CMPS 2301 Introduction to Computer Systems & Networking

Lab 10: Sliding Window/Bridging

Completed by Mauryan Uppalapati and Alex Olteanu

Objectives

- To practice networking algorithms.
- To understand the sliding window algorithm.
- To build a foundation for understanding routing

You may complete this lab individually or with one colleague from this class. If you work in a pair, state clearly both lab members at the top of your submission files.

For this lab, you may print and draw on this document by hand or illustrate it using word (or any other technology of your choice).

CMPS 2301 Introduction to Computer Systems & Networking

Lab 10: Sliding Window/Bridging

CMPS 2301 Introduction to Computer Systems & Networking

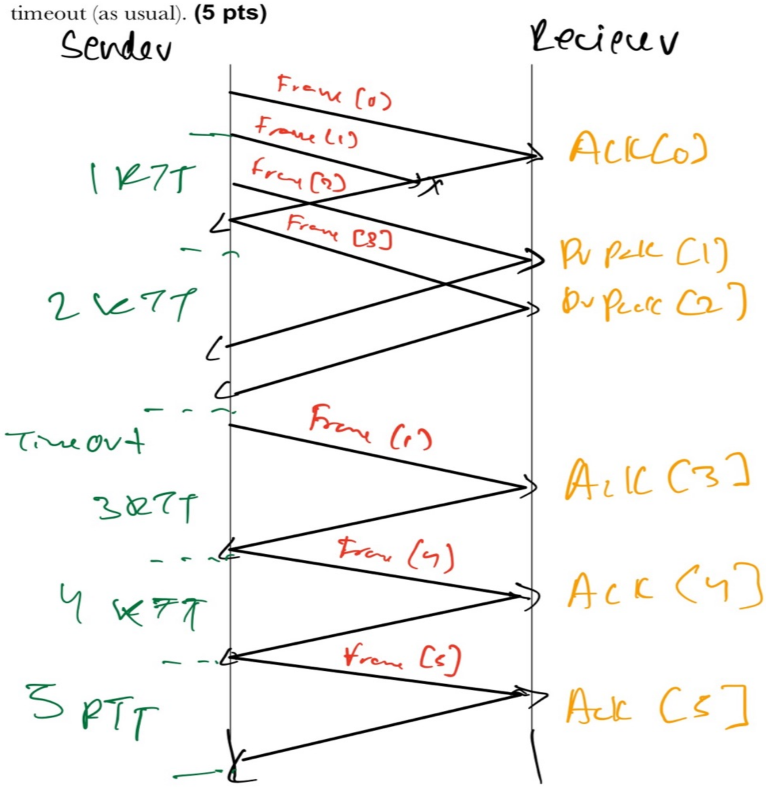
Lab 10: Sliding Window/Bridging

1 Reliable Transmission (15 pts)

1) Consider a variation of the Sliding Window algorithm. Assume the receiver sends a duplicate acknowledgment if it does not receive the expected frame. For example, it sends `DUP_ACK[2]` when it expects to see `Frame[2]` but receives `Frame[3]` instead. Also, the receiver sends a cumulative acknowledgment after it receives all the outstanding frames. For example, it sends `ACK[5]` when it receives the lost frame `Frame[2]` after it already received `Frame[3]`, `Frame[4]`, and `Frame[5]`. Use a timeout interval of about `2×RTT`.

Draw a timeline diagram for the sliding window algorithm with `SWS = RWS = 3` frames in the following two situations. Label all frames, acknowledgements, and round trip times for the dropped frame.

a) The sender wishes to transmit frames 0-5. Frame 1 is lost. Retransmission takes place upon timeout (as usual). **(5 pts)**



