasped time = 6186242
retransmits = 0
Window size = 9 Elasped time = 6205198
retransmits = 0
Window size = 10 Elasped time = 7455407
retransmits = 0
Window size = 11 Elasped time = 7255007
retransmits = 0
Window size = 12 Elasped time = 6239166
retransmits = 0
Window size = 13 Elasped time = 7602604
retransmits = 0
Window size = 14 Elasped time = 7789472
retransmits = 0
```

Full Text Output:

```
Window size = 1 Elasped time = 30747044
retransmits = 1
Window size = 2 Elasped time = 15618825
retransmits = 0
Window size = 3 Elasped time = 10587039
retransmits = 0
Window size = 4 Elasped time = 7717569
retransmits = 0
Window size = 5 Elasped time = 5818585
retransmits = 0
Window size = 6 Elasped time = 5013646
retransmits = 0
Window size = 7 Elasped time = 4696631
retransmits = 0
Window size = 8 Elasped time = 6186242
```

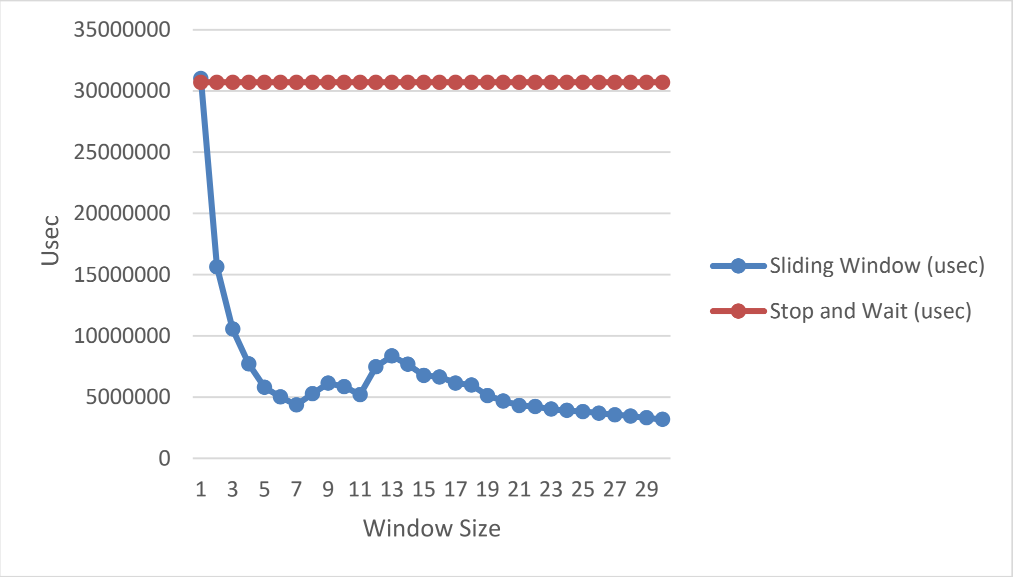
```
retransmits = 0
Window size = 9 Elasped time = 6205198
retransmits = 0
Window size = 10 Elasped time = 7455407
retransmits = 0
Window size = 11 Elasped time = 7255007
retransmits = 0
Window size = 12 Elasped time = 6239166
retransmits = 0
Window size = 13 Elasped time = 7602604
retransmits = 0
Window size = 14 Elasped time = 7789472
retransmits = 0
Window size = 15 Elasped time = 7484636
retransmits = 0
Window size = 16 Elasped time = 6617670
retransmits = 0
Window size = 17 Elasped time = 6388500
retransmits = 0
Window size = 18 Elasped time = 8026722
retransmits = 180
Window size = 19 Elasped time = 6035289
retransmits = 0
Window size = 20 Elasped time = 4639604
retransmits = 0
Window size = 21 Elasped time = 4403853
retransmits = 0
Window size = 22 Elasped time = 4233833
retransmits = 0
Window size = 23 Elasped time = 4043228
retransmits = 0
Window size = 24 Elasped time = 3919136
retransmits = 0
Window size = 25 Elasped time = 3775743
retransmits = 0
Window size = 26 Elasped time = 3691149
retransmits = 0
Window size = 27 Elasped time = 3566887
retransmits = 0
Window size = 28 Elasped time = 3443489
retransmits = 0
Window size = 29 Elasped time = 3277628
retransmits = 0
Window size = 30 Elasped time = 3221550
retransmits = 0
finished
```

Performance Results

The unreliable send was clearly the fastest method of sending messages with an average time of around 2,500,000 usec.

The stop and wait algorithm took much longer on average during my testing with a sending time of around 30,000,000 usec.

The sliding window time varied greatly based on the window size. The smaller window sizes had send times comparable to that of the stop and wait algorithm. The larger window sizes took only slightly longer than the unreliable send.



It is clear from the data that the sliding window algorithm seems to be getting faster the larger the window size for a 1gbps network.

The average ping time between uw1-320-02 and uw1-320-04 was 0.215 ms or 215 us.

RTT = 215 us

Bandwidth = 1gbps

RTT * bandwidth = 215 * 10^-6 sec * 1*10^9 bits/sec = 215 * 10^3 bits = 215000 bits.

Packet size = 1500bytes = 12000 bits

Theoretical max window size = 215000/12000 = 17.92 ≈ 18

The maximum window size from my data appeared to be around 26 but performance did not improve much from window size 20 onward.

I did not see any noticeable difference in the number of retransmitted messages between stop and wait and sliding window, mainly because retransmits did not happen very often.