Kebi Hong

CS 455 PA 2

Design document

**Idea for designing methods:**

* **A\_output():** This method is called whenever the upper level wants to send a packet from A to B. We first construct a Packet with the data we received from the message sent by the upper level. Before we send the packet, we need to check that if the window has enough room to send this packet, so we put this packet in a queue first. If the window has room, we send the packet, otherwise, we wait.
* **A\_input():** A is going to receive ACKs from B. A received a correct ACK only if the ACK is uncorrupted. After that we will shift the window to receive new packets. We might send several duplicated packets before we received a correct ACK, and we would assume we received(drop) all the duplicated packets after we received an ACK. Therefore we will calculate the RTT for those packets with current time.
* **A\_timerinterrupt():**This routine will be called when A's timer expires. At that moment, we didn’t receive the ACK we want, and we would drop all the packets we sent. The Packet we want to send (in order) is still in the window, therefore, we retransmit all the packets in the window. If a packet is sent for the first time, we will add the time to the RTT.
* **bInput():** If B receives a corrupted packet, B will drop it. If B receives a packet that is out of order, B will drop it. B only receives a packet if it is uncorrupted, in order and packet’s sequence number = expected sequence number. Then we deliver this packet to level 5.
* **A\_init():** 
  + seqNum: This is initialed to be 0, and we increase it by 1 every time we send a packet, it will not excessed the window size.
  + Min\_Seq: The beginning of the window, it will increase by 1 every time A correctly receive an ACK.
  + max\_Seq: The ending of the window, it will increase by 1 every time A correctly receive an ACK. Min\_Seq + window\_size -1 = min\_Seq.
  + nexttoSend: The actual non-duplicated packet to send. It will increase by 1 when the upper level asks A to send a message.
  + Totalsent: initialed to 0, it will increase by 1 every time we send a packet.
* **B\_init():**
  + exptSeq: The sequence number of the packet the B is expected to receive. exptSeq will increase by 1 every time B sends a ACK to A.

**RTT VS. Loss & Corruption**

Number of messages to simulate: 1000

Average time between messages from sender's layer 5: 10

Window size: 8

Retransmission timeout: 20

Trace level: 2

Random seed: 2233

**RTT when loss = 0.0, 0.25, 0.5, 0.75 corruption = 0.0**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Loss rate = 0.0 | Loss rate = 0.25 | Loss rate = 0.5 |
| Total packet sent | 4087 | 4066 | 4098 |
| # original packet sent | 70 | 105 | 174 |
| # ACK sent | 62 | 97 | 166 |
| # packets sent to Level 5 | 62 | 97 | 166 |
| # Retransmission | 4017 | 3961 | 3924 |
| %packet corrupt | 0.0 | 0.0 | 0.0 |
| %packet loss | 0.98 | 0.97 | 0.96 |
| Average RTT | 1857.65 | 1723.34 | 1650.52 |

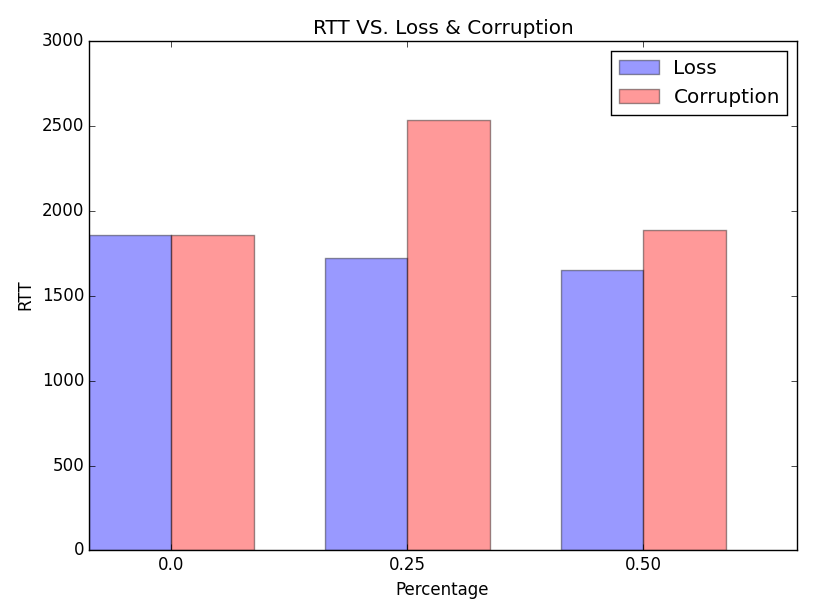
**RTT when loss = 0.0, 0.25, 0.5, 0.75 corruption = 0.25**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Loss rate = 0.0 | Loss rate = 0.25 | Loss rate = 0.5 |
| Total packet sent | 4110 | 4110 | 4113 |
| # original packet sent | 54 | 60 | 63 |
| # ACK sent | 46 | 52 | 63 |
| # packets sent to Level 5 | 46 | 52 | 63 |
| # Retransmission | 4056 | 4050 | 4050 |
| %packet corrupt | 0.90 | 0.89 | 0.89 |
| %packet loss | 0.86 | 0.86 | 0.85 |
| Average RTT | 2536.18 | 2645.93 | 1513.77 |

**RTT when loss = 0.0, 0.25, 0.5, 0.75 corruption = 0.5**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Loss rate = 0.0 | Loss rate = 0.25 | Loss rate = 0.5 |
| Total packet sent | 4111 | 4111 | 4111 |
| # original packet sent | 39 | 8 | 34 |
| # ACK sent | 34 | 8 | 34 |
| # packets sent to Level 5 | 34 | 8 | 34 |
| # Retransmission | 4072 | 4103 | 4077 |
| %packet corrupt | 0.96 | 0.99 | 0.96 |
| %packet loss | 0.74 | 0.75 | 0.75 |
| Average RTT | 1887.27 | NA | 1148.86 |

**Graph the Result: RTT VS. Loss & Corruption**

****