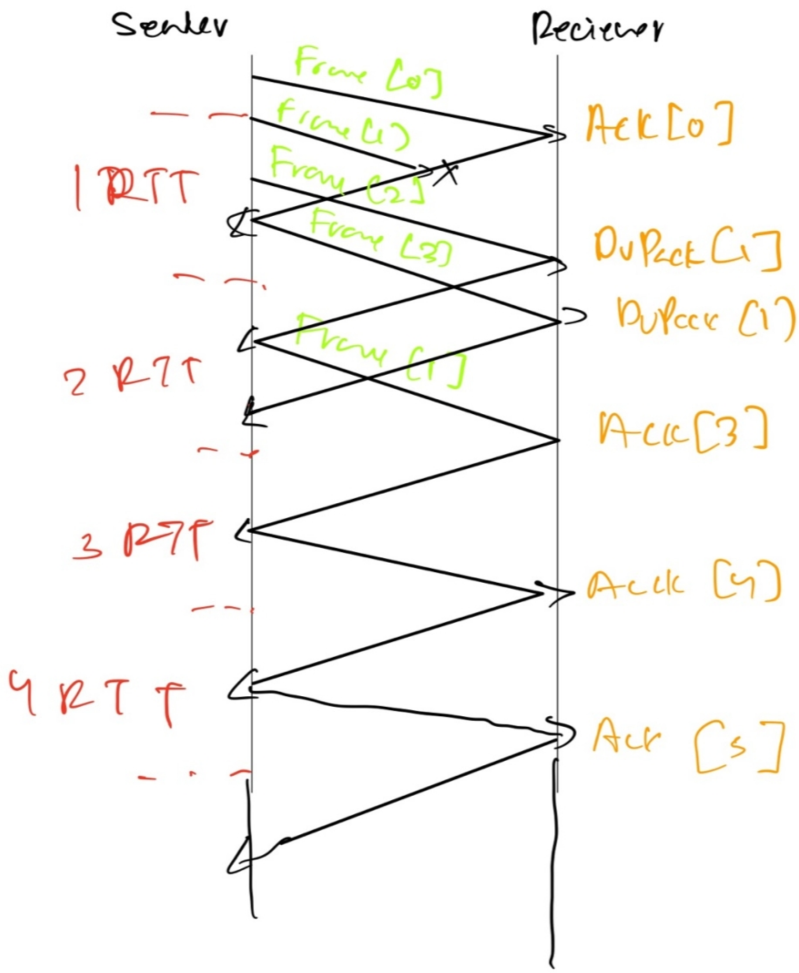
CMPS 2301 Introduction to Computer Systems & Networking

Lab 10: Sliding Window/Bridging

CMPS 2301 Introduction to Computer Systems & Networking

Lab 10: Sliding Window/Bridging

b) The sender wishes to transmit frames 0-5. Frame 1 is lost. Retransmission takes place either upon receipt of the first DUP_ACK or upon timeout. Does this scheme reduce the transaction time? (Note that some end-to-end protocols, such as variants of TCP, use similar schemes for fast retransmission.) **(5 pts)**



Lab 10: Sliding Window/Bridging

2) Suppose that we run the sliding window algorithm with `SWS = 5` and `RWS = 3`, and no out-of-order arrivals. As usual, each frame is marked with its sequence number, in order to let the receiver detect undelivered frames, and also order them. The sender maintains a counter variable that is used to keep sequence number. As you can imagine, this variable is not unbounded. It will eventually reach its maximum value and will have to wrap around. But what is the smallest value for `MaxSeqNum` that could possibly work? You can provide your number in a form of a number or a formula involving SWS and RWS. Justify your answer in a couple of sentences. **(5 pts)**

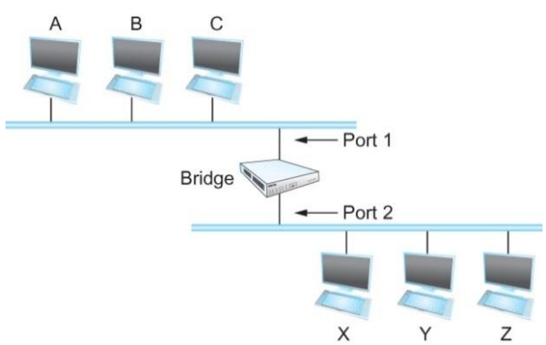
The smallest possible value of MaxSeqNum would have to be 8. This is since you cannot prove that the frame with the 8th index is on the receiving side if frame[0] is also arriving on the receiver side. The earliest frame you could receive is frame[6] - frame[8]. Furthermore frame[6] must be acknowledged and received while frame[5] is being delivered. All frames before frame[5] would be sent and due to the out-of-order arrival hypothesis, the first frame would no longer be able to arrive.

Lab 10: Sliding Window/Bridging

2 Bridging (10 pts)

A **local area network (LAN)** supplies networking capability to a group of computers in close proximity to each other such as in an office building, a school, or a home. Most LANs these days are supported by Ethernet technology. We will focus our attention on the earlier versions of Ethernet, which were based on broadcasting transmissions. A network would be a set of hosts connected to a common cable. Any signal placed on the Ethernet by a host would be broadcasted over the entire network, similarly how all radio stations broadcast the news and music over the shared medium (air). In that type of network, all hosts would be able to hear transmissions by all other hosts. (Imagine a lecture session with instructor and students. When instructor (host) transmits a frame "Luke Skywalker, where is your homework?" everyone can hear it but only Luke will act upon it).

That network would have physical size limitation, so in order to grow them, there would be a need to connect several of such networks together using bridges. (A bridge is essentially a LAN switch that forwards packets between shared-media LANs (Ethernets) that would forward the frames from one Ethernet to the other). For example, on the picture below, two LANs are connected with a bridge.



CMPS 2301 Introduction to Computer Systems & Networking

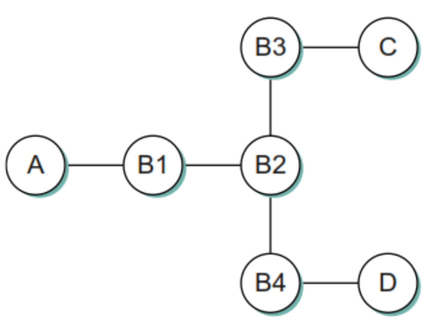
Lab 10: Sliding Window/Bridging

Learning Bridges

Observe that there is no need to forward all the frames that a bridge receives. For example, on the previous figure, when a frame from host A that is addressed to host B arrives on port 1, there is no need for the bridge to forward the frame out over port 2. How does a bridge come to learn on which port the various hosts reside? Can the bridge learn this information

by itself? The answer is yes. Bridge inspects source address of frames. If frames from A come in Port 1, Port 1 should be used to forward to A. When a bridge first boots, this table is empty. Entries are added over time. A timeout is associated with each entry to protect handle situations in which hosts are moved from one network to another. If the bridge receives a frame that is addressed to host not currently in the table, forward the frame out on all other ports. Strategy works fine if the extended LAN does not have a loop in it. If there is a loop, frames potentially loop through the extended LAN forever.

3) Consider the following arrangement of learning bridges:



Assuming all are initially empty, give the forwarding tables for each of the bridges B1 to B4 after the following transmissions:

CMPS 2301 Introduction to Computer Systems & Networking

Lab 10: Sliding Window/Bridging

Each forwarding table will maintain <**destination_address, interface**> pairs. Mark ports/interfaces with the unique neighbor reached directly from that port; that is, the ports for B1 are to be labeled "A" and "B2" and use these values in the forwarding table. **(5 pts)**

A sends to C.

Bridge	Destination,Port
B1	A,A
B2	A,B1
B3	A,B2
B3	C,C

C sends to A.

Bridge	Destination,Port
B3	C,C
B2	C,B3
B1	C,B2
B1	A,A

D sends to C.

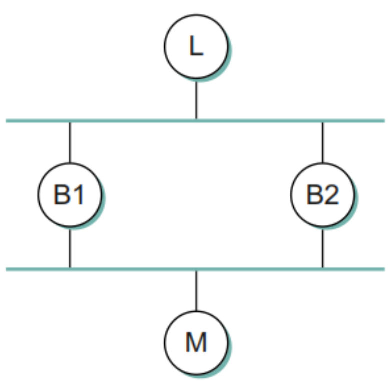
Bridge	Destination,Port
B4	D,D
B2	D,B4
B3	D,B2
B3	C,C

Lab 10: Sliding Window/Bridging

Forwarding Table

Bridge	Destination	Port
B1	A C	A B2
B2	A C D	B1 B3 B4
B3	A C D	B2 C B2
B4	A D	B2 D

4) Suppose learning bridges B1 and B2 form a loop:



Also assume that they do **not** implement the loop detection algorithm. Each bridge maintains a single table of <**address, interface**> pairs. Provide a short explanation describing what will happen if M sends to L? **(5 pts)**

If M attempts to send to L without a loop detection algorithm, then the data packet will be continuously forwarded in an infinite loop. This happens when information is routed through the same bridges over and over